Release Note

Software Version 2.7.3

For AT-9900, AT-8900, SwitchBlade, AT-9800, AT-8800, Rapier, Rapier i, AT-8700XL, and AT-8600 Series Switches and AR400 and AR700 Series Routers

Introduction 4
Upgrading to Software Version 2.7.3 6
Overview of New Features 7
Multiple Spanning Tree Protocol (MSTP) 10
Software QoS 10
Link Access Control Protocol (LACP) 10
SNMPv3 11
Stacking 11
Port Authentication Enhancements 11
Configurable RADIUS timers 12
    New Command 12
    Modified Commands 12
RADIUS RSO 13
TACACS+ Authentication and Telnet 14
    New Commands 14
Text Message at Login 15
Remote Security Officer (RSO) Login 16
Interrupting Text Flow with the CLI 16
Support for Long File Names 17
    Upgrading to new software versions 17
    Regressing to Previous Software Versions 17
TFTP Load Over IPv6 18
    Modified commands 18
Extended Show Debug Command 19
    Modified commands 19
Temperature and Fan Monitoring 22
Telnet Session Limit and Timeout 23
Reverse Telnet 24
    Client command limitations 24
    New Commands 25
    Modified Command 27
Adding SNMP Management Stations by IP Address Range 27
Syslog Facility Override 28
    Modified Commands 28
DHCP Probing for IP Addresses 29
Classification By L4 Mask and Inner VLAN Settings 30
QoS Counters 32
Actions for QoS Traffic Classes and Flow Groups 36
L2 QoS Actions in Hardware Filters 38
IP DSCP Override 39
CPU Transmit User Priority Override 39

CPU Transmit Queue Priority Override 40
Changes to QoS on Trunked Ports 40
Enable/Disable Port Egress Queue and Flow Control 41
Disabling 10/100 Ports at the Hardware Level 43
Increase in VLAN Name Length 43
Protected VLANs and Private VLANs 44
Protected VLANs 44
Private VLANs on Rapier i and AT-8800 Switches 44
Private VLANs on AT-8900 and AT-9900 Switches 47
RSTP BPDU Loopback Detection 50
Virtual Bridge (VLAN) MIB 51
MAC Address Thrashing Protection for Ports 52
Recovery Disposal to PPPoE 53
TPAD Over TCP/IP Improvements 54
Ping and Trace Using DNS 56
Blackhole Routing 57
Hardware Equal Cost Multipath Routing (ECMP) 60
Hardware IP Route Learning Delay 61
Disable Source Routing 62
Modified Commands 62
Local Interfaces 63
Using the Local Interface in BGP 63
New commands 64
Modified commands 68
Use of Default Route by Proxy ARP 72
Modified Commands 72
Support for 5000 Next Hops 73
New Command 73
Modified Command 73
BGP Improvements 74
Soft Resetting of Modified Peers 74
Peer Templates 74
Extensions to Route Maps 74
Prefix Lists 74
Tagging Static Routes for Import Filtering 74
Handling Spikes in System Memory Use 74
Stopping BGP from Overloading System Memory 74
MD5 Authentication for BGP 75
Route Flap Damping 75
Fast Fallover 75
Route Reflection 75
Stripping Private AS Numbers from Update Messages 75
New Community Number Format 76
OSPF Not-So-Stubby Area Option (NSSA) 77
Modified Commands 77
OSPF Auto Cost Calculation 79
Authenticating OSPF 80
Password Authentication 80
Cryptographic Authentication 80
New Commands 81
Importing BGP Routes into OSPF 85
Enabling BGP Route Import 85
Limiting the Number of Routes 85
Advertising Desired Routes 85
Configuration Example 86
DVMRP Interoperability 86
IGMP Snooping All-groups 87
New Commands 87
Modified Command 88

Static IGMP 90
Specifying Router Multicast Addresses for IGMP Snooping 92
Logging and SNMP Traps for PIM-SM 93
PIM4 Hash Mask Length Support 94
Modified Commands 94
Specifying the MLD Query Version 95
ICMP Router Discovery Advertisements 96
Router Discovery Process 96
Router Advertisement Messages 97
Router Solicitation Messages 97
Router Advertisement Interval 97
Preference Level 97
Lifetime 97
Configuration Example 97
Adopting the VRRP IP Address 99
Benefits of VRRP IP Address Adoption 99
Risks of VRRP IP Address Adoption 99
Recommendations 99
Configuration of VR IP Address Adoption 100
Values for IPv6 Router Advertisement *AdvRetransTimer* *101*
Defence Against SYN Flood Attacks 101
Debugging and Displaying Firewall ARP Requests 102
New IPsec Log Messages 104
Inbound packet discarded 104
Outbound packet discarded 104
ISAKMP Counters and Log Message 105
Retransmission received after completion 111
Compare the ID of the IKE Packet 111
Increase Number of IKE Key Exchanges 111
PPTP Pass Through 112
IPsec NAT-Traversal 114
Basic NAT-T Operations 114
NAT-T on the Router or Switch 115
Commands Modified for NAT-T 116
NAT-T Configuration Example 119
Email Relaying 124

Introduction

Allied Telesyn announces the release of Software Version 2.7.3 on the products shown in Table 1. This Release Note describes all features that are new for Software Version 2.7.3 on any product. The features and enhancements that apply to each product series are shown in "Overview of New Features" on page 7.

Table 1: Products supported by Software Version 2.7.3

Product series	Models
AT-9900	AT-9924T, AT-9924SP, AT-9924T/4SP
AT-8900	AT-8948
SwitchBlade	AT-SB4104, AT-SB4108
AT-9800	AT-9812T, AT-9816GB
Rapier i	Rapier 24i, Rapier 48i, Rapier16fi
Rapier	Rapier G6, Rapier G6f, Rapier 16f
AT-8800	AT-8824, AT-8848
AT-8700XL	AT-8724XL, AT-8748XL
AT-8600	AT-8624T/2M
AR700	AR725, AR745
AR400	AR440S, AR441S, AR450S

This Release Note should be read in conjunction with the Installation and Safety Guide or Quick Install Guide, Hardware Reference, and Software Reference for your switch or router. These documents can be found on the Documentation and Tools CD-ROM packaged with your switch or router, or at

www.alliedtelesyn.co.nz/documentation/documentation.html

www.alliedtelesyn.co.uk/en-gb/support/

This Release Note has the following sections:

1. **Upgrading to Software Version 2.7.3**

   This section lists the file names that may be downloaded from the web site.

2. **Description of New Features in Software Version 2.7.3**

   This section describes the features that are new for Software Version 2.7.3 and how to configure the new minor features. Features are listed in approximately the following order:

   a. Brief summaries of the major features

   b. Device and network management

   c. Switching, QoS and Layer 2

   d. Routing

   e. Network and data security

3. **Documentation for Major Features**

   This section has full documentation for major features in Software Version 2.7.3. This is included as chapters or parts of chapters from the Software Reference. This section contains:

   a. MSTP

   b. Classifier

   c. Software QoS

   d. LACP

   e. BGP (containing new features and improved documentation)

   f. SNMPv3

   g. Stacking

   h. Enhancements to Port Authentication

Caution: Information in this document is subject to change without notice and does not represent a commitment on the part of Allied Telesyn International. While every effort has been made to ensure that the information contained within this document and the features and changes described are accurate, Allied Telesyn International can not accept any type of liability for errors in, or omissions arising from, the use of this information.

Upgrading to Software Version 2.7.3

Software Version 2.7.3 is available as a flash release that can be downloaded directly from the Software Updates area of the Allied Telesyn web site at:

www.alliedtelesyn.co.nz/support/updates/patches.html

www.alliedtelesyn.co.uk/en-gb/support/

Software versions must be licenced and require a password to activate. To obtain a licence and password, contact your authorised Allied Telesyn distributor or reseller.

Table 2: File names for Software Version 2.7.3

Product name	Release file	GUI resource file
AT-9924T	89-273.rez	—
AT-9924SP	89-273.rez	—
AT-9924T/4SP	89-273.rez	—
AT-8948	89-273.rez	—
SwitchBlade 4004	sb-273.rez	d_sb4e16.rsc
SwitchBlade 4008	sb-273.rez	d_sb8e16.rsc
AT-9812T	sb-273.rez	d9812e16.rsc
AT-9816GB	sb-273.rez	d9816e16.rsc
Rapier 24i	86s-273.rez	dr24ie16.rsc
Rapier 48i	86s-273.rez	dr48ie16.rsc
Rapier16fi	86s-273.rez	dr16ie16.rsc
Rapier G6	86s-273.rez	d_rg6e16.rsc
Rapier G6f	86s-273.rez	drg6fe16.rsc
Rapier 16f	86s-273.rez	dr16fe16.rsc
AT-8824	86s-273.rez	d8824e16.rsc
AT-8848	86s-273.rez	d8848e16.rsc
AT-8724XL	87-273.rez	d8724e16.rsc
AT-8748XL	87-273.rez	d8748e16.rsc
AT-8624T/2M	sr-273.rez	dsr24e16.rsc
AR725	52-273.rez	d_725e16.rsc
AR745	52-273.rez	d_745e16.rsc
AR440S	54-273.rez	d440se16.rsc
AR441S	54-273.rez	d441se16.rsc
AR450S	54-273.rez	d450se16.rsc

Overview of New Features

This section lists the new features and enhancements by product series. For supported models, see Table 1 on page 4. Note that many of these enhancements were first released in earlier software versions on some models; the table shows all product series that each enhancement applies to.

Table 3: New features and enhancements in Software Version 2.7.3

	AR400	AR700	Rapier	AT-8800	AT-8700XL	AT-8600	AT-9800	SwitchBlade	AT-8900	AT-9900
Multiple Spanning Tree Protocol (MSTP)			✓	✓	✓	✓			✓	✓
Software QoS	✓	✓	✓							
Link Access Control Protocol (LACP)			✓	✓	✓	✓		✓	✓	✓
SNMPv3	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Stacking			✓	✓	✓	✓			✓	✓
Port Authentication Enhancements: MAC-based	✓	✓	✓	✓	✓	✓			✓	✓
Port Authentication Enhancements: Guest VLAN	✓		✓	✓	✓	✓	✓	✓	✓	✓
Port Authentication Enhancements: VLAN assignment	✓		✓	✓	✓	✓	✓	✓	✓	✓
Port Authentication Enhancements: Authentication using RADIUS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Configurable RADIUS timers	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RADIUS RSO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TACACS+ Authentication and Telnet	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Text Message at Login	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Remote Security Officer (RSO) Login: IPv4	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Remote Security Officer (RSO) Login: IPv6	✓	✓	✓	✓			✓	✓	✓	✓
Interrupting Text Flow with the CLI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Support for Long File Names	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TFTP Load Over IPv6	✓	✓	✓	✓			✓	✓	✓	✓
Extended Show Debug Command	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Temperature and Fan Monitoring									✓	✓
Telnet Session Limit and Timeout	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Reverse Telnet	✓	✓	✓	✓	✓	✓				
Adding SNMP Management Stations by IP Address Range	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Syslog Facility Override	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DHCP Probing for IP Addresses	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Classification By L4 Mask and Inner VLAN Settings									✓	✓
QoS Counters									✓	✓

Table 3: New features and enhancements in Software Version 2.7.3 (Continued)

	AR400	AR700	Rapier	AT-8800	AT-8700XL	AT-8600	AT-9800	SwitchBlade	AT-8900	AT-9900
Actions for QoS Traffic Classes and Flow Groups									✓	✓
L2 QoS Actions in Hardware Filters									✓	✓
IP DSCP Override									✓	✓
CPU Transmit User Priority Override									✓	✓
CPU Transmit Queue Priority Override									✓	✓
Changes to QoS on Trunked Ports										✓
Enable/Disable Port Egress Queue and Flow Control									✓	✓
Disabling 10/100 Ports at the Hardware Level			✓	✓	✓	✓	✓		✓	✓
Increase in VLAN Name Length	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Protected VLANs and Private VLANs: Protected VLANs			✓	✓	✓	✓				
Protected VLANs and Private VLANs: Private VLANs			✓	✓					✓	✓
RSTP BPDU Loopback Detection			✓	✓	✓	✓	✓	✓	✓	✓
Virtual Bridge (VLAN) MIB			✓	✓	✓	✓	✓	✓	✓	✓
MAC Address Thrashing Protection for Ports									✓	✓
Recovery Disposal to PPPoE	✓	✓	✓	✓			✓	✓	✓	✓
TPAD Over TCP/IP Improvements	✓[a]									
Ping and Trace Using DNS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Blackhole Routing									✓	✓
Hardware Equal Cost Multipath Routing (ECMP)									✓	✓
Hardware IP Route Learning Delay									✓	✓
Disable Source Routing	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Local Interfaces	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Use of Default Route by Proxy ARP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Support for 5000 Next Hops									✓	✓
BGP: **Soft Resetting of Modified Peers**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Peer Templates**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Extensions to Route Maps**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Prefix Lists**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Tagging Static Routes for Import Filtering**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Handling Spikes in System Memory Use**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Stopping BGP from Overloading System Memory**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **MD5 Authentication for BGP**	✓	✓	✓	✓			✓	✓	✓	✓

Table 3: New features and enhancements in Software Version 2.7.3 (Continued)

	AR400	AR700	Rapier	AT-8800	AT-8700XL	AT-8600	AT-9800	SwitchBlade	AT-8900	AT-9900
BGP: **Route Flap Damping**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Fast Fallover**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Route Reflection**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **Stripping Private AS Numbers from Update Messages**	✓	✓	✓	✓			✓	✓	✓	✓
BGP: **New Community Number Format**	✓	✓	✓	✓			✓	✓	✓	✓
OSPF Not-So-Stubby Area Option (NSSA)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OSPF Auto Cost Calculation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Authenticating OSPF	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Importing BGP Routes into OSPF	✓	✓	✓	✓			✓	✓	✓	✓
DVMRP Interoperability	✓	✓	✓	✓			✓	✓	✓	✓
IGMP Snooping All-groups	✓		✓	✓	✓	✓	✓	✓	✓	✓
Static IGMP			✓	✓	✓	✓	✓	✓	✓	✓
Specifying Router Multicast Addresses for IGMP Snooping	✓		✓	✓	✓	✓	✓	✓	✓	✓
Logging and SNMP Traps for PIM-SM	✓	✓	✓	✓			✓	✓	✓	✓
PIM4 Hash Mask Length Support	✓	✓	✓	✓			✓	✓	✓	✓
Specifying the MLD Query Version	✓	✓	✓	✓			✓	✓	✓	✓
ICMP Router Discovery Advertisements	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Adopting the VRRP IP Address	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Values for IPv6 Router Advertisement AdvRetransTimer	✓	✓	✓	✓			✓	✓	✓	✓
Defence Against SYN Flood Attacks	✓	✓	✓	✓					✓	
Debugging and Displaying Firewall ARP Requests	✓	✓	✓	✓			✓	✓	✓	
New IPsec Log Messages	✓	✓	✓	✓					✓	
ISAKMP Counters and Log Message	✓	✓	✓	✓					✓	
Compare the ID of the IKE Packet	✓	✓	✓	✓					✓	
Increase Number of IKE Key Exchanges	✓	✓	✓	✓					✓	
PPTP Pass Through	✓	✓	✓	✓			✓	✓		
IPsec NAT-Traversal	✓	✓	✓	✓					✓	
Email Relaying	✓	✓	✓	✓					✓	

a. AR440S and AR441S routers

Multiple Spanning Tree Protocol (MSTP)

Multiple Spanning Tree Protocol (MSTP) was developed to address limitations in the existing protocols, Spanning Tree Protocol (STP) and Rapid Spanning Tree Protocol (RSTP). These limitations apply mainly to networks that use multiple VLANS with topologies employing alternative physical links. MSTP is defined in IEEE Standard 802.1Q 2003.

Information about MSTP and its commands is in the MSTP chapter at the end of this Release Note.

Software QoS

Software Quality of Service lets you intelligently manage network traffic to allow stable and predictable end-to-end network performance. In particular, it helps you provide sensitive traffic with the network resources it needs even when the network is congested, especially real-time voice and video traffic. Software QoS operates on Layer 2 interfaces such as PPP, ETH, frame relay and ATM interfaces, and Layer 3 tunnels such as IPv6 to IPv4, IPsec and GRE tunnels.

To configure Software QoS you need to use both Classifier and QoS commands. Information about these features and their commands are in the Generic Packet Classifier and Software Quality of Service (QoS) chapters at the end of this Release Note.

Link Access Control Protocol (LACP)

The Link Access Control Protocol (LACP) follows the IEEE Standard 802.3-2002, CSMA/CD access method and physical layer specifications. It enables trunk groups, called aggregated links, to be created automatically.

LACP operates where systems are connected over multiple communication links. It constantly monitors these links and automatically adds or removes them from trunk groups.

Information about LACP and its commands is in the LACP chapter at the end of this Release Note.

SNMPv3

SNMPv3 provides enhanced security management features whilst maintaining compatibility with earlier versions SNMPv1 and SNMPv2. The basic additional features of version 3 are:

- Message Authentication
- Hashing and time stamping is employed to ensure that messages are received from valid sources.
- Message Confidentiality
- Encryption can be applied to messages to ensure content privacy.

Information about SNMP and its commands is in the SNMP chapter at the end of this Release Note.

Stacking

Stacking is a way to synchronise information across multiple switches and manage them as one logical device. Stacking uses a proprietary protocol to manage a group of separate switches as one.

Information about Stacking and its commands is in the Stacking chapter at the end of this Release Note.

Stacking has been previously released on AT-8600, AT-8700XL, Rapier and AT-8800 switches. Enhancements since this prior release include:

- When you set the time through the CLI, SNMP, or NTP, the stack synchronises to that time. The time is then periodically checked and resynchronised across the stack.
- The **copy** command has been modified so that switches in a stack can copy files to stack members.
- Stacking is disabled by default.

Port Authentication Enhancements

Port authentication now includes:

- MAC-based authentication, as an alternative to the existing 802.1x authentication
- 802.1x authentication guest VLANs
- 802.1x authentication dynamic VLAN assignment
- enhancements to 802.1x authentication methods using RADIUS
- See also "Configurable RADIUS timers" on page 12

Information about these enhancements and the modified commands is in "Port Authentication Enhancements" at the end of this Release Note.

Configurable RADIUS timers

The following timer parameters have been added to the **set radius** command, to improve response times in environments where some servers may be unavailable:

- **timeout** specifies how long the device should wait for a response from the RADIUS server, before assuming the communication has failed. The default is 6 seconds.
- **retransmitcount** is the number of times that the device will attempt to contact the RADIUS server, before it goes on to the next server. The default is 3 attempts.
- **deadtime** is the length of time for which the server should be considered dead. The default is 0 minutes. When a RADIUS server cannot be contacted, it is considered 'dead' for a period of time.

New Command

set radius

Syntax

```
SET RADius [TIMEOut=1..15] [DEAdtime=0..1440]
   [RETransmitcount=1..5]
```

Description This command sets the timeout and retry parameters for all radius servers.

The **timeout** parameter specifies the length of time for which the device will wait for a server to respond to a given request, before the request is deemed to have timed out. The default value is 6 seconds.

The **deadtime** parameter specifies a length of time for which non responsive servers cannot be used for authentication. The default value is 0 minutes.

The **retransmitcount** parameter specifies the number of times the device will attempt to resend a given request to a RADIUS server before the server is considered non-responsive. The default value is 3.

Modified Commands

show radius

Syntax

```
SHow RADius
```

Description This command displays the list of known RADIUS servers, and a list of settable RADIUS parameters. RADIUS servers are used for user authentication.

Figure 1: Example output from the **show radius** command

```
RADIUS Server Parameters
--------------------------------------------------------------------------------
Server Retransmit Count....... 2
Server Timeout................ 7 sec
Server Dead Time.............. 0 min
--------------------------------------------------------------------------------
Server         Port   AccPort  Secret   LocalInterface   Status
--------------------------------------------------------------------------------
192.168.17.11   1645  1646     ******   local4           Alive
172.31.253.9    1645  0        ******   Not set          Alive
172.20.15.20    1337  0        ******   Not set          Dead (3min)
--------------------------------------------------------------------------------
```

Table 4: Parameters in the output of the **show radius** command

Parameter	Meaning
Server Retransmit Count	The number of times the device will attempt to contact a given RADIUS server, before moving on to the next server.
Server Timeout	The length of time the device will wait for a response from a RADIUS server for any given request.
Server Dead Time	Should a dead time be set, non responsive servers will not be used again for authentication, for a time equal to that of the Server Dead Time.
Status	The status of the server, either Alive or Dead. A value of Alive means that the server will be used for authentication. A value of Dead means that the server will not be used, until its dead period has timed out. The value in brackets for a dead server indicates the time in minutes before the dead period expires.

RADIUS RSO

If the router or switch is configured to have login accounts authenticated by a RADIUS server and RSO, the router or switch will now first authenticate with a RADIUS server. The RSO status is then checked, to only allow users to log in on asyn0, via SSH, or Telnet in from an IP address in the RSO address list. The router or switch also supports multiple user privilege levels using RADIUS.

TACACS+ Authentication and Telnet

If your login to the router or switch is authenticated with TACACS+, you can only use outbound telnet if your TACACS+ privilege level is also equal to or higher than the minimum TACACS+ privilege level required for using telnet on the router or switch. By default, no TACACS+ users can telnet from the router or switch. To set a privilege level, use the command:

```
set tacplus telnet={0..15|none}
```

Note that a user can have a TACACS+ privilege level that is equivilent to User or Manager but be unable to use telnet on the router or switch if the TACACS+ privilege level required for using telnet is higher than the privilege level they have been assigned. For example, if the TACACS+ privilege level required for using telnet is set to 10 and there are two users with Manager privileges, one with a TACACS+ privilege level of 9 and one with a privilege level of 10, only the user with a privilege level of 10 can use telnet on the router or switch.

New Commands

set tacplus telnet

Syntax `SET TACPlus TELnet={0..15|None}`

Description This command determines whether or not TACACS+ authenticated users can telnet from the router or switch.

The **telnet** parameter specifies the minimum TACACS+ privilege level required for using telnet on the router or switch. A value of **none** disables telnet for all TACACS+ authenticated users. A value of **1** indicates that all users can telnet. A value of **7** indicates that Manager privilege or better is required. A value of **15** is equivalent to Security Officer privilege. The default is **none**.

Examples To allow telnet for TACACS+ authenticated Security Officers, use the command:

```
set tacp tel=15
```

Related Commands **show tacplus telnet**

show tacplus telnet

Syntax `SHOW TACPLUS TELNET`

Figure 2: Example output from the **show tacplus telnet** command

```
TACACS+ telnet privilege level: NONE
```

Table 5: Parameters in the output of the **show tacplus telnet** command

Parameter	Meaning
TACACS+ telnet privilege level	The level of TACACS+ privilege required for using telnet on the router or switch; a number in the range 0 to 15, or **none**. **None** indicates that no TACACS+ authenticated user can use telnet.

Text Message at Login

Before users get the prompt that lets them log in, contents from a file named *login.txt* is displayed if it exists in flash memory. The `login.txt` file lets various kinds of messages be sent to users. The following diagram is an example of output from the `login.txt` file (in bold text to indicate which portion of the startup display is from `login.txt`).

```
INFO: Self tests beginning.
INFO: RAM test beginning.
PASS: RAM test, 65536k bytes found.
INFO: BBR tests beginning.
.....
.....
INFO: Switch startup complete

Warning: This equipment is for authorised persons only. If you
do not have proper clearance, please logout now.

Login:
```

Users with Manager privileges or higher create the file named *login.txt* by using the **edit** command or by loading an existing text file. The contents of the file must be in printable ASCII characters but with no control characters. When no login.txt file exists, the login prompt is displayed without a message.

For more information to help create a login.txt file, see the **edit** command and the **load** command in the *Software Reference*.

After someone with User privileges successfully logs in, the router or switch activates an auto-executing file, autoexec.scp, if one is in flash memory. Users with Manager privileges or higher also create these script files. For more information about scripts, see the *Scripting* chapter in the *Software Reference*.

Remote Security Officer (RSO) Login

The Remote Security Officer (RSO) feature lets a remote user connect to a router or switch via Telnet from an authorised IP address, and login using a name with Security Officer privilege as if the user were at a terminal connected directly to the router or switch. The RSO feature is configured by defining authorised IP addresses using the **add user rso** and **delete user rso** commands. These commands now accept ranges of IP addresses:

```
add user rso ip=ipadd [mask=ipadd]
add user rso ip=ipadd[-ipadd]
delete user rso ip=ipadd[-ipadd]
```

where *ipadd* is an IP address in dotted decimal notation. If a mask is not specified, the default is 255.255.255.255.

IPv6 addresses are now also supported, enabling Remote Security Officers to login over an IPv6 network:

```
add user rso ip=ipv6add[/prefix-length]
add user rso ip=ipv6add[-ipv6add]
delete user rso ip=ipv6add/prefix-length
delete user rso ip=ipv6add[-ipv6add]
```

where *ipv6add* is an IPv6 address. If a prefix length is not specified, the default is 128.

Interrupting Text Flow with the CLI

A new function has been added to let you interrupt (or "break") text paging or continuously streaming text on the command line interface. The key combination is Ctrl-Q.

This capability will be useful with stand alone commands such as **show** commands that display many output screens. The text is buffered and undisplayed text is deleted. The command prompt is then restored.

The paging prompt will continue giving users the option to display the next line of text output or next page, print text continuously with no further prompts, or abort text output.

This functionality will not work on commands that produce output of indeterminate length, such as **enable** and **disable** commands where output starts with *enable* and stops with *disable*.

Support for Long File Names

File names of up to twenty-eight characters long and extensions of three characters (DOS 28.3 format) are supported since software version 2.6.4, or for products that did not have software version 2.6.4, the next available release.

All software versions support short filenames (DOS 8.3 format). Software version 2.5.1 and later support long file names in either DOS 16.3 or DOS 28.3 format. The table below summarises which software versions support different DOS filename formats.

Table 6: The DOS filename formats supported by different software versions

Software version	Dos 8.3 format	DOS 16.3 format	DOS 28.3 format
2.4.x and earlier	Yes	No	No
2.5.1 and later	Yes	Yes	No
2.6.4 and later	Yes	Yes	Yes

Upgrading to new software versions

When upgrading to this software version from a previous software version, file names retain their DOS naming format. DOS 8.3 format filenames remain in DOS 8.3 format and DOS 16.3 format filenames remain in DOS 16.3 format.

Regressing to Previous Software Versions

If you install a software version that supports DOS 28.3 format, and then you install a previous software version that supports **only** DOS 8.3 format (Table 6), long file names that were in DOS 28.3 format are truncated to DOS 8.3 format. When you reinstall a software version that supports DOS 28.3 format, these truncated file names are restored to their DOS 28.3 format and no information is lost.

If you install a software version that supports DOS 28.3 format, and then you install a previous software version that supports DOS 16.3 format (Table 6), long file names in DOS 28.3 format are permanently truncated to DOS 8.3 format. For example, the file `AB12345678.SCP` is permanently renamed `AB123~01.SCP`. Any long file names that were in DOS 28.3 format remain truncated in DOS 8.3 format when you reinstall a software version that supports DOS 28.3 format.

TFTP Load Over IPv6

It is now possible to upload and download files to and from the router or switch over an IPv6 address.

Modified commands

The **ipv6add** option has been added to the **server** parameter for the **load**, **set loader**, and **upload** commands:

```
LOAd [METhod=TFtp] [DELay=delay] [DESTFile=destfilename]
   [DEStination={BOOTblock|CFlash|FLash|NVs}]
   [{FIle|SRCFile}=filename] [SErver={hostname|ipadd|
   ipv6add}]

SET LOAder [ATTribute={CErt|CRl|CAcert|DEFault}]
   [BASeobject={dist-name|DEFault}] [DElay={delay|
   DEFault}] [DESTFile=destfilename]
   [DEStination={BOOTblock|CFLASH|FLash|NVs}]
   [HTTPproxy={hostname|ipadd|DEFault}] [METhod=TFtp]
   [PASSword=password] [ASYn={port|DEFault}]
   [PROxyport={1..65535|DEFault}] [SRCFile|FIle=filename]
   [SErver={hostname|ipadd|ipv6add|DEFault}]
   [SERVport={1..65535|DEFault}] [USErname=username]

UPLoad [METhod=TFtp] [DESTFile=destfilename]
   [FIle=filename] [SErver={hostname|ipadd|ipv6add}]
```

where:

- *ipv6add* is a valid IPv6 address

Extended Show Debug Command

The command **show debug** displays the output of a list of other **show** commands. A **full** parameter has been added and displays a longer list of commands:

```
show debug [stack|full]
```

The output also depends on the router or switch's security mode and the user's privileges. The possible command list variations are given in Table 13.

The **stack** parameter limits the output to a stack dump, if one is available. The output depends on whether the last fatal condition was a hardware reset or a software reboot. After a software reboot, the output is a stack dump. After a hardware reset, no stack dump information is available and a message to this effect is displayed. If the **stack** parameter is not specified, both a stack dump if available and the output of a list of **show** commands is generated.

Modified commands

show debug

Syntax `SHow DEBug [STAck|FULl]`

Description This command displays a snapshot of the state of the router or switch immediately before the last fatal condition, and is used for debugging.

The output depends on the router or switch's security mode, the user's privileges, whether full is specified or not, and the switch or router model. Table 7 on page 20 lists possible command list variations, when the full parameter is or is not specified, under different combinations of security mode and privilege level. Only the commands from this table that apply on your model will be run.

The **stack** parameter limits the output to a stack dump, if one is available. The output depends on whether the last fatal condition was a hardware reset or a software reboot. After a software reboot, the output is a stack dump. After a hardware reset, no stack dump information is available and a message to this effect is displayed. If the **stack** parameter is not specified, both a stack dump if available and the output of a list of **show** commands is generated.

The **full** parameter extends the list of **show** commands, as described in the second half of Table 7.

Table 7: The list of **show** commands that are executed by the **show debug** command, when the **full** parameter is or is not specified, under different combinations of security mode and privilege level

full parameter specified?	security mode	privilege level	list of commands executed
No	normal	manager	show system show files show install show feature show release show config dynamic show buffer scan show cpu show log show exception show ffile check
No	secure	security officer	
No	secure	manager	show system (without current configuration file) show files show install show release show buffer scan show cpu show log show exception show ffile check
Yes	normal	manager	show system show files show install show feature show release show config dynamic show interface show ip interface show ip arp show ip route full show ip count show switch show switch counter show switch fdb show switch table=mac show switch table=ip show switch hotswap show startup show system sysr show system sysr slave show system sysr counters show flash show switch port=all show switch port=all counter show buffer scan show cpu show log show exception show ffile check
Yes	secure	security officer	

Table 7: The list of **show** commands that are executed by the **show debug** command, when the **full** parameter is or is not specified, under different combinations of security mode and privilege level (Continued)

full parameter specified?	security mode	privilege level	list of commands executed
Yes	secure	manager	show system (without current configuration file) show files show install show release show interface show ip interface show ip arp show ip route full show ip count show switch show switch counter show switch fdb show switch table=mac show switch table=ip show switch hotswap show startup show system sysr show system sysr slave show system sysr counters show flash show switch port=all show switch port=all counter show buffer scan show cpu show log show exception show ffile check

Temperature and Fan Monitoring

New information has been added to the show system command output to indicate temperature, fan and PSU state. In particular, the show system output now shows if temperature and fan monitoring fails.

Figure 3: New section of output of show system command

```
PSU1: (AC)      Fan: Normal  Temp: Normal  Power: Normal
PSU2: (AC)      Fan: Normal  Temp: Normal  Power: Warning

Current temperature : Normal

FAN
-----------------
Main fan     Normal
-----------------
```

Table 8: New parameters displayed in the **show system** command

Parameter	Meaning
PSU1 PSU2	The type of unit in each bay (in parentheses). Facing the rear of the switch, PSU1 is the left bay and PSU2 is the right bay.
Fan	Whether the fan voltage of each PSU or FOM is within the supported range; a warning is displayed if not.
Temp	Whether the temperature of each PSU or FOM is within the supported range; a warning is displayed if not.
Power	Whether the power output of each PSU is within the supported range; a warning is displayed if not.
Current Temperature	Internal temperature status of the board; either "Normal", "Warning" (operating outside the supported range), or "Failed" (monitoring has failed, so the switch cannot report whether the temperature is within the supported range). The switch's system LED also indicates the "Failed" state by flashing three times (flash, flash, flash, pause, flash, flash, flash, pause...).
FAN	Status of each internal fan; either "Normal", "Warning" (operating outside the supported range), or "Failed" (monitoring has failed, so the switch cannot report whether the fan is operating within the supported range). The switch's system LED also indicates the "Failed" state by flashing three times (flash, flash, flash, pause, flash, flash, flash, pause...).

Telnet Session Limit and Timeout

This enhancement enables you to limit the number of concurrent telnet sessions, and to specify an idle timeout for telnet sessions. To configure this, use the **set telnet** command.

set telnet

Syntax

```
SET TELNET [IDLETIMEOUT=0..4294967295] [INSERTNULL={ON|
   OFF}] [LISTENPORT=port] [MAXSESSIONS=0..30]
   [TERMTYPE=termstring]
```

Description

The **idletimeout** parameter specifies a period of time, in seconds, for the telnet server's idle timer. If the specified time period lapses since the last time a telnet session received data from the remote client, the session is terminated; this applies from the moment that the telnet session becomes established, regardless of whether the user has logged in or not. If you specify 0 (zero), the idle timer remains off, and the session must be explicitly terminated. The default is 0.

If you modify the telnet server idle timeout period while there are established telnet sessions, the idle timers for those sessions are reset so that they use the new timeout value. The sessions' timer is reset, so they lose any idle time accumulated prior to the idle time modification.

The **maxsessions** parameter specifies the number of concurrent telnet sessions that are supported by the local router or switch. Once this limit is reached, any subsequent session requests are rejected. The session limit cannot be set below the number of currently established telnet sessions. By default, 30 concurrent telnet sessions are supported.

show telnet

Syntax

```
SHOW TELNET
```

The **show telnet** command now displays the number of currently-established sessions, the maximum number of sessions allowed, and the idle timeout in seconds.

Figure 4: Example output from the **show telnet** command

```
TELNET Module Configuration
-----------------------------------------
Telnet Server .................. Enabled
Telnet Server Listen Port ...... 23
Telnet Terminal Type ........... UNKNOWN
Telnet Insert Null's ........... Off
Telnet Current Sessions ........ 1
Telnet Session Limit ........... 12
Telnet Idle Timeout ............ 180
-----------------------------------------
```

Reverse Telnet

This implementation of reverse Telnet is in accordance with RFC 2217 “Telnet Com Port Control Option”.

What does reverse Telnet do?

Reverse Telnet allows traffic to flow in both directions between the Telnet client and server during a session. In the normal mode of Telnet, a client initiates a session to a port on an access server, and once the session is established, messages are passed only from the server to the client. But with reverse Telnet, once the session is established and the client authenticated, the client can send commands (typically from a PC) to the server. The client can configure the target com port and receive data over that port. Client commands are detailed in RFC 2217.

Reverse Telnet TCP ports

When reverse Telnet is enabled, the device will initialise Telnet listen sessions for TCP ports. There is a TCP port, starting with 2001, for each asyn port on the device except asyn0. (Asyn0 is for management only and can not be used for reverse Telnet.) For example, a device with the four asyn ports asyn1 to asyn4 may have four reverse Telnet listen sessions on TCP ports 2001, 2002, 2003, and 2004. If an asyn port supports the following baud rates, they can be used for this implementation of reverse Telnet: 1200, 2400, 4800, 9600, 28800, 38400, 57600, 152000, and 512000.

Client command limitations

A client connected to the router or switch with reverse Telnet can send it commands. These commands are executed from the client machine (typically a PC), and are **not** executed at the CLI of the Allied Telesyn device. The following client commands deviate from RFC 2217, *Special Com Port Control Commands*. See the RFC for details about client commands.

The only supported options for the *set control* client command are:

- request flow control
- set flow control
- set break
- set DTR state
- set RTS state
- request inbound flow control

The *set linestate mask* and *set modemstate mask* client commands are accepted and processed, but notifications of com port and modem line change are not implemented.

The *purge data* client command is accepted but not actioned.

New Commands

disable rtelnet

Syntax `DISable RTELnet`

Description This command disables the reverse Telnet capability on the device. You can only execute this command if reverse Telnet has previously been enabled with the **enable rtelnet** command.

Example To disable reverse Telnet, use the command:

```
dis rtel
```

Related Commands **enable rtelnet**

enable rtelnet

Syntax `ENAble RTELnet`

Description This command enables the reverse Telnet capability on the device. When this command is executed, the device initialises a Telnet listen session on available TCP ports from and including port 2001.

You must enable Telnet with the **enable telnet server** command before you can enable reverse Telnet.

Example To enable reverse Telnet, use the command

```
ena rtel
```

Related Commands **disable rtelnet**

disable rtelnet debug

Syntax `DISable RTELnet DEBug={ALL|CONFig|ERRORcode|OPTions|TRACE}`

Description This command disables debug mode for the reverse Telnet capability on the device. You can only execute this command if debug has previously been enabled with the **enable rtelnet debug** command.

The **all** parameter disables debug output for all types of debug.

The **config** parameter disables debugging for the setup phase of the reverse Telnet session.

The **errorcode** parameter disables the translation of error codes returned by reverse Telnet into words.

The **options** parameter disables debug output on the negotiation of Telnet options between the device and the client.

The **trace** parameter disables debug output of function names as they are called during code execution.

Example To disable config debug, use the command:

```
dis rtel deb=conf
```

Related Commands enable rtelnet debug

enable rtelnet debug

Syntax `ENAble RTELnet DEBug={ALL|CONFig|ERRORcode|OPTions|TRACE}`

Description This command enables debug mode for the reverse Telnet capability on the device. You can only execute this command if reverse Telnet has previously been enabled.

The **all** parameter enables debug output for all types of debug.

The **config** parameter enables debugging for the setup phase of the reverse Telnet session. The debug output shows a sequence of messages showing the connection being established between the TTYs of the Telnet session and the asyn port.

The **errorcode** parameter enables the translation of error codes returned by reverse Telnet into words. For example, “Listen session open failed”.

The **options** parameter enables debug output on the negotiation of Telnet options between the device and the client. Unknown options are shown with their option code.

The **trace** parameter enables debug output of function names as they are called during code execution. Trace debug output may be long.

Example To enable errorcode debug, use the command:

```
ena rtel deb=error
```

Related Commands enable rtelnet

Modified Command

show telnet

Syntax `SHow TELnet`

Description This command displays information about the current Telnet settings.

Figure 5: New parameter in the output of the **show telnet** command

```
TELNET Module Configuration
-------------------------------------------------------------
  Telnet Server .................. Enabled
  Telnet Server Listen Port ...... 23
  Telnet Terminal Type ........... UNKNOWN
  Telnet Insert Nulls ............ Off
  Telnet Com Port Control ........ Enabled
  Telnet Current Sessions ........ 1
  Telnet Session Limit ........... 12
  Telnet Idle Timeout ............ 180
-------------------------------------------------------------
```

Table 9: New parameters in the output of the **show telnet** command

Parameter	Meaning
Telnet Com Port Control	Whether or not reverse Telnet is enabled; one of "Enabled" or "Disabled".

Adding SNMP Management Stations by IP Address Range

You can now add management stations to a community either individually by entering just the station's IP address, or you can add a range of management stations by entering an IP address that ends with a '/' character followed by a number between 1 and 32. The number that follows the '/' character operates as an address mask that defines a range of management station addresses. Note that this enhancement is effective only on SNMP versions I and II.

The following commands have been modified in order to apply this enhancement:

```
create snmp community
add snmp community
delete snmp community
show snmp community
```

The full syntax of these commands is included in the SNMP chapter at the end of this Release Note.

Syslog Facility Override

This enhancement enables you to override the facility parameter in syslog messages.

Modified Commands

To set the syslog facility, use the highlighted new parameters in the following existing commands. The **show log output** command has also been modified to display the facility setting.

create log output

Syntax

```
CREATE LOG OUTPUT={TEMPORARY|PERMANENT|output-id}
   DESTINATION=SYSLOG [FACILITY={DEFAULT|LOCAL1..LOCAL7}]
   [other-options]
```

Description The **facility** parameter specifies whether to override the mapping between logging facility type and syslog facility identifiers. The **default** option keeps the mapping between type and facility. If you specify a local value (**local1, local2**, ... **local7**) then the syslog facility identifier will always be sent with the value specified. The **facility** parameter is valid only if **destination** is set to **syslog**. The default for **facility** is **default**.

set log output

Syntax

```
SET LOG OUTPUT={TEMPORARY|PERMANENT|output-id}
   [DESTINATION=SYSLOG] [FACILITY={DEFAULT|LOCAL1..LOCAL7}]
   [other-options]
```

Description The **facility** parameter specifies whether to override the mapping between logging facility type and syslog facility identifiers. The **default** option keeps the mapping between type and facility. If you specify a local value (**local1, local2**, ... **local7**) then the syslog facility identifier will always be sent with the value specified. The **facility** parameter is valid only if **destination** is set to **syslog**. The default for **facility** is **default**.

DHCP Probing for IP Addresses

When creating a DHCP range, you can now specify how the DHCP server checks whether an IP address is being used by other hosts by specifying the new **probe** parameter, with the command:

```
create dhcp range=name ip=ipadd number=number policy=name
    [gateway=ipadd] [probe={arp|icmp}]
```

The **probe** parameter specifies how the DHCP server checks whether an IP address is being used by other hosts. If **arp** is specified, the server sends ARP requests to determine if an address is in use. If **icmp** is specified, the server sends ICMP Echo Requests (pings). The default is **icmp**.

To modify the server's method for probing IP addresses, use the new command:

```
set dhcp range=name probe={arp|icmp}
```

Note that **arp** cannot be specified if the range includes a gateway (by specifying the **gateway** parameter when it was created), or if the network uses Proxy ARP.

Note that **arp** cannot be specified if the range includes a gateway (by specifying the **gateway** parameter when it was created), or if the network uses Proxy ARP.

Classification By L4 Mask and Inner VLAN Settings

The following new optional parameters have been added to the **create classifier** and **set classifier** commands:

```
[INNERTPID=tpid|ANY]
[INNERVLANID=VLAN=1..4094|ANY]
[INNERVLANPRIORITY=0..7|ANY]
[L4DMASK=mask|ANY]
[L4SMASK=mask|ANY]
[TPID=tpid|ANY]
[VLANPRIORITY=0..7|ANY]
```

where

- *mask* is a 2-byte hexadecimal number.
- *tpid* is a 2-byte hexadecimal number.

The **l4smask** parameter specifies the mask or range of TCP/UDP source ports in the packet. The default is **any**.

The **l4dmask** parameter specifies the mask or range of TCP/UDP source ports in the packet. The default is **any**.

The **vlanpriority** parameter specifies the 802.1P priority in the VLAN tag. The default is **any**.

The **tpid** parameter specifies the Tag Protocol Identifier field in the packet. The default is **any**.

The **innervlanpriority, innertpid** and **innervlanid** parameters are for use on double tagged traffic. **inner** refers the second 802.1Q tag in the packet. This is the one being tunnelled. Table 10 shows where in the packet the inner and outer tags will be matched. Matching outer parameters on customer ports is not possible.

Table 10: Offsets into packet for Inner and Outer parameters

	Outer VLAN parameters (normal)	Inner VLAN parameters
Customer port	Only VLAN ID.	1st tag
Core port	1st tag	2nd tag
Nested VLANs disabled	1st tag	2nd tag

When attaching the classifier to a hardware filter and nested VLANs are not being used, it is assumed that all incoming packets are double tagged. When nested VLANs are being used, outer parameters also cannot be used to match on customer ports. If the classifier is being attached to a number of ports, they will all be treated like core ports if at least one port is a core port.

The **innervlanpriority** parameter specifies the second 802.1P field in the packet. The default is **any**.

The **innertpid** parameter specifies the TPID in the second 802.1Q tag in the packet. The default is **any**.

The **innervlanid** parameter specifies the tunnelled VLAN ID in the second 802.1Q tag in the packet. The default is **any**.

QoS Counters

A new command, **set switch enhancedmode**, has been added to enable you to monitor QoS counters. You can then reset the counters using new **reset** commands, and display the counters using new and expanded **show** commands, as listed below.

set switch enhancedmode

Syntax `SET SWITCH ENHANCEDMODE={QOSCOUNTERS|NONE}`

Description This command sets the enhanced mode of the switch to include monitoring of QoS counters for egress queues and traffic classes. The default is **none**.

Examples To turn monitoring of QoS counters on use the command:

```
set switch enhancedmode=qoscounters
```

To turn monitoring of QoS counters off use the command:

```
set switch enhancedmode=none
```

reset qos port

Syntax

```
RESET QOS PORT={port-list} COUNTERS
   TRAFFICCLASS[={trafficclass-list|DEFAULT|ALL}]
```

where:

- *port-list* is a single port of group of ports; a range of integers (specified as 0-4) or a comma separated list of policy numbers and/or ranges (0, 3, 4-9).
- *trafficclass-list* is an integer in the range of 0-1023; a range of integers (specified as 0-3) or a comma separated list of integers and/or ranges without spaces.

Description This command resets the counters for the specified traffic classes attached to the specified port(s).

Examples To clear the traffic class counters on the traffic classes attached to port 4, use the command:

```
reset qos port=4 counters trafficclass=all
```

reset qos accelerator

Syntax

```
RESET QOS ACCELERATOR COUNTERS
   TRAFFICCLASS[={trafficclass-list|DEFAULT|ALL}]
```

where:

- *trafficclass-list* is an integer in the range of 0-1023; a range of integers (specified as 0-3) or a comma separated list of integers and/or ranges without spaces.

Description This command resets the counters for the traffic classes attached to the accelerator card.

Examples To clear the traffic class counters on the traffic classes attached to port 4, use the command:

```
reset qos accelerator counters trafficclass=1-5
```

show qos port counters

Syntax

```
SHOW QOS PORT[={port-list|ALL}] COUNTERS
   TRAFFICCLASS[={trafficclass-list|DEFAULT|ALL}

SHOW QOS PORT[={port-list|ALL}] COUNTERS
   EGRESSQUEUE[=queue-list]
```

where:

- *port-list* is a single port of group of ports; a range of integers (specified as 0-4) or a comma separated list of policy numbers and/or ranges (0, 3, 4-9).
- *queue-list* is either an integer in the range 0 to 7; a range of integers (specified as 0-3) or a comma separated list of integers and/or ranges, without spaces.
- *trafficclass-list* is a integer in the range of 0-1023; a range of integers (specified as 0-3) or a comma separated list of integers and/or ranges without spaces.

Description This command displays QoS configuration information for one or more ports. If no value is given for the **port** parameter, or if **all** is specified, information about all ports is displayed. If a value is given for the **port** parameter, information about the specified port or ports is displayed.

If **counters trafficclass** is specified the counters for the traffic classes belonging to the policy that is attached to the port will be displayed. If no value or **all** is specified, then information about all traffic classes is displayed. If **default** is specified then only information about the default traffic class will be displayed. If a list of traffic classes is specified then only information about those traffic classes is displayed.

If **counters egressqueue** is specified the counters for the specified ports queues will be displayed. If no value is specified, then information about all egress queues is displayed. If a list of queues are specified, then only information about those egress queues is displayed.

Figure 6: Example output from the **show qos port counters egressqueue** command

```
Port 1 Egress Queue Counters:
  Port queue length .........      48 (maximum 128)
  Egress queue length:
    Queue 0 .................      0 (maximum 16)
    Queue 1 .................      16 (maximum 16)
    Queue 2 .................      16 (maximum 16)
    Queue 3 .................      16 (maximum 16)
    Queue 4 .................      0 (maximum 16)
    Queue 5 .................      0 (maximum 16)
    Queue 6 .................      0 (maximum 16)
    Queue 7 .................      0 (maximum 16)
```

Figure 7: Example output from the **show qos port counters trafficclass** command

```
QOS Counter Information
Port 1:
  Policy: 0
  Traffic Class 0:
    Aggregate Bytes ..............
    BwConformanceClass1 bytes ....                    15220
    BwConformanceClass2 bytes ....                    0
    BwConformanceClass3 bytes ....                    0
    Dropped bytes ................                    15220
  Default Traffic Class:
    Aggregate Bytes ..............                    680
    BwConformanceClass1 bytes ....                    680
    BwConformanceClass2 bytes ....                    0
    BwConformanceClass3 bytes ....                    0
    Dropped bytes ................                    0
```

show qos accelerator

Syntax

```
SHOW QOS ACCELERATOR COUNTERS
   TRAFFICCLASS[={trafficclass-list|DEFAULT|ALL}]

SHOW QOS ACCELERATOR COUNTERS EGRESSQUEUE[=queue-list]
```

where:

- *queue-list* is either an integer in the range 0 to 7; a range of integers (specified as 0-3) or a comma separated list of integers and/or ranges, without spaces.
- *trafficclass-list* is a integer in the range of 0-1023; a range of integers (specified as 0-3) or a comma separated list of integers and/or ranges without spaces.

Description This command displays QoS configuration information for the accelerator card counters.

If **counters trafficclass** is specified the counters for the traffic classes belonging to the policy attached to the accelerator card will be displayed. If no value or **all** is specified, then information about all traffic classes is displayed. If **default** is specified then only information about the default traffic class will be displayed.

If a list of traffic classes is specified then only information about those traffic classes is displayed.

If **counters egressqueue** is specified the counters for the specified queues will be displayed. If no value is specified, then information about all egress queues is displayed. If a list of queues are specified, then only information about those egress queues is displayed.

Figure 8: Example output from the **show qos accelerator counters egressqueue** command

```
Accel Card Egress Queue Counters:
  Total queue length ........      0 (maximum 896)
  Egress queue length:
    Queue 0 ..................      0 (maximum 896)
    Queue 1 ..................      0 (maximum 896)
    Queue 2 ..................      0 (maximum 896)
    Queue 3 ..................      0 (maximum 896)
    Queue 4 ..................      0 (maximum 896)
    Queue 5 ..................      0 (maximum 896)
    Queue 6 ..................      0 (maximum 896)
    Queue 7 ..................      0 (maximum 896)
```

Figure 9: Example output from the **show qos accelerator counters trafficclass** command

```
QOS Counter Information
Accelerator Interface:
  Policy: 1
  There are no traffic classes to display.
  Default Traffic Class:
    Aggregate Bytes ..............                    0
    BwConformanceClass1 bytes ....                    0
    BwConformanceClass2 bytes ....                    0
    BwConformanceClass3 bytes ....                    0
    Dropped bytes ................                    0
```

Actions for QoS Traffic Classes and Flow Groups

This enhancement enables traffic that is classified into a flow group or traffic class to be:

- discarded

  This option means unwanted traffic can be dropped without being processed.

- sent directly to a port on a VLAN
- sent to the mirror port

  This option allows you to mirror packets classified on any port. It is much more flexible than standard mirroring, which only monitors traffic on a single port. You can choose to also forward the mirrored traffic normally, discard it, or send it to a particular port and VLAN.

- forwarded normally.

To act on classified traffic at the flowgroup level, use one of the commands:

```
create qos flowgroup=id-list
   action={none|forward|forward,sendmirror|discard|
   discard,sendmirror|sendmirror|sendmirror,sendvlanport|
   sendvlanport} [vlan=vlan-id] [port=port] [other-options]

set qos flowgroup=id-list
   action={none|forward|forward,sendmirror|discard|
   discard,sendmirror|sendmirror|sendmirror,sendvlanport|
   sendvlanport} [vlan=vlan-id] [port=port] [other-options]
```

where:

- *vlan-id* is an integer in the range 1 to 4094.
- *port* is any valid port number.

The **action** parameter specifies the action that is to be performed on any traffic belonging to the flow group. The default action is **forward**.

If you specify an **action** of **none**, the action will be overridden with the setting of the flow group's traffic class.

If you specify an **action** of **forward**, the traffic will be forwarded normally.

If you specify an **action** of **discard**, the traffic will be dropped.

If you specify an **action** of **sendvlanport**, both **vlan** and **port** must also be specified. Traffic will be sent to the VLAN specified by the **vlan** parameter and the port specified by the **port** parameter. The VLAN must exist and the specified port must be a member of that VLAN. The switch determines whether the port is tagged or untagged for that VLAN, and sends the traffic with the correct tag if the port is tagged. If the port is untagged for the specified VLAN the frame is sent untagged.

If you specify an **action** of **sendmirror**, the traffic will be sent to the pre-configured mirror port.

To act on classified traffic at the trafficclass level, use one of the commands:

```
create qos trafficclass=id-list
   action={forward|forward,sendmirror|discard|
   discard,sendmirror|sendmirror|sendmirror,sendvlanport|
   sendvlanport} [vlan=vlan-id] [port=port] [other-options]

set qos trafficclass=id-list
   action={forward|forward,sendmirror|discard|
   discard,sendmirror|sendmirror|sendmirror,sendvlanport|
   sendvlanport} [vlan=vlan-id] [port=port] [other-options]
```

where:

- *vlan-id* is an integer in the range 1 to 4094.
- *port* is any valid port number.

The actions have the same effect as the flow group actions, except that **action=none** is not valid.

To act on unclassified traffic, which will be processed by the default traffic class, use one of the commands:

```
create qos policy=id-list
   dtcaction={forward|forward,sendmirror|discard|
   discard,sendmirror|sendmirror|sendmirror,sendvlanport|
   sendvlanport} [vlan=vlan] [port=port] [other-options]

set qos policy=id-list
   dtcaction={forward|forward,sendmirror|discard|
   discard,sendmirror|sendmirror|sendmirror,sendvlanport|
   sendvlanport} [vlan=vlan] [port=port] [other-options]
```

where:

- *vlan-id* is an integer in the range 1 to 4094.
- *port* is any valid port number.

The DTC actions have the same effect as the flow group actions, except that **dtcaction=none** is not valid.

L2 QoS Actions in Hardware Filters

This enhancement enables you to classify traffic and use a hardware filter to set its queue, 802.1p user priority or bandwidth class. This is a mechanism for elevating the probability of CPU reception for packets that you determine to be "important".

In heavily congested networks, data streams can sometimes use up all the available bandwidth of the CPU receive process. This increases the probability that infrequently-sent packets are lost, for example routing protocol packets (BGP, OSPF, PIM, DVMRP) or STP packets. By creating an appropriate classifier and hardware filter, such packets can be given preferential treatment, in terms of CPU reception.

Caution: Using this option modifies the processing of packets in congested network situations. We recommend you use this option to increase the priority of control packets, not to reduce the priority of other packets. If you use this option to reduce the priority of control packets, you increase the probability that the switch will drop them at times of congestion.

To configure the hardware filter, use the command:

```
ADD SWITCH HWFILTER=filter-id CLASSIFIER=rule-id
   ACTION=SETL2QOS [L2QOSQUEUE=0..7] [PRIORITY=0..7]
   [BANDWIDTHCLASS=1..3]
```

The **action** parameter specifies what action the classifier entry will take with packets that match the classifier. If you specify **setl2qos**, then packets matching the classifier will have their bandwidth class (or drop precedence), queue, and 802.1p user priority modified to the values specified by the **bandwidthclass, l2qosqueue**, and **priority** parameters. The action **setl2qos** cannot be specified at the same time as **drop**.

The **l2qosqueue** parameter specifies the queue to send packets to that match the classifier. The default is 0.

The **priority** parameter specifies the 802.1p user priority to remark packets with that match the classifier. The default is 0.

The **bandwidthclass** parameter specifies the bandwidth class (drop precedence) to assign packets to that match the classifier. The default is 1.

IP DSCP Override

This enhancement enables you to give a particular DSCP to packets that come from the switch's CPU, using the following command.

Syntax `SET IP DSCPOVERRIDE={NONE|0..63}`

Description This command sets the value that will be written into the DSCP field in the IP header of packets that are sent from the CPU to the switching chip for forwarding.

The **dscpoverride** parameter specifies the DSCP that is written into the DSCP field. The default value of the parameter is **none**, whereby the DSCP field in the packets will not be altered from whatever value the originating software module might have written into it.

Examples To set the DSCP in the IP header to 56, use the command:

```
set ip dscpoverride=56
```

To stop overriding IP DSCP, use the command:

```
set ip dscpoverride=none
```

CPU Transmit User Priority Override

This enhancement enables you to give a particular 802.1p priority to packets that come from the switch's CPU, using the following command.

Caution: Using this option modifies the processing of packets in congested network situations. We recommend you use this option to increase the priority of control packets, not to reduce the priority of other packets. If you use this option to reduce the priority of control packets, you increase the probability that the switch will drop them at times of congestion.

Syntax `SET SWITCH CPUTXPRIORITY={NONE|0..7}`

Description This command specifies the 802.1p value that will be inserted into packets that are sent from the CPU to the switching chip to be transmitted.

The **cputxpriority** parameter specifies the value that will be put into the 802.1p field of tagged packets being sent from the CPU. The default value of the parameter is **none**, where the CPU does not set the 802.1p field in packet headers. **none** can be specified to stop the overriding.

Examples To set the switch to place the value 7 in the 802.1p field of packets sent from the CPU, use the command:

```
set switch cputxpriority=7
```

To disable overriding CPU transmit priority, use the command:

```
set switch cputxpriority=none
```

CPU Transmit Queue Priority Override

This enhancement enables you to send packets that come from the switch's CPU to a specific queue, using the following command.

Caution: Using this option modifies the processing of packets in congested network situations. We recommend you use this option to increase the priority of control packets, not to reduce the priority of other packets. If you use this option to reduce the priority of control packets, you increase the probability that the switch will drop them at times of congestion.

Syntax `SET SWITCH CPUTXQUEUE={NONE|0..7}`

Description This command specifies the transmit queue that will be used for packets that are sent from the CPU to the switching chip for transmission.

The **cputxqueue** parameter specifies the egress queue into which packets from the CPU will be placed by the switching chip. The default value of the parameter is **none**, which means that packets from the CPU will be put into egress queue 0. **none** can be specified to stop the overriding.

Examples To set the switch to put CPU-initiated packets into egress queue 7, use the command:

```
set switch cputxqueue=7
```

To stop specifying an egress queue for CPU-initiated packets, use the command:

```
set switch cputxqueue=none
```

Changes to QoS on Trunked Ports

On the AT-9924T/4SP, AT-9924T and AT-9924SP, you can now only apply a QoS policy to trunked ports if all members of the trunk group are in the same port group. Ports 1-12 form one port group and ports 13-24 form another group.

Enable/Disable Port Egress Queue and Flow Control

This enhancement enables you to enable and disable egress queues and flow control on switch ports, using the following commands.

disable switch port

Syntax

```
DISABLE SWITCH PORT={port-list|ALL} [AUTOMDI]
   [EGRESSQUEUE=queue-list] [FLOW=PAUSE|JAMMING]
```

where:

- *port-list* is a port number, a range of port numbers (specified as n-m), or a comma separated list of port numbers and/or ranges. Port numbers start at 1 and end at m, where m is the highest numbered switch Ethernet port, including uplink ports.
- *queue-list* is an egress queue number, a range of queue numbers (specified as n-m), or a comma separated list of queue numbers and/or ranges. Egress queue numbers start at 0 and end at 7.

Description

This command disables a port or group of ports on the switch or disables auto MDI/MDI-X mode on the ports or disables flow control or disables egress queues. If the port is disabled, it will no longer send or receive packets. If egress queues are disabled packets scheduled for those queues will no longer be transmitted. Disabling a switch does not disable STP operation on the port. Ports should be disabled if there faulty wiring or equipment attached to the ports, or as a security measure to stop access from intruders. Switch ports are enabled by default.

The **port** parameter specifies the port or ports to be disabled or to have auto MDI/MDI-X mode disabled.

The **flow** parameter specifies the type of flow control to be disabled for the port. If **pause** is specified, flow control for full duplex ports by sending PAUSE frames will be disabled. If **jamming** is disabled, flow control for half duplex by sending a jamming signal will be disabled. No flow control is enabled by default.

The **automdi** parameter disables auto MDI/MDI-X mode, and sets the polarity to the default of MDI-X. Auto MDI/MDI-X mode is enabled by default.

The **egressqueue** parameter specifies the egress queue(s) that are to be disabled. If you specify this parameter, you must supply a value for it. At switch startup all egress queues are enabled by default.

Examples

To disable auto MDI/MDI-X mode on ports 2 and 4 use this command:

```
disable switch port=2,4 automdi
```

To disable egress queues 0 and 3 to 5, on port 1, use the command:

```
disable switch port=1 egressqueue=0,3-5
```

To disable pause frames on port 1 use the command:

```
disable switch port=1 flow=pause
```

enable switch port

Syntax

```
ENABLE SWITCH PORT={port-list|ALL} [AUTOMDI]
   [EGRESSQUEUE=queue-list] [FLOW=PAUSE|JAMMING]
```

where:

- *port-list* is a port number, a range of port numbers (specified as n-m), or a comma separated list of port numbers and/or ranges. Port numbers start at 1 and end at m, where m is the highest numbered switch Ethernet port, including uplink ports.
- *queue-list* is an egress queue number, a range of queue numbers (specified as n-m), or a comma separated list of queue numbers and/or ranges. Egress queue numbers start at 0 and end at 7.

Description

This command enables a port or group of ports on the switch or enables auto MDI/MDI-X mode on the ports or enables flow control or enables egress queues. If the port is enabled, it will start to send or receive packets. Enabling a switch does not enable STP operation on the port. Switch ports are enabled by default.

The **port** parameter specifies the port or ports to be enabled or to have auto MDI/MDI-X mode disabled.

The **flow** parameter specifies the type of flow control to be enabled for the port. If **pause** is specified, flow control for full duplex ports by sending PAUSE frames will be enabled. If **jamming** is enabled, flow control for half duplex by sending a jamming signal will be enabled. No flow control is enabled by default.

The **automdi** parameter enables auto MDI/MDI-X mode, and sets the polarity to the default of MDI-X. Auto MDI/MDI-X mode is enabled by default.

The **egressqueue** parameter specifies the egress queue(s) that are to be enabled. If you specify this parameter, you must supply a value for it. At switch startup all egress queues are enabled by default.

Examples

To enable auto MDI/MDI-X mode on ports 2 and 4 use this command:

```
enable switch port=2,4 automdi
```

To enable egress queues 0 and 3 to 5, on port 1, use the command:

```
enable switch port=1 egressqueue=0,3-5
```

To enable pause frames on port 1 use the command:

```
enable switch port=1 flow=pause
```

Disabling 10/100 Ports at the Hardware Level

When disabling a port or group of ports on the switch, you can now specify that 10/100 Ethernet ports are disabled at the hardware level, using the command:

```
disable switch port={port-list|all} [link={enable|disable}]
   [other-options...]
```

The **link** parameter specifies whether 10/100 Ethernet ports are either enabled or disabled at the hardware level. If **disable** is specified, this is the equivalent of disconnecting the cable. If the **link** parameter is not specified, the link remains physically enabled. The default is **enable**.

If a port has been disabled at the hardware level, when it is reset it is enabled at the hardware level and autonegotiation of speed and duplex mode is activated.

Increase in VLAN Name Length

The VLAN parameter now enables you to have a VLAN name of up to 32 characters long. For example, in the command:

```
CREATE VLAN=vlan-name VID=2..4094
```

vlan-name is a unique name from 1 to 32 characters. Valid characters are uppercase and lowercase letters, digits (0-9), the underscore character (“_”), and the hyphen (-). The *vlan-name* cannot be a number or **all** or **default**.

Protected VLANs and Private VLANs

This enhancement extends the available mechanisms for blocking Layer 2 traffic between ports that are members of a VLAN. You can now use:

- Protected VLANs—available on Rapier, Rapier i, AT-8800, AT-8700XL and AT-8600 switches
- Private VLANs—available on Rapier i, AT-8800, AT-8900 and AT-9900 switches

Protected VLANs

If a VLAN is protected, Layer 2 traffic between ports that are members of a protected VLAN is blocked. Traffic can be Layer 3 switched to another VLAN. This feature prevents members of a protected VLAN from communicating with each other yet still allows members to access another network. Layer 3 Routing between ports in a protected VLAN can be prevented by adding a Layer 3 filter. The protected VLAN feature also allows all of the members of the protected VLAN to be in the same subnet.

A typical application is a hotel installation where each room has a port that can be used to access the Internet. In this situation it is undesirable to allow communication between rooms.

To create a protected VLAN, use the command:

```
CREate VLAN=vlan-name VID=2..4094 PRotected
```

Private VLANs on Rapier i and AT-8800 Switches

A private VLAN contains switch ports that are isolated from other ports in the VLAN, but can access another network through an uplink port or uplink trunk group. These ports are called *private ports.* Private ports may be standalone or be combined into groups. Standalone private ports can only communicate with the uplink port, not with other ports in the VLAN. Private ports that are in a group can communicate with other ports in the group and with the uplink port, but cannot communicate with the other private ports in the VLAN.

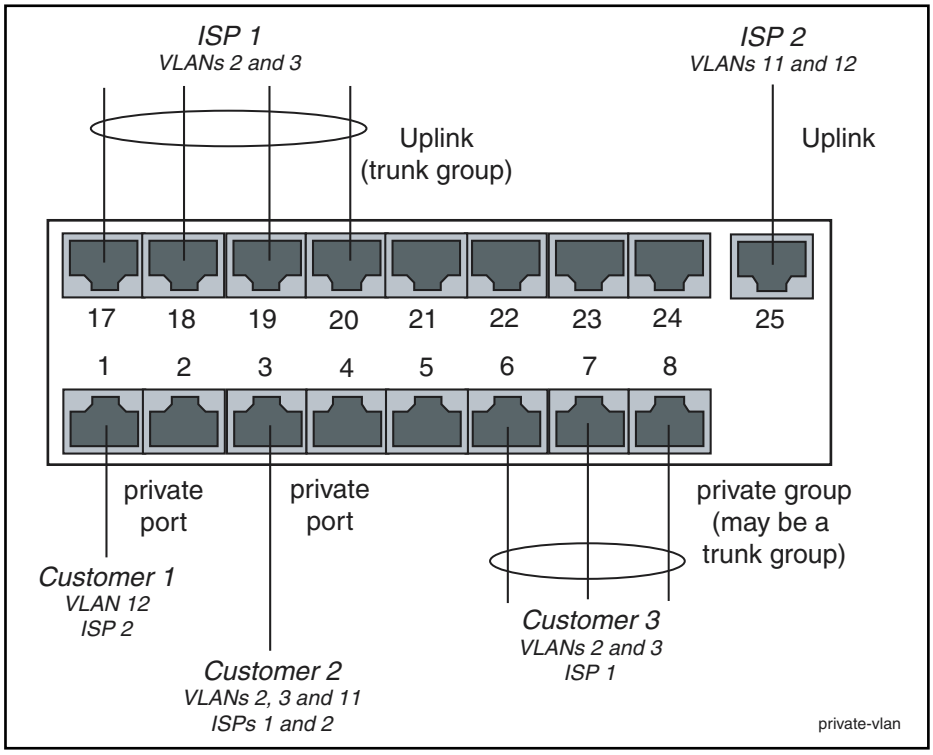
The switch forwards traffic between private ports and the uplink port, and between private ports within a group, according to its normal forwarding rules. The only difference is that forwarding to other private ports is blocked unless the ports are in the same group. Note that all traffic between private ports is blocked, not only Layer 2 traffic.

A typical application is a hotel installation where each room has a port that can access the Internet. In this situation it is undesirable to allow communication between rooms. Another application is to simplify IP address assignment. Ports can be isolated from each other while belonging to the same subnet.

Figure 10 on page 45 shows an example of a network using private VLANs. In this scenario, two service providers are each providing multiple services through multiple VLANs over separate uplinks. Customers are subscribed to services from one or both service providers. Each customer's ports are isolated from other customers, but communicate with the ISP or ISPs through the

appropriate uplink port. A single customer may use multiple ports, connected to individual PCs or trunked together to increase bandwidth. If a customer uses multiple ports, these ports are able to communicate with each other.

Figure 10: Example network configuration using private VLANs



Membership Rules for Private VLANs on Rapier i and AT-8800 Switches

Each private VLAN:

- Must contain one uplink port or uplink trunk group
- May contain multiple private ports
- Cannot contain any non-private ports
- Cannot be the Default VLAN (vlan1)

Each private port:

- Can be a member of multiple private VLANs
- Cannot be a private port in some VLANs and a non-private port in other VLANs
- Cannot be an uplink port in another VLAN

Each uplink port:

- Can be a member of multiple private VLANs
- Cannot be a member of both private and non-private VLANs

Each private or uplink port:

- May be tagged or untagged but can only be an untagged member of one port-based VLAN
- May be trunked with other ports of the same type

Private VLANs on Rapier 48i Switches

The ports on Rapier 48i switches are divided into two instances:

- ports 1-24 plus uplink port 49
- ports 25-48 plus uplink port 50

Private VLANs on a Rapier 48i switch can consist of only ports from one instance. Both the private ports and the uplink port must be in the same instance.

Configuring Private VLANs on Rapier i and AT-8800 Switches

To create a private VLAN and add ports to it:

1. Create the VLAN

To create a VLAN and specify that it is private, use the command:

```
create vlan=vlan-name vid=2..4094 private
```

2. Add the uplink port or trunk group

To add the uplink to a private VLAN, use one of the commands:

```
add vlan={vlan-name|1..4094} port=port-list
    [frame={tagged|untagged}] uplink
```

where *port-list* is either a single port number for a single uplink port, or a list of port numbers for a trunk group. If you are adding a trunk group to the VLAN as an uplink, the ports must already be trunked together, and you must specify all the ports.

3. Add the private ports

To add a private port or ports to a private VLAN, use one of the commands:

```
add vlan={vlan-name|1..4094} port={port-list|all}
    [frame={untagged|tagged}] [group]
```

The **group** parameter specifies that the listed ports may communicate with each other, but not with any other private ports in the VLAN.

4. Delete ports from a private VLAN as necessary

To delete private ports from a private VLAN, use one of the commands:

```
delete vlan={vlan-name|1..4094} port=port-list
```

```
delete vlan={vlan-name|1..4094} port=all
```

A private VLAN cannot contain private ports when an uplink is deleted from the VLAN, because a private VLAN must always have an uplink. To delete the uplink port or ports and any private ports from a private VLAN, use the **port=all** option in the above command.

If the port is a member of a private group, you must delete all ports in the group at once. This stops groups from having different member ports in different VLANs.

Private VLANs on AT-8900 and AT-9900 Switches

A private VLAN contains switch ports that cannot communicate with each other but can access another network. These ports are called *private ports*. Each private VLAN contains one or more private ports, and a single uplink port or uplink trunk group.

A typical application is a hotel installation where each room has a port that can access the Internet. In this situation it is undesirable to allow communication between rooms. Another application is to simplify IP address assignment. Ports can be isolated from each other while belonging to the same subnet.

The switch forwards all traffic that is received on a private port out that VLAN's uplink port, regardless of VLAN ID or MAC destination address. Packets received on an uplink port are forwarded in the normal way (i.e. as for non-private VLANs) for all types of packets. See Table 11 on page 47 for a summary. Note that all traffic between private ports is blocked at all layer, not only Layer 2 traffic.

It is not possible to use protocols such as Telnet or SNMP to manage the switch via private ports. To manage the switch use either non-private ports or uplink ports.

The switch cannot act as a DHCP server to hosts connected to private ports. However, a DHCP server running on a device connected to a private port's uplink port can act in this role.

Table 11: Action taken on packets received on a private port or uplink port

Packet type	Received on private port	Received on uplink port
Unicast	Forwarded out uplink port	Destination port determined on basis of MAC address (as for non-private VLANs)
Multicast	Forwarded out uplink port	Forwarded out all private ports in the VLAN that are members of the multicast group for that VLAN
Broadcast	Forwarded out uplink port	Forwarded out all private ports in the VLAN

Membership rules for private VLANs on AT-8900 and AT-9900 Switches

Each private VLAN:

- Must contain one uplink port or uplink trunk group
- May contain multiple private ports
- Cannot contain any non-private ports
- Cannot be the Default VLAN (vlan1)

Each private port:

- Can be a member of multiple private VLANs, but all these VLANs must have the same uplink port or uplink trunk group
- Cannot be a private port in some VLANs and a non-private port in other VLANs
- Cannot be an uplink port in another VLAN

Each uplink port:

- Can be a member of multiple private VLANs
- Cannot be a member of both private and non-private VLANs

Each private or uplink port:

- May use any VLAN classification rule (port, subnet or protocol)
- May be tagged or untagged, but can only be an untagged member of one port-based VLAN
- May be trunked

Configuring Private VLANs on AT-8900 and AT-9900 Switches

To create a private VLAN and add ports to it

1. Create the VLAN

To create a VLAN and specify that it is private, use the command:

```
create vlan=vlan-name vid=2..4094 private
```

2. Add the uplink port or trunk group

To add the uplink ports to a private VLAN, use one of the commands:

```
add vlan={vlan-name|1..4094} port=port-list
    [frame={untagged|tagged}] uplink

add vlan={vlan-name|1..4094} port=port-list subnet={ipadd|
    all} uplink

add vlan={vlan-name|1..4094} port=port-list
    protocol={protocol-type|index-list|all} uplink
```

where *portlist* is either a single port number for a single uplink port, or a list of port numbers for a trunk group. If you are adding a trunk group to the VLAN as an uplink, the ports must already be trunked together, and you must specify all the ports.

3. Add the private ports

Private ports can be added only after the uplink port or ports have been added to the private VLAN.

To add private ports to a private VLAN, use one of the commands:

```
add vlan={vlan-name|1..4094} port={port-list|all}
    [frame={untagged|tagged}]

add vlan={vlan-name|1..4094} port={port-list|all}
    subnet={ipadd|all}

add vlan={vlan-name|1..4094} port={port-list|all}
    protocol={protocol-type|index-list|all}
```

To delete ports from a private VLAN

To delete private ports from a private VLAN, use one of the commands:

```
delete vlan={vlan-name|1..4094} port={port-list|all}

delete vlan={vlan-name|1..4094} port={port-list|all}
    subnet={ipadd|all}

delete vlan={vlan-name|1..4094} port={port-list|all}
    protocol={protocol-type|index-list|all}
```

A private VLAN cannot contain any private ports when an uplink port is deleted from the VLAN because a private VLAN must always contain one uplink port.

To delete the uplink port from a private VLAN, first delete any private ports from the VLAN, then use one of the previous commands.

Private VLANs and port trunking on AT-8900 and AT-9900 Switches

All ports in a trunk group must have the same VLAN configuration.

Follow these rules for private VLANs and port trunking:

- The port list cannot include any uplink ports when creating a trunk group
- The ports cannot be a mix of private ports and non-private ports when creating a trunk group
- Multiple uplink ports can be added to a private VLAN as a trunk group

To add an uplink trunk group to a private VLAN:

1. Create a trunk group.
2. Add all of the trunk group's ports to the VLAN as uplink ports.

You cannot delete a port from a trunk group if the trunk group is the uplink ports of a private VLAN. To delete a port from such a trunk group:

1. Remove the private ports from the VLAN.
2. Remove the uplink trunk group from the VLAN.
3. Remove the port or ports from the trunk group.
4. Add the uplink trunk group to the VLAN.
5. Add the private ports to the VLAN.

RSTP BPDU Loopback Detection

The output of the **show stp port** command has been changed to display when RSTP detects a downstream loop and puts the port into the "Backup (Loopback Disabled)" state. It also shows the number of times it transitions to that state.

Figure 11: Example output from the **show stp port** command

```
Port .................. 4
     RSTP Port Role ...... Disabled
     State ............... Discarding
     Point To Point ...... No (Auto)
     Port Priority ....... 128
     Port Identifier ..... 8004
     Pathcost ............ 200000
     Designated Root ..... 32768 : 00-00-cd-05-19-28
     Designated Cost ..... 0
     Designated Bridge ... 32768 : 00-00-cd-05-19-28
     Designated Port ..... 8004
     EdgePort ............ No
     VLAN membership ..... 1
     Counters:
       Loopback Disabled            0
```

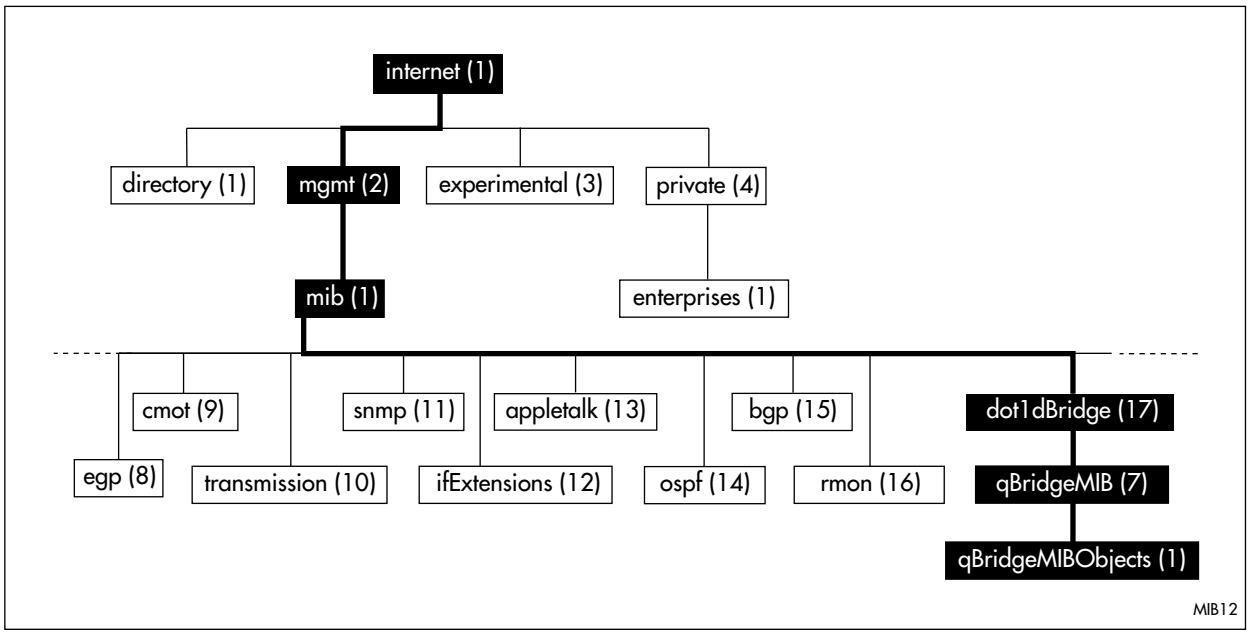
The "RSTP Port Role" entry can now have a status of "Backup (Loopback Disabled)", which indicates the port has transmitted and received the same RSTP BPDU. This state has the same behaviour as the "Backup" state; it drops all packets except BPDUs.

The "Loopback Disabled" counter indicates the number of transitions to the "Backup (Loopback Disabled)" state for the port.

Virtual Bridge (VLAN) MIB

Support has been added for RFC 2674, "Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering and Virtual LAN Extensions" which defines MIB objects for managing IEEE 802.1Q VLANs. Objects defined in this MIB reside in the mib(1) sub-tree (Table 12 on page 51), under the *dot1dBridge* sub-tree defined in RFC 1493, and have the object identifier *qBridgeMIBObjects* ({ mib-2 dot1dBridge(17) qBridgeMIB(7) 1 }).

Figure 12: The Virtual Bridge (VLAN) sub-tree of the Internet-standard Management Information Base (MIB)



The MIB is organised into four logical groups:

- The *dot1qBase* Group contains general objects that apply to any device that supports IEEE 802.1Q VLANs.
- The *dot1qTp* Group contains objects that describe the operation and status of transparent bridging, including the dynamic filtering databases for unicast and multicast forwarding.
- The *dot1qStatic* Group contains objects that describe static configurations for transparent bridging, including static entries in the filtering databases for unicast and multicast forwarding.
- The *dot1qVlan* Group contains objects that describe the configuration and status of VLANs, including statically configured VLANs and VLANs configured dynamically by protocols like GVRP.

The following objects are implemented:

- All objects in the *dot1qBase* Group.
- The *dot1qVlanNumDeletes* object in the *dot1qVlan* Group.
- The *dot1qVlanCurrentTable* object in the *dot1qVlan* Group.
- The *dot1qVlanStaticTable* object in the *dot1qVlan* Group.
- The *dot1qNextFreeLocalVlanIndex* object in the *dot1qVlan* Group.
- The *dot1qPortVlanTable* object in the *dot1qVlan* Group.

MAC Address Thrashing Protection for Ports

This enhancement reduces the effect of MAC address thrashing on the switch ports. Thrashing occurs when the same MAC addresses are constantly learned on a group of ports, for example, as a result of a switching loop. Thrashing protection is available by default, but you can change the threshold at which the switch considers that a port is thrashing, by using the new **thrashlimit** and **thrashrefill** parameters in the command:

```
SET SWItch POrt={port-list|ALL} [THRASHLimit={NONE|1..65536}]
    [THRASHRefill=1..65536] [other-options...]
```

The **thrashlimit** parameter specifies the maximum number of MAC address learning events that can occur in a single burst on the port before the port is considered to be thrashing. The switch limits the effect of thrashing by disabling MAC address learning on a thrashing port for 1 second. After the switch re-enables learning, the number of MAC addresses that a port can learn in a burst depends on the time elapsed since thrashing and the value of the **thrashrefill** parameter, up to a maximum of **thrashlimit** (using a token bucket model). Thrashing protection operates independently of the port security feature configured using the **learn** parameter, and does not apply to trunked ports. If you specify **none**, thrashing protection is disabled. The default is 8192, which is suitable for most network scenarios.

The **thrashrefill** parameter specifies the rate at which the port's thrash limit recovers after thrashing, in MAC address learning events per second. The **thrashrefill** value must be less than the thrash limit, which is enforced by reducing the thrash refill if necessary. Thrashing protection does not apply to trunked ports. The default is 1024, which is suitable for most network scenarios.

The **show switch port** command has been modified to display the settings for thrashing protection and the current state of the port, as shown in bold in Figure 13 on page 53.

Table 12: Thrashing protection parameters in the output of the **show switch port** command

Parameter	Meaning
Current learned, lock state	Number of MAC addresses currently learned on this port and the state of locking for this port. The current learned parameter is only incremented if there is a Learn Limit set for the port. The lock state is one of: not locked, locked by limit, locked by command, or locked by thrashing.
Address learn thrash limit	Thrashing protection information; or Disabled if thrashing protection is disabled; or Trunk if the port is trunked and therefore thrashing protection does not apply. The thrashing protection information starts with the number of MAC address learning events that the port will currently accept before locking, followed in parentheses by the maximum burst of learn events that the port will accept, the rate at which the learn limit regenerates, and whether the port is currently locked because of excessive address learning.

Figure 13: Example output of the **show switch port** command

```
 Switch Port Information
------------------------------------------------------------
Port ........................... 9
   Description ................... -
   Status ........................ ENABLED
   Link State .................... Up
   UpTime ........................ 00:00:06
   Port Media Type ............... ISO8802-3 CSMACD
   Configured speed/duplex ....... Autonegotiate
   Actual speed/duplex ........... 1000 Mbps, full duplex
   MDI Configuration (Polarity) .. Automatic (MDI)
   Loopback ...................... Off
   Configured master/slave mode .. Not applicable
   Actual master/slave mode ...... Not applicable
   Acceptable Frames Type ........ Admit All Frames
   Disabled Egress Queues ........ 3
   BCast & MCast rate limit ...... -
   BCSC rate Limiting ............ disabled
   Egress rate limit ............. 9072 Kb/s
   Learn limit ................... -
   Intrusion action .............. Discard
   Current learned, lock state ... 0, locked by thrashing
   Address learn thrash limit .... 0 (8192 max, 1024 per second, locked)
   Relearn ....................... OFF
   Mirroring ..................... Disabled
   Is this port mirror port ...... No
   Enabled flow control(s) ....... -
   Ingress Filtering ............. Off
   Trunk Group ................... -
   STP ........................... default
   Cable Length .................. <50m
------------------------------------------------------------
```

Recovery Disposal to PPPoE

If the primary link fails, either LQM or ECHO detects the failure and resets the link. Previously, only LQM could do this.

TPAD Over TCP/IP Improvements

When an AR440S or AR441S router is configured as a TPAD router, it can now:

- automatically establish an X.25 and ISDN connection when it receives a TPAD message over TCP/IP.
- automatically close the ISDN connection when the TPAD connection is idle.
- read an `NNUI` string it receives as `ANUA:NUI`.

The **create tpad** and **set tpad** commands have new parameters for controlling these features: **autodial**, **autodisconnecttime**, **autonnuiconversion, autonua**, **autonui**. The **show tpad** command displays corresponding new output parameters.

Automatically connect X.25

When the router receives an APACS 30 standard message from a transaction terminal over a TCP port, by default it now automatically establishes an X.25 connection to the authorisation service specified by the **over** parameter, activating the LAPB and ISDN connections required for this X.25 interface. It uses the NUA (Network User Address) specified by the **autonua** parameter (by default **autonua=13)**, and the NUI (Network User Information) specified by the **autonui** parameter (by default **autonui=natwest_test)**. To turn this automatic dialling (**autodial**) off or on, or to modify the NUA or NUI it uses to connect, use one of the commands:

```
CREate TPAD=name OVer=x25-interface [AUTODIAl={YES|NO}]
    [AUTONUA=nua] [AUTONUI=nui] [other-tpad-parameters]

SET TPAD=name [AUTODIAl={YES|NO}] [AUTONUA=nua] [AUTONUI=nui]
    [other-tpad-parameters]
```

Automatically disconnect call

The router can then automatically close the X.25 and ISDN connections when the TPAD instance has been idle for a specified time (**autodisconnecttime**). By default, **autodisconnecttime=0** — it does not timeout. To configure TPAD to automatically close an ISDN call, use one of the commands:

```
CREate TPAD=name OVer=x25-interface
    [AUTODISconnecttime=0..65535] [other-tpad-parameters]

SET TPAD=name [AUTODISconnecttime=0..65535]
    [other-tpad-parameters]
```

Automatically convert NUI

By default, when the TPAD router receives a character string from the transaction terminal, it converts strings like this:

```
Nnui
```

to strings like this:

```
Anua:nui
```

where:

- *nua* is the Network User Address (NUA) specified by the **autonua** parameter.

By default, it uses the NUA (Network User Address) **autonua=13**. To turn this automatic conversion off or on, or to modify the NUA, use one of the commands:

```
CREate TPAD=name OVer=x25-interface
   AUTONNuiconversion={YES|NO} [AUTONUA=nua]
   [other-tpad-parameters]

SET TPAD=name  AUTONNuiconversion={YES|NO} [AUTONUA=nua]
   [other-tpad-parameters]
```

Display TPAD Settings

To display the TPAD automatic dialling, automatic disconnection and automatic conversion settings, use the command:

```
SHOW TPAD[=name]
```

Figure 14: Example output from the **show tpad** command.

```
TPAD Configuration
Status ..................... ENABLED
List of Instances:
------------------------------------------------------------
OurBank:
   Status ................................. Enabled
   Debug .................................. Enabled
   Checksum ............................... On
   Transfer Type .......................... Multithread
   Tcp Port ............................... 1024
   Over ................................... x25t0
   Max-Transfer ........................... 64
   Max-Connections ........................ 8
   Source ................................. All
   Number retries on Call ................. 2
   Fast Disconnect(ATH) ................... 2
   ISDN call Name ......................... 2
   No Carrier Response .................... ON
   Silently Retry Call .................... OFF
   RL Automatic Dialling .................. ON
   RL Automatic NUA ....................... 13
   RL Automatic NUI ....................... natwest_test
   RL Automatic NNUI->ANUA:NUI ............ OFF
   RL Automatic Disconnect Time............ 0
------------------------------------------------------------
```

Table 13: New parameters displayed in the output of the **show tpad** command

Parameter	Meaning
RL Automatic Dialling	Whether the router automatically establishes an X.25 connection when it receives an APACS 30 message from a transaction terminal.
RL Automatic NUA	The NUA (network user address) that the router uses to convert NNUI to ANUA:NUI if **autonnuiconversion=yes**, and to establish the X.25 connection if **autodial=yes**.
RL Automatic NUI	The NUI (network user information) that the router uses to establish the X.25 connection if **autodial=yes**.
RL Automatic NNUI->ANUA:NUI	Whether the router automatically converts NNUI to ANUA:NUI.
RL Automatic Disconnect Time	How long, in seconds, the TPAD instance remains idle before the router automatically disconnects its X.25 and ISDN connections, or 0 if it does not timeout.

Ping and Trace Using DNS

You can now ping or trace the route to a domain name. The router or switch will perform a DNS lookup to resolve the name. The new syntax is:

```
ping domain-name [other-options...]
trace domain-name [other-options...]
```

To configure a DNS for the router or switch to use to resolve domain names, use the command:

```
add ip dns primary=ipadd [secondary=ipadd] [other-options...]
```

You can also specify a domain name as the default destination, using the commands:

```
set ping [ipaddress=]domain-name [other-options...]
set trace [ipaddress=]domain-name [other-options...]
```

The router or switch stores the domain name as the default, and performs a DNS lookup to resolve the name when you enter the **ping** or **trace** command.

The **show ping** and **show trace** commands display the domain name in the destination fields and, if the router or switch has resolved the name, the IP address to which it has been resolved.

Blackhole Routing

A blackhole route allows the switch to silently drop packets destined for a specified IP address.

In some network configurations, when an interface goes down, packets addressed to that interface cause a broadcast storm. Blackhole routing prevents this. For example, consider the network shown in Figure 15 on page 58. The first section of the figure shows the normal routing situation. If a host in vlan1 sends a packet to a host in vlan2, switch A looks up its routing table, finds an interface route to 192.168.2.0/24 on vlan2, and forwards the packet correctly. However, if vlan2 goes down, as shown in the middle section of the figure, the switch no longer finds a functional route to vlan2 in its routing table. The switch then uses its default route and sends the packet over vlan3 to switch B. Switch B looks up its routing table, finds a static route for 192.168.2.0/24 on vlan3, so returns the packet to switch A. Switch A forwards it to switch B again, and so on.

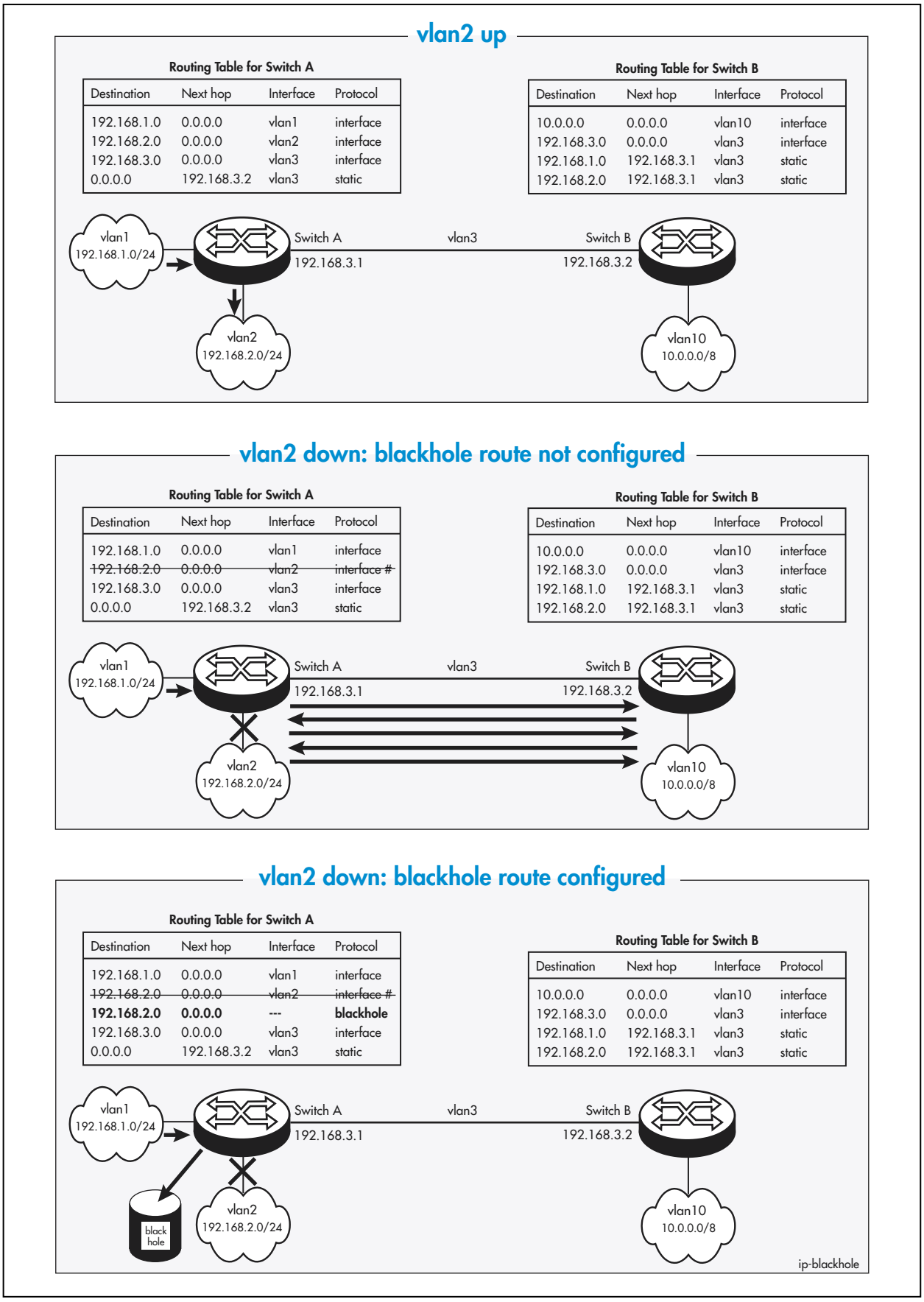
In the bottom section of the figure, switch A has a blackhole route defined for 192.168.2.0/24. This is a route that drops the packet by sending it to a virtual interface. If vlan2 is down, switch A drops packets destined for 192.168.2.0/24, instead of sending them to switch B. By default, the blackhole route has a higher preference value than an interface route, but a lower preference value than a dynamic route or the default route. This makes the blackhole route the preferred route when the interface goes down. It also means that the default route will never be used for packets to a specified destination if a blackhole to that destination exists. The default preference for a blackhole route is 5.

Caution: A blackhole route points to a virtual interface, so the interface is always “up”, and is always available to become the preferred route.

You can also use blackhole routes to limit the impact of a DOS attack. If a device in your network comes under a DOS attack, you can give the edge switches a blackhole route to the device under attack. The edge switches would then silently drop traffic that is destined for the device under attack, which would prevent the excess traffic from entering the network.

If the device under attack is in a subnet that is attached to an edge switch, the subnet would still be under attack because the edge switch still has an interface route to that subnet. In this case, you need to give the device under attack a 32-bit mask (255.255.255.255) and configure the blackhole route to point to the device’s address and the 32-bit mask. We also recommend that you give the device under attack a 32-bit mask if there are other devices on the same subnet, so that traffic destined for the other devices is still routed.

Figure 15: An example of blackhole routing preventing a network storm



How to configure a blackhole route

To create a blackhole route, use the command:

```
add ip route=ipadd blackhole [mask=ipadd] [metric=1..16]
    [preference=0..65535]
```

Note that **mask**, **metric**, and **preference** are the only other IP route parameters that are valid for blackhole routes.

To remove a blackhole route, use the command:

```
delete ip route=ipadd blackhole [mask=ipadd]
```

To modify a blackhole route, use the command:

```
set ip route=ipadd blackhole [mask=ipadd] [metric=1..16]
    [preference=0..65535]
```

How to see information about blackhole routes

To display details of blackhole routes, use the command:

```
show ip route[=ipadd] [{general|cache|count|full}]
```

You can identify blackhole routes because they have a **Protocol** entry of "blackhole", and no interface (indicated by an **Interface** entry of "-"). The following figure shows an example of an entry for a blackhole route, in bold text.

```
IP Routes
-------------------------------------------------------------------------------
Destination       Mask              NextHop            Interface          Age
DLCI/Circ.        Type     Policy   Protocol   Tag     Metrics      Preference
-------------------------------------------------------------------------------
0.0.0.0           0.0.0.0           202.36.163.21      vlan1                1
-                 remote   0        rip        -       5                  100
10.0.0.0          255.0.0.0         0.0.0.0            vlan1                4
-                 direct   0        interface  -       1                    0
10.0.0.0          255.0.0.0         0.0.0.0            -                  123
-                 direct   0        blackhole  -       1                    5
11.0.1.0          255.255.255.0     10.42.0.22         vlan1                4
-                 direct   0        static     -       1                   60
192.168.69.0      255.255.255.0     202.36.163.35      vlan1                0
-                 remote   0        rip        -       2                  100
192.168.201.0     255.255.255.0     202.36.163.21      vlan1                1
-                 remote   0        rip        -       5                  100
192.168.202.0     255.255.255.0     202.36.163.21      vlan1                1
-                 remote   0        rip        -       6                  100
-------------------------------------------------------------------------------
```

Hardware Equal Cost Multipath Routing (ECMP)

Equal Cost Multipath Routing (ECMP) allows the switch to distribute traffic over multiple equal-cost routes to a destination. When the switch learns such multiple routes, it puts them in an ECMP route group. When it sends traffic to that destination, it distributes the traffic across all routes in the group.

AT-8900 and AT-9900 series switches can now perform ECMP routing in their switching hardware (*hardware ECMP*), as well as in software in their CPU (*software ECMP*). The key differences are:

	Hardware ECMP	**Software ECMP**
Routes are equal cost if they have the same ...	destination IP address and mask.	destination IP address, mask, preference and metric.
Traffic is distributed over the routes ...	one flow at a time, so all packets in a session take the same route.	one packet at a time, so different packets in a session take different routes.
Each equal-cost route group can contain ...	8 individual routes.	16 individual routes.
Routing is ...	wire-speed.	not wire-speed.

Configuring ECMP

Table 14: Procedure for using hardware ECMP

Step	**Command**	**Action**
1	enable ip route multipath	If ECMP routing has been disabled, enable it. ECMP routing is enabled by default.
2	add ip route=*ipadd* interface=*interface* nexthop=*ipadd* [*other-options...*]	Add static routes as required. You can create multiple static routes to the same destination.
3		Configure dynamic routing protocols as required.

Table 15: Procedure for using software ECMP

Step	**Command**	**Action**
1	enable ip route multipath	If ECMP routing has been disabled, enable it. ECMP routing is enabled by default.
2	create classifier=*rule-id* ipdaddr=*ipaddmask* add switch hwfilter=*filter-id* classifier=*rule-id* action=copy,discard	If necessary, create a hardware filter for the traffic that you want to apply software ECMP routing to, so that the switch sends it to the CPU. The switch also automatically sends packets to the CPU if it does not have a valid route for the packet in its hardware routing table.

Table 15: Procedure for using software ECMP

Step	Command	Action
3	add ip route=*ipadd* interface=*interface* nexthop=*ipadd* [*other-options...*]	Add static routes as required. You can create multiple static routes to the same destination.
4		Configure dynamic routing protocols as required.
5	set ip route preference=1..65535 protocol={bgp-ext|bgp-int|ospf-ext1|ospf-ext2|ospf-inter|ospf-intra|ospf-other|rip}	If you want routes from different routing protocols to have equal cost, give the protocols the same preference setting.

To disable ECMP, use the command:

```
disable ip route multipath
```

Hardware IP Route Learning Delay

This enhancement enables you to improve software route convergence time by delaying hardware route learning until after software route processing. It is intended for switches that learn a very large number of IP routes. The default setting will be suitable for most other networks. To change the delay, use the new command:

```
SET SWItch HWLearndelay=0..100000
```

This command sets the length of time, in milliseconds, between activity in the IP route learning system and the beginning of hardware route learning. The default value is 4.

In environments with a very high number of IP routes in use, increasing this delay to several seconds allows you to prioritise software route processing higher than hardware route learning, which improves the software route convergence time. It also increases the latency of hardware route learning. Departure from the default setting is not recommended for most network systems, and may impact the routing and CPU performance of the switch.

For example, to force the switch to wait 5 seconds after the last IP route update before updating hardware routing, use the command:

```
set swi hwl=5000
```

The **show switch** command has been modified to display the delay, in the form:

```
IP route learn delay ....... 4 ms
```

Disable Source Routing

By default, all source-routed packets are discarded. New options have been added to the commands **enable ip srcroute** and **disable ip srcroute**, to allow finer control of which packets are forwarded or discarded.

Modified Commands

enable ip srcroute

Syntax `ENAble IP SRCRoute[={LOOSE|STrict|ALL}]`

Description This command enables the forwarding of source-routed IP packets. If a specific type of source-routed IP packet is specified, as defined in RFC 791, forwarding of that type is enabled. Otherwise, forwarding of all source-routed IP packets is enabled.

disable ip srcroute

Syntax `DISable IP SRCRoute[={LOOSE|STrict|ALL}]`

Description This command disables the forwarding of source-routed IP packets. If a specific type of source-routed IP packet is specified, as defined in RFC 791, forwarding of that type is disabled. Otherwise, forwarding of all source-routed IP packets is disabled.

Local Interfaces

A local interface is one that is always available for higher layer protocols to use and advertise to the network. Although a local interface is assigned an IP address, it does not have the usual requirement of connecting to a lower layer physical entity. This lack physical attachment creates the perception of a local interface always being accessible via the network.

Local interfaces can be utilised by a number of protocols for various purposes. They can be used to improve access to a router or switch, as well as increasing its reliability, security, scalability and protection. In addition, local interfaces can add flexibility and simplify management, information gathering and filtering.

One example of this increased reliability is for OSPF to advertise a local interface as a interface-route into the network irrespective of the physical links that may be “up” or “down” at the time. This provides a higher probability that the routing traffic will be received and subsequently forwarded. Further reliability and performance could be provided by configuring parallel BGP paths to a local interface on a peer device, which would result in improved load sharing.

Access and security can be improved through filtering. Incoming traffic can be filtered by rules that specify local interfaces as the only acceptable destination addresses.

Information gathering and filtering as well as management can potentially be simplified if protocols such as SNMP use local interfaces for receiving ender sending trap and log type information.

To add a new local interface, use the **add ip local** command.

To delete a new local interface, use the **delete ip local** command.

Using the Local Interface in BGP

When the router or switch is acting as a BGP speaker, it uses an IP address to identify itself to its peers in these situations:

- when establishing the TCP session and sending TCP messages
- in the *open* message it sends at the beginning of the session
- when it considers itself to be the next hop for a route that it is advertising to its peers.

In open messages and when it is the next hop, the router or switch may use the IP address of the local interface to identify itself. The following sections explain this in detail.

Address selection rules

The address the router or switch uses in each of these situations depends on the situation and whether you have configured a router ID or a local interface address. The rules for each situation are:

1. TCP session source address

   If a local IP address has been set for the peer, use it. Otherwise allow TCP to select a source IP address, which it will do based on the outgoing interface.

2. BGP Identifier in *open* message

If the router ID has been set, use it. Otherwise, if a local IP address has been set for the peer, use that. If neither has been set, use the highest IP address configured on any of the router or switch's interfaces.

3. Next hop address

   If the router or switch learned the route from an IBGP peer, use the learned next hop address—the next hop that the IBGP peer supplied for the route.

   If the router or switch learned the route from an EBGP peer and the learned next hop is in the same subnet as the router or switch, use the learned next hop.

   If the router or switch learned the route from an EBGP peer and the learned next hop is in a different subnet to the router or switch, then:

   - if a local IP address has been set for the peer to which the router or switch is sending the update, use it
   - otherwise, if the router or switch has an IP route to that network, use the IP address of the interface via which the route reaches that network
   - otherwise, use the IP address of the interface via which the router or switch reaches the peer to which it is sending the update

How to configure router ID

To configure a router ID, use the command:

```
set bgp routerid=ipadd [other-options...]
```

How to configure local interface

To configure a local interface, first create the local interface and give it an IP address by using the command:

```
add ip local=1..15 ipaddress=ipadd [other-options...]
```

Then apply the local interface to the BGP peer by using one of the commands:

```
add bgp peer=ipadd remoteas=1..65534 local=1..15
   [other-options...]
```

```
set bgp peer=ipadd local=1..15 [other-options...]
```

New commands

add ip local

Syntax

```
ADD IP LOCal=1..15 IPaddress=ipadd [FILter={0..99|None}]
   [GRE=[0..100|None}] [POLicyfilter={100..199|None}]
   [PRIorityfilter={200..299|None}]
```

where:

- *ipadd* is an IP address in dotted decimal notation.

Description

This command adds a local interface to the router or switch. Up to fifteen local interfaces can be added to a single router or switch. These are in addition to the default local interface that is automatically added at start up, and can be configured through the **set ip local** command. A local interface is *virtual* in the sense that it is not associated with a physical interface. Each local interface can be assigned an IP address, which can then be used as the source address of IP packets generated internally by IP protocols such as RIP, OSPF, PING and NTP.

Higher layer protocols such as RIP, OSPF, PING and NTP must assign a source IP address to packets passed to IP for forwarding.

The following rules are used to determine which IP address to use as the source address:

1. If the higher layer protocol's configuration specifies the use of either a source IP address or a local interface, then the configured address is used as the packet's source IP address. For example, the **sipaddress** parameter of the **ping** command specifies the source IP address to use in ping packets. While the **local** parameter of the **add bgp peer** command specifies a local interface to use to obtain a source IP address.
2. If the default local interface has been assigned an IP address, then this will be used as the packet's source IP address. Otherwise, the IP routing module determines the interface over which the packet is to be transmitted, and assigns the IP address of the interface as the packet's source IP address.

The **local** parameter specified is a unique identifying number that is used to identify a particular local interface. The naming convention, or alias, for this interface is the concatenation of the word *local* along with this identifying number.

The **filter** parameter specifies which filter will be applied to IP packets transmitted or received over the interface. The filter must already have been defined with the **add ip filter** command. An interface may have a maximum of one traffic filter, one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the interface. The default is not to apply a filter.

The **gre** parameter specifies the GRE (Generic Routing Encapsulation) entity associated with the interface. The specified GRE entity must have been created previously using the **add gre** command. The default is NONE.

The **ipaddress** parameter specifies the IP address of the interface. This must be the IP address of one of the router or switch's active IP interfaces. Note that specifying an IP address of 0.0.0.0 effectively 'unsets' the IP address of the local interface specified.

The **policyfilter** parameter specifies the policy filter that will be applied to IP packets received over the interface. The filter must already have been defined using the **add ip filter** command. Although an interface can only have one traffic filter, one policy filter and one priority filter; each of these filters can be assigned to more than one interface. Policy filters are applied to packets as they are transmitted. The default setting is **none**, that is, not to apply a filter.

The **priorityfilter** parameter specifies the filter that will be applied to IP packets received over the interface. The filter must have already been defined with the **add ip filter** command. Although an interface can only have one traffic filter, policy filter, and priority filter; each of these filters can be assigned to more than one interface. Priority filters are applied to packets as they are transmitted. The default setting is **none**, that is, not to apply a filter.

Examples To add the local interface 3 with an IP address of 192.168.33.1, use the command:

```
add ip loc=3 ip=192.168.33.1
```

Related Commands

delete ip local
set ip local

delete ip local

Syntax `DELete IP LOCal=1..15`

Description This command deletes a local interface from the IP module. The selected local interface will no longer be used by the IP routing module.

When an IP interface is deleted, any static routes and ARP entries specific to the interface will also be deleted.

Examples To delete local interface 5, use the command:

```
del ip local=5
```

Related Commands **add ip local**
set ip local

set ip local

Syntax

```
SET IP LOCal[={DEFault|1..15}] [FILter={0..99|None}]
   [GRE={0..100|None}] [IPaddress=ipadd]
   [POLicyfilter={100..199|None}]
   [PRIorityfilter={200..299|None}]
```

where:

- *ipadd* is an IP address in dotted decimal notation

Description This command modifies the parameters of one the router or switch's local interfaces. If the LOCAL argument is either not specified or DEFAULT, then the router or switch's default local interface is modified.

Each of the local IP interfaces are virtual and are able to represent the IP routing module itself. Each of the local interface interfaces can be assigned IP addresses that can then be used as the source address of IP packets generated internally by IP protocols such as RIP, OSPF, PING and NTP. Higher layer protocols such as RIP, OSPF, PING and NTP must assign a source IP address to packets passed to IP for forwarding. Use the following rules to determine which IP address to use as the source address:

1. If the higher layer protocol's configuration specifies a source IP address to use, then the configured address is used as the packet's source IP address.

   For example, the **sipaddress** parameter in the **ping** command specifies the source IP address to use in ping packets.

2. If a local IP interface has been assigned an IP address, then the IP address of that local interface is used as the packet's source IP address.

3. Otherwise, the IP routing module determines the interface over which the packet is to be transmitted, and assigns the IP address of the interface as the packet's source IP address.

The **filter** parameter specifies the filter to apply to IP packets transmitted or received over the interface. The filter must already have been defined with the **add ip filter** command. An interface may have a maximum of one traffic filter,

one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the interface. The default is not to apply a filter.

The **gre** parameter specifies the GRE (Generic Routing Encapsulation) entity associated with the interface. The specified GRE entity must have been created previously using the **add gre** command in the Generic Routing Encapsulation (GRE) Chapter. The default is **none**.

The **ipaddress** parameter specifies the IP address of the interface. The IP address must be the IP address of one of the router or switch's active IP interfaces. Specifying an IP address of 0.0.0.0 effectively 'unsets' the IP address of the local interface.

The **policyfilter** parameter specifies the policy filter to apply to IP packets received over the interface. The filter must already have been defined with the **add ip filter** command. An interface may have a maximum of one traffic filter, one policy filter, and one priority filter, but the same traffic, policy, or priority filter can be assigned to more than one interface. Policy filters are applied to packets as they are transmitted. The default is not to apply a filter.

The **priorityfilter** parameter specifies the priority filter to apply to IP packets transmitted over the interface. The filter must already have been defined with the **add ip filter** command. An interface may have a maximum of one traffic filter, one policy filter, and one priority filter, but the same traffic, policy, or priority filter can be assigned to more than one interface. Priority filters are applied to packets as they are transmitted. The default is not to apply a filter.

Examples

To set the IP address of the local IP interface to 192.168.33.11, use:

```
set ip loc ip=192.168.33.11
```

```
To set the local interface 3 to 192.168.33.11, use:
```

```
set ip local=3 ip=192.168.33.1
```

To remove the IP address of the local IP interface, use:

```
set ip loc ip=0.0.0.0
```

Related Commands

add ip local
delete ip local
set ip local

set snmp local

Syntax

```
SET SNmp LOCal={NONE|1..15} [VERsion={V1|V2|V3|ALL}]
```

Description

This command sets the local interface to be used with a particular version of SNMP. Once set, the IP address of the local interface specified will be used as the source IP address for all SNMP packets of the version specified.

The **version** parameter specifies which version of SNMP packets that the local interface will apply to. The default is ALL.

The **local** parameter specifies a local interface to be used as the source IP address for all packets of a particular SNMP version that the router or switch

generates and sends. The local interface IP address is also be used as the SNMP agent IP address in these outgoing packets. The local interface must already be configured and be in the range 1 to 15. If no local interface has been set for SNMP, the router or switch will select a source address on the bases of the route, i.e. the source address will be the IP address that the SNMP packet will be issued from.

Examples To set the local interface 5 for SNMPv3 packets, use the command:

```
set snmp loc=5 ver=v3
```

Related Commands **show snmp**

Modified commands

These commands have been modified for the Local Interfaces enhancement to include the parameters shown in **bold**.

add bgp peer
set bgp peer
add bgp peertemplate
set bgp peertemplate

Syntax

```
ADD BGP PEer=ipadd REMoteas=1..65534 [AUthentication={MD5|
   NONE}] [CLIEnt={NO|YES}] [CONnectretry={DEFault|
   0..4294967295}] [DESCription=description]
   [EHOps={DEFault|1..255}] [FASTFallover={NO|YES}]
   [HOLdtime={DEFault|0|3..65535}] [INFilter={NONE|
   300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}] [NEXthopself={NO|
   YES}] [OUTFilter={NONE|300..399}] [OUTPathfilter={NONE|
   1..99}] [OUTRoutemap=routemap] [PASSword=password]
   [PRIVateasfilter={NO|YES}] [SENdcommunity={NO|YES}]

SET BGP PEer=ipadd [AUthentication={MD5|NONE}]
   [CLIEnt={NO|YES}] [CONnectretry={DEFault|
   0..4294967295}] [DESCription=description]
   [EHOps={DEFault|1..255}] [FASTFallover={NO|YES}]
   [HOLdtime={DEFault|0|3..65535}] [INFilter={NONE|
   300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=[routemap]] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}] [NEXthopself={NO|
   YES}] [OUTFilter={NONE|300..399}] [OUTPathfilter={NONE|
   1..99}] [OUTRoutemap=[routemap]] [PASSword=password]
   [PRIVateasfilter={NO|YES}] [REMoteas=1..65534]
   [SENdcommunity={NO|YES}]
```

```
ADD BGP PEERTemplate=1..30 [CLIEnt={NO|YES}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description] [HOLdtime={DEFault|0|
   3..65535}] [INFilter={NONE|300..399}]
   [INPathfilter={NONE|1..99}] [INRoutemap=routemap]
   [KEEpalive={DEFault|1..21845}] [LOCal={NONE|1..15}]
   [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}] [NEXthopself={NO|
   YES}] [OUTFilter={NONE|300..399}] [OUTPathfilter={NONE|
   1..99}] [OUTRoutemap=routemap] [PRIVateasfilter={NO|
   YES}] [SENdcommunity={NO|YES}]

SET BGP PEERTemplate=1..30 [CLIEnt={NO|YES}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description] [HOLdtime={DEFault|0|
   3..65535}] [INFilter={NONE|300..399}]
   [INPathfilter={NONE|1..99}] [INRoutemap=routemap]
   [KEEpalive={DEFault|1..21845}] [LOCal={NONE|1..15}]
   [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}] [NEXthopself={NO|
   YES}] [OUTFilter={NONE|300..399}] [OUTPathfilter={NONE|
   1..99}] [OUTRoutemap=routemap] [PRIVateasfilter={NO|
   YES}] [SENdcommunity={NO|YES}]
```

Description The **local** parameter specifies the local interface. In certain circumstances, the router or switch uses this address as the source for BGP packets it generates and sends to this BGP peer, or to peers that use this peer template. For a description of when the router or switch uses the local interface, see "Using the Local Interface in BGP" on page 63.

set bgp

Syntax

```
SET BGP [CLUSter=ipadd] [CONfederationid={NONE|1..65534}]
   [LOCalpref={DEFault|0..4294967295}] [MED={NONE|
   0..4294967294}] [PREFExt={DEFault|1..255}]
   [PREFInt={DEFault|1..255}] [ROuterid=ipadd]
   [SELEction_timer=3..60] [TABlemap[=routemap]
```

Description The **routerid** parameter specifies a 4-byte number that uniquely identifies the router or switch in a network system in certain circumstances, specified as an IP address in dotted decimal notation. For a description of when the router or switch uses the router ID, see "Using the Local Interface in BGP" on page 63. The default is the default local interface's IP address, if it is configured. Otherwise, the default is the highest interface IP address on the router or switch.

add pim brscandidate
set pim brscandidate

Syntax

```
ADD PIM BSRCandidate [BSMinterval={10..15000|DEFault}]
   [HAShmasklength=0..32] [INTerface={local-interface|
   vlan-interface}] [PREFerence=0..255]

SET PIM BSRCandidate [HAShmasklength=0..32]
   [INTerface={local-interface|vlan-interface}]
   [PREFerence=0..255]
```

Description The **interface** parameter specifies an interface for the router or switch to use when advertising itself as a candidate bootstrap router. The IP address of the of this interface will be advertised by the router or switch. The **interface** supplied can be either a configured local interface or a configured VLAN interface. If the parameter is not specified the router or switch will advertise its first active IP interface instead.

add pim rpcandidate
set pim rpcandidate

Syntax

```
ADD PIM RPCandidate[=rp-address] GROup=group-address
   [ADVinterval={10..15000|DEFault}]
   [INTerface={local-interface|vlan-interface}]
   [MASK=ipaddress] [PRIOrity=0..255]

SET PIM RPCandidate GROup=group-address
   [INTerface={local-interface|vlan-interface}]
   [MASK=ipaddress] [PRIOrity=0..255]
```

Description The **interface** parameter specifies an interface for the router or switch to use when advertising itself as the candidate rendezvous point for a multicast group. The IP address of the of this interface will be advertised by the router or switch. The **interface** supplied can be either a configured local interface or a configured VLAN interface. If the parameter is not specified the router or switch will advertise its first active IP interface instead.

add tacplus server
set tacplus server

Syntax

```
ADD TACPlus SERVer=ipaddress [Key=key] [LOCal={NONE|
   1..15}] [PORT=port] [SINGLEconnection={Yes|No]
   [TIMEOUT=1..10]

SET TACPlus SERVer=ipaddress [Key=key] [LOCal={NONE|
   1..15}] [POrt=port] [SINGLEconnection={Yes|No}]
   [TIMEOUT=1..10]
```

Description The **local** parameter specifies a local interface to be used as the source for all TACACS+ packets the device sends to this TACACS+ server. The local interface must already be configured and be in the range 1 to 15. If either the parameter is not set or the option **none** is specified the router or switch will select a source from the current available interfaces instead.

add radius server

Syntax `ADD RADius SERVER=`*`ipadd`* `SECret=`*`secret`* `[POrt=`*`port-number`*`]`
`[ACCPort=`*`port-number`*`]` **`[LOCal={NONE|1..15}]`**

Description The **local** parameter specifies a local interface to be used as the source for all **radius** packets the router or switch generates and subsequently sends to this **radius** server. The local interface IP address will also be used as the NAS IP address in these outgoing packets. The local interface must already be configured and be in the range 1 to 15. If either the parameter is not set or the option **none** is specified the router or switch will select a source from the current available interfaces instead.

Use of Default Route by Proxy ARP

Proxy ARP is used by the router or switch when there are hosts on the LAN that have not been configured with correct information about which IP subnets are being used on the local LAN and which are being used on remote LANs. Such hosts are likely to send ARP requests for IP addresses outside the range of addresses being used on the local LAN. If the router or switch knows a route to the address being erroneously ARPed for, then the router or switch intercepts the ARP broadcast packets and sends replies in which it substitutes its own physical address for that of the remote host. By responding to the ARP request, the router or switch ensures that all subsequent packets from the local host to that remote host are directed to the router or switch's physical address and it can then forward these to the remote host.

Prior to this enhancement, the router or switch only acted as a proxy when it had a specific route to the remote host. This enhancement allows the router or switch to also send proxy ARP responses for hosts that it would reach via the default route.

Modified Commands

To enable proxy ARP to use the default route, use the highlighted new options in the following existing commands.

The PArp entry in the output of the **show ip interface** command has also been modified, and now indicates whether this interface supports proxy ARP and if ARP responses will be generated even for hosts that can only be reached via the default route.It is one of “On” (respond to ARP requests only if a specific route exists), “Def” (respond to ARP requests if a specific route or a default route exists), or “Off”.

add ip interface

Syntax

```
ADD IP INTERFACE=interface [PROXYARP={FALSE|NO|OFF|ON|
   TRUE|YES|STRICT|DEFROUTE}] [other-options]
```

Description The **proxyarp** parameter enables or disables proxy ARP responses to ARP requests. If you specify **strict, on, true** or **yes**, the router or switch will respond to proxy ARP requests using specific routes if they exist. If you specify **off, false** or **no**, the router or switch will not respond to ARP requests. If you specify **defroute**, the router or switch will respond to proxy ARP Requests using specific routes if they exist or a default route (0.0.0.0) if it exists. The **proxyarp** parameter is valid for VLAN interfaces. The default is **off**.

set ip interface

Syntax `SET IP INTERFACE=`*`interface`* `[PROXYARP={FALSE|NO|OFF|ON|TRUE|YES|`**`STRICT`**`|`**`DEFROUTE`**`}] [`*`other-options`*`]`

Description The **proxyarp** parameter enables or disables proxy ARP responses to ARP requests. If you specify **strict, on, true** or **yes**, the router or switch will respond to proxy ARP requests using specific routes if they exist. If you specify **off, false** or **no**, the router or switch will not respond to ARP requests. If you specify **defroute**, the router or switch will respond to proxy ARP Requests using specific routes if they exist or a default route (0.0.0.0) if it exists. The **proxyarp** parameter is valid for VLAN interfaces. The default is **off**.

Support for 5000 Next Hops

This enhancement enables the switch to store an increased number of routes in hardware.

New Command

set switch enhancedmode

Syntax `SET SWItch ENHancedmode={QOSCounters|NEXThop|NONE}`

Description This command rearranges the switch's memory either so that it can store QoS traffic class counters or so that it can store a greater number of routes than usual.

If you specify **enhancedmode=qoscounters**, the switch monitors QoS counters for traffic classes. The maximum number of traffic classes is reduced.

If you specify **enhancedmode=nexthop**, the switch stores up to 5000 individual routes (nexthops). The maximum number of multicast groups and traffic classes is reduced.

Modified Command

The **show switch** command has been modified to display the **enhancedmode** setting. When this enhancement is enabled, a line in the output shows

```
EnhancedMode operation ..... NEXTHOP
```

BGP Improvements

The BGP Chapter has been revised and now contains more procedures describing how to configure BGP. The following enhancements to BGP are detailed in the improved and extended BGP chapter at the end of this Release Note.

Soft Resetting of Modified Peers

It is no longer necessary to disable a peer before modifying its settings. You can choose to configure the switch or router to automatically update modified peers, or to manually update each peer after changing it.

Peer Templates

Instead of configuring settings for each peer individually, you can now create peer templates. Peer templates make it easier to create BGP peers when many peers have identical inbound and outbound filtering policies, or timer values.

Extensions to Route Maps

You can now create route maps that match on the next hop or origin attribute of update messages. You can also use route maps to remove the Multi-Exit Discriminator (MED) attribute from update messages.

Prefix Lists

Prefix lists are a list of matched subnets. Creating a prefix list and applying it in a route map increases your control over the routes you import into BGP or advertise. You can filter according to the prefix list, or change the attributes of the routes in the list.

Tagging Static Routes for Import Filtering

You can now control precisely which static routes you import into BGP by tagging routes with a number, creating a route map to match the tags, and applying the route map when importing a range of static routes.

Handling Spikes in System Memory Use

While BGP is running, other software modules may cause memory use to spike for brief periods of time. BGP now backs off and delays its processing until memory is more abundant. You can customise the backoff thresholds and time.

Stopping BGP from Overloading System Memory

If BGP uses an excessive share of total system memory, the switch or router now automatically disables BGP. The limit is customisable, and set to 95% of the total memory by default.

MD5 Authentication for BGP

Each BGP peer connection can now be authenticated with MD5, so you can be sure of the source and validity of BGP messages and the routing updates they contain.

Route Flap Damping

Under some network conditions, BGP generates an excessive rate of update messages due to "route flapping", in which some routes frequently oscillate between being reachable and unreachable. BGP route flap damping, as defined in RFC 2439, limits the impact and visibility of route flapping to a router or switch's BGP peers.

Fast Fallover

By default, when the interface that supports an EBGP peer session goes down, the corresponding peer session is not reset until that session's hold timer expires.

Fast fallover is an option that you can enable for individual peers, that resets the session as soon as the router or switch's interface to the peer goes down. It provides fast failover in case of link failures, because the router or switch withdraws paths as soon as the link goes down, rather than waiting for up to three minutes to propagate the change.

Route Reflection

Route Reflection improves the scalability of the AS, by giving specific IBGP peers the authority to advertise IBGP-learned routes to a predefined subset of their IBGP peers.

Stripping Private AS Numbers from Update Messages

Private AS numbers are not globally unique, so they should not be leaked to global BGP routing tables, in which context they become ambiguous. To prevent private AS numbers from crossing administrative boundaries, the router or switch now supports the stripping of private AS numbers from the AS Path attributes of outgoing update messages. This enhancement is disabled by default.

New Community Number Format

This enhancement enables the administration of BGP community numbers in the correct format:

aa:*xx*

where

- *aa* is the AS number, in the range 0 to 65534
- *xx* is a value specified by the ASN administrator, in the range 0 to 65534

The old format is also still valid, and is of the form YYYYY, calculated using the formula:

```
AS number x 65536 + community value
```

You can use the new format in the:

- **include** and **exclude** parameters of the command **add ip communitylist**
- **community** parameter of the commands **add ip routemap** and **set ip routemap**
- **community** parameter of the command **show bgp route**

By default, community numbers are displayed using the new format. To display the original format instead, specify the optional parameter **oldcommunityformat** in the commands:

```
show ip routemap oldcommunityformat
show ip communitylist oldcommunityformat
```

OSPF Not-So-Stubby Area Option (NSSA)

An NSSA is an optional type of Open Shortest Path First (OSPF) area. NSSAs are similar to the existing OSPF stub area configuration option but have the additional capability of importing AS external routes in a limited fashion. NSSAs are described in RFC1587, "The OSPF Not-So-Stubby Area (NSSA) Option".

Modified Commands

Use the highlighted new parameter options in the following existing commands to configure an NSSA. Several **show** commands have also been modified to display NSSA information:

- **show ospf area**
- **show ospf**

  'AS boundary router status" displays NSSA
- **show ospf lsa**

  You can now specify the parameter t**ype=asnssa**

add ospf area

Syntax

```
ADD OSPF AREA={BACKBONE|area-number}
   [AUTHENTICATION={NONE|PASSWORD}] [STUBAREA={ON|OFF|YES|
   NO|NSSA|TRUE|FALSE}] [STUBMETRIC=0..16777215]
   [SUMMARY={SEND|NONE|OFF|NO|FALSE}]
```

The **stubarea** parameter specifies whether or not the router or switch treats the area as a stub area. The value **nssa** specifies that the area is a Not-so-stubby-area (NSSA). External routes can be imported as type-7 advertisements in a NSSA.

The **summary** parameter controls the generation of summary LSAs into stub areas. If **stubarea** is set to **nssa** then the default is **send**, otherwise the default is **none**.

set ospf area

Syntax

```
SET OSPF AREA={BACKBONE|area-number}
   [AUTHENTICATION={NONE|PASSWORD}] [STUBAREA={ON|OFF|YES|
   NO|NSSA|TRUE|FALSE}] [STUBMETRIC=0..16777215]
   [SUMMARY={SEND|NONE|OFF|NO|FALSE}]
```

The **stubarea** parameter specifies whether or not the router or switch treats the area as a stub area. The value **nssa** specifies that the area is a Not-so-stubby-area (NSSA). External routes can be imported as type-7 advertisements in a NSSA.

The **summary** parameter controls the generation of summary LSAs into stub areas. If **stubarea** is set to **nssa** then the default is **send**, otherwise the default is **none**.

set ospf

Syntax

```
SET OSPF [ASEXTERNAL={ON|OFF|NSSA}] [DEFROUTE={ON|OFF|
   TRUE| FALSE|YES|NO} [TYPE={1|2}] [METRIC=0..16777215]]
   [DYNINTERFACE={STUB|ASEXTERNAL|NONE|NO|OFF|FALSE}]
   [RIP={OFF|EXPORT|IMPORT|BOTH}] [ROUTERID=ipadd]
   [PTPSTUB={ON|OFF|YES|NO|TRUE|FALSE}]
   [STATICEXPORT=(YES|NO)]
```

The **asexternal** parameter specifies whether or not the router or switch will act as an Autonomous System boundary router and how routes are imported. A router or switch is said to be an Autonomous System (AS) boundary router if it has some interfaces in the OSPF AS and some interfaces that are not in the AS. Typically the router or switch will have some "external" routes in its routing table associated with the interfaces that are not in the AS. If **asexternal** is set to **on** these external routes will be advertised into the AS as type-5 LSAs for non-NSSAs and as type 7 LSAs for NSSAs. If **asexternal** is set to **nssa**, external routes will only be added to NSSAs as type 7 LSAs, which will be translated if appropriate to a type 5 LSA at an NSSA ABR. **asexternal** should be set to **nssa** if this router or switch only uses NSSAs. If **asexternal** is set to **off** these external routes will not be advertised into the AS. The default is **off**.

OSPF Auto Cost Calculation

This enhancement allows OSPF interfaces to automatically set the OSPF metric of an IP interface on the basis of the bandwidth of the interface, instead of the system administrator manually setting the OSPF metric. Automatic setting takes into account that the speed of an interface can change over time, when ports change link state or change speed via auto negotiation or manual setting. If metrics are manually set, some interfaces are preferred when they should not be because the network configuration dynamically changes.

Note that interface speed used in the cost calculation is the average interface speed. For example, if the interface is a VLAN with two ports up, and one port has a speed of 10 and the other a speed of 100, then the metric will be 18.

To configure auto cost calculation:

1. Do not set the OSPF metric manually in the **add ip interface** command. If you have, remove the manual setting, using the command:

   ```
   set ip interface=int ospfmetric=default
   ```

   The **ospfmetric** parameter specifies the cost of crossing the logical interface, for OSPF. If **default** is specified the interface is restored to the default metric value. The setting of the OSPF metric to a value other than **default** provides a mechanism to provide a metric for an interface that is preferred over the OSPF automatic metric setting (if enabled via **set ospf autocost=on**).

2. Set **autocost** to on and change the reference bandwidth if necessary, using the command:

   ```
   set ospf autocost=on [refbandwidth=10..10000]
   ```

   The **autocost** parameter specifies whether or not the router or switch will assign OSPF interface metrics based on the available interface bandwidth. If an OSPF metric has been manually assigned using the **add ip interface ospfmetric**=x, the manual metric setting will take priority over an automatic metric setting. The default is **off**.

   The **refbandwidth** parameter specifies the reference bandwidth in megabits per second used for calculating the OSPF metric. The cost is calculated as **refbandwidth** / Interface Bandwidth. Using the default settings, the automatic cost calculation will result in an OSPF metric of 10 for a fast Ethernet (100M) interface. The **autocost** parameter must be set to **on** for the parameter **refbandwidth** to take effect. The default is 1000.

3. To check the settings, use the command:

   ```
   show ospf
   ```

Authenticating OSPF

You can authenticate OSPF packets as described in Appendix D of RFC 2328. See this RFC for a detailed description of how OSPF packets are authenticated.

An authentication type can be chosen for each interface. RFC 1583 states that authentication is configured per area, but RFC 2328 states that authentication is configured per interface. This implementation of OSPF authentication represents a compromise for these two solutions. An authentication type is set up per area to act as a default for all interfaces in the area. The default setting for interfaces is to use the area default, but each interface can be individually set to any authentication method.

There are two ways to authenticate an OSPF packet:

- simple password authentication
- cryptographic authentication with MD5

Password Authentication

Password authentication can be configured for OSPF areas to authenticate incoming packets. The password can be up to 8 characters long, and is configured for each interface.

To configure an OSPF area with password authentication, use the command:

```
add ospf area={backbone|area-number} authentication=password
```

The password itself is configured on a per-interface basis with the **add ospf interface** and **set ospf interface** commands.

To configure an OSPF interface with password authentication, use the command:

```
add ospf interface=interface authentication=password
   password=password
```

Valid characters for the password are any printable character. If the password contains spaces it must be enclosed in double quotes

For password authentication to succeed, you need to configure all interfaces in the same physical network with the same password.

Cryptographic Authentication

An MD5 digest can be appended to the OSPF packet for authentication. The digest is based on the contents of the packet and a shared secret key. The key that you configure must be the same for all interfaces sharing the same physical network. MD5 keys are defined for each interface, and can be up to 16 characters long. The key is case-sensitive, and valid characters are letters and digits only. If authentication is set to MD5 and no key is configured, a default key is created that has an ID of 0 and no key.

For MD5 authentication to succeed, first configure the OSPF area, then the interface, and then add the MD5 key to the interface. You can use the **set ospf area** and **set ospf interface** commands to modify the authentication type.

To configure an OSPF area with MD5 authentication, use the command:

```
add ospf area={backbone|area-number} authentication=md5
```

To configure an OSPF interface with MD5 authentication, use the command:

```
add ospf interface=interface authentication=md5
```

To add an MD5 key that will be used for interface authentication, use the command:

```
add ospf md5key=key id=1...255 interface=interface
```

Normally, only one MD5 key is added at a time for any given OSPF interface. We recommend changing the MD5 key every month, and periodically deleting old and inactive keys. When changing MD5 keys, add the new key on all routers in the network. While the keys are being added, routers will send duplicate messages using both keys. The old key becomes inactive when all interfaces transmit packets using the new key. The packets will not be duplicated using the inactive key. The output of the **show ospf md5key** command shows whether a key is active or inactive.

Deleting MD5 Keys

You can delete a MD5 key when it has become inactive or when it is being used. You may want to delete a key that is currently active if an illicit router is using the key. To delete an active key immediately, specify the **force** parameter in the **delete ospf md5key** command.

Caution: Force deleting an active MD5 key may lead to partial network failure if succeeding keys are not configured on all interfaces in the physical network.

Before deleting a key, configure a succeeding key on all interfaces in the physical network. You can delete the previously active key without using the FORCE parameter. The new key can take over authentication duties when the active key is deleted.

New Commands

add ospf md5key

Syntax `ADD OSPF MD5KEY=key ID=1...255 INTERFACE=interface`

where:

- *key* is a character string, 1-16 characters in length and case sensitive. Valid characters are letters and digits only: a-z, A-Z, 0-9.
- *interface* is a valid interface name.

Description This command adds an MD5 key for use when authenticating OSPF packets on a particular interface. For MD5 authentication to succeed, all key and ID values must be identical on all interfaces in the same physical network.

The **md5key** parameter specifies the 1-16 character key used for MD5 authentication. The key is entered as alphanumeric characters and is case

sensitive. The more characters in the key, the greater the security it offers. We recommend that the MD5 key is at least 9 characters long and is changed every month.

The **id** parameter specifies the identification number for this key. The ID is used in the authentication of packets to identify to the remote device which key is being used in this packet.

The **interface** parameter specifies the OSPF interface with which this key is associated. Each interface has its own set of keys, and keys must be identified by interface as well as key ID. The interface can be any OSPF interface for which MD5 authentication is required.

Example To add an MD5 authentication key called "mj48dhw05" with an ID of 3 to interface vlan1, use the command:

```
add ospf md5key=mj48dhw05 id=3 interface=vlan1
```

Related Commands

add ospf interface
delete ospf md5key
show ospf md5key

delete ospf md5key

Syntax `DELETE OSPF MD5KEY ID=1..255 INTERFACE=`*`interface`* `[FORCE]`

where:

- *interface* is a valid interface name

Description This command deletes an MD5 key used for authenticating OSPF packets on a particular interface. If the key is being used when the command is executed, the command will fail unless the **force** parameter is also specified. Before deleting a key, configure a succeeding key on all interfaces in the physical network. You can delete the previously active key without using the **force** parameter. The new key can take over authentication duties when the active key is deleted. The output of the **show ospf md5key** command shows whether a key is active or inactive.

The **id** parameter specifies the identification number of the key that will be deleted.

The **interface** parameter specifies the OSPF interface with which this key is associated. Each interface has its own set of keys, and keys must be identified by interface as well as key ID.

The **force** parameter specifies that the MD5 key should be deleted even when it is being used. If an illicit router is using the key, using the FORCE parameter will ensure that the key is deleted.

Caution: Force deleting an active MD5 key may lead to partial network failure if succeeding keys are not configured on all interfaces in the physical network.

Example To delete a key with the ID 35 that was used last week and should be replaced, use the command:

```
delete ospf md5key id=35 interface=vlan1
```

Related Commands **add ospf md5key**
show ospf md5key

show ospf md5key

Syntax `SHOW OSPF MD5KEY [INTERFACE=`*`interface`*`]`

where:

- *interface* is a valid interface name

Description This command displays information about the OSPF MD5 keys for all interfaces, or for a specific interface (Figure 16, Figure 17, Table 16). Each interface has its own set of keys.

The **interface** parameter specifies the OSPF interface for the MD5 keys to be displayed.

Figure 16: Example output from the **show ospf md5key** command

```
OSPF MD5 keys
-------------------------------------------
Interface   ID    Key               Active
-------------------------------------------
vlan1       1     O3jf87Pls         No
            2     xm39s2F28         Yes

vlan2       3     ba2958d2x         Yes
-------------------------------------------
```

Figure 17: Example output from the **show ospf md5key interface** command

```
OSPF MD5 keys
-------------------------------------------
Interface   ID    Key               Active
-------------------------------------------
vlan1       1     O3jf87Pls         No
            2     xm39s2F28         Yes
-------------------------------------------
```

Table 16: Parameters in the output of the **show ospf md5key** command

Parameter	Meaning
Interface	The OSPF interface to which the keys belong
ID	The key's identification number
Key	The MD5 key
Active	Whether or not the key is currently being used to authenticate packets being received from one or more neighbours.

Example To show the MD5 keys for the vlan1 interface, use the command:

```
show ospf md5key interface=vlan1
```

Related Commands **add ospf md5key**
delete ospf md5key

Importing BGP Routes into OSPF

With this enhancement you can import routes from BGP into OSPF. OSPF will then redistribute these routes. This enhancement adds three parameters to the **set ospf** command, and modifies the output of the **show ospf** command. The new parameters are **bgpimport**, **bgpfilter** and **bgplimit**.

BGP can learn thousands of routes, so it's important to consider the network impact of importing these routes. Routing devices in the OSPF domain may become overloaded if they store too many routes. You can prevent this by limiting the number of routes that will be imported.

Caution: Do not enable the importing of BGP routes into OSPF unless you are sure about the consequences for the OSPF domain.

Enabling BGP Route Import

To enable importing BGP routes into OSPF, use the command:

```
set ospf bgpimport=on
```

Limiting the Number of Routes

There are two ways to limit the number of BGP routes imported into OSPF. One way is to specify a maximum number of routes with the command:

```
set ospf bgplimit=1...300
```

When the limit is reached, the importing of routes will stop until existing routes are removed. Because they are BGP routes, actions of BGP control when the routes disappear.

The other way to limit the imported routes is to configure a routing filter. This filter is used in conjunction with the **bgpfilter** parameter in the **set ospf** command to control the passing of routing information in and out of the device. To configure a filter, use the **add ip filter** command:

```
add ip filter=filter-number {action=include|exclude}
   source=ipadd [smask=ipadd] [entry=entry-number]
```

Use this filter to limit imported BGP routes with the command:

```
set ospf bgpfilter=300...399
```

where the filter number is the previously configured filter.

Take care when configuring the IP filter. If the number of imported routes reaches the **bgplimit** parameter, you may not have imported all the routes specified with the **bgpfilter** parameter.

Advertising Desired Routes

The order in which routes are added is arbitrary. This means that to have desired BGP routes advertised by OSPF, you must take care setting the **entry** number for the route filter with the **add ip route** command. Assign a low entry number to a filter used to import preferred BGP routes. Alternatively, set the

bgplimit parameter above the total number of routes that BGP will ever add to the routing table.

Configuration Example

This example supposes that you want to import the route 192.168.72.0 into the OSPF routing domain, but no other routes. This route is received on the gateway router as a BGP route. The following steps show the sequence of commands to use in this scenario.

1. Set up the IP filter:

   ```
   add ip filter=300 source=192.168.72.0 smask=255.255.255.255
      action=include
   ```

2. Set up OSPF BGP import parameters:

   ```
   set ospf bgpimport=on bgpfilter=300 bgplimit=1
   ```

3. Check that BGP has added the route to the IP route table:

   ```
   show ip route=192.168.72.0
   ```

   The route should be visible in the output of the command.

4. Check that OSPF has imported the route:

   ```
   show ospf lsa=192.168.72.0
   ```

   The output should show that there is an AS external LSA with this ID.

DVMRP Interoperability

DVMRP now operates successfully with another vendor's equipment.

IGMP Snooping All-groups

This enhancement allows you to prevent a port or ports from acting as an all-groups entry.

Sometimes the device cannot differentiate between certain multicast addresses and permanent host groups at Layer 2. For example, this happens with the addresses 239.0.0.2 and 224.0.0.2 where 224.0.0.2 is the all-routers multicast group. If the device receives an IGMP report for the 239.0.0.2 address, which has a MAC address of 01-00-5e-00-00-02, the device will create an all-groups entry in the MARL. All further multicast groups will be added to this port, so multicast traffic will be forwarded out the port.

By preventing a port or ports from receiving an all-groups entry, you can limit the number of router ports on the device, and therefore the volume of multicast traffic sent over the device's ports. Once disabled with the **disable ip igmp allgroup** command, the port will no longer create MARL entries when the device receives an IGMP report, query, or multicast data over any other port. For example, if port 9 has been disabled as an all-groups port, an all-groups entry will be created for port 9. This will happen when the port receives packets that will create an IGMP router port, such as reserved multicast groups and IGMP queries. However, a subsequent IGMP report received over port 7 will have an entry made for port 7 only. The IGMP group received on port 7 will not be added to port 9.

The all-groups disabled ports can be viewed in the output of the **show ip igmp** and **show igmpsnooping** commands.

New Commands

enable ip igmp allgroup

Syntax `ENAble IP IGMP ALLGroup=[`*`port-list`*`|ALL]`

where:

- *port-list* is a port number, a range of port numbers (specified as n-m), or a comma separated list of port numbers and/or ranges. Port numbers start at 1 and end at m, where m is the highest numbered Ethernet switch port, including uplink ports.

Description This command enables the specified port or ports to act as a router port. All ports are allowed to be a router port by default, so this command is used to re-enable a port as a router port if it has previously been disabled with the **disable ip igmp allgroup** command.

Example To enable ports 1, 5, and 7 to act as an all-group entry, use the command:

```
ena ip igmp allg=1,5,7
```

Related Commands **disable ip igmp allgroup**
show ip igmp

disable ip igmp allgroup

Syntax `DISable IP IGMP ALLGroup=[`*`port-list`*`|ALL]`

where:

- *port-list* is a port number, a range of port numbers (specified as n-m), or a comma separated list of port numbers and/or ranges. Port numbers start at 1 and end at m, where m is the highest numbered Ethernet switch port, including uplink ports.

Description This command disables the specified port or ports from acting as a router port. Once disabled, the port will no longer receive MARL entries when the device receives an IGMP report, query, or multicast data over any other port.

Example To prevent ports 1, 5, and 7 from acting as an all-group entry, use the command:

```
dis ip igmp allg=1,5,7
```

Related Commands **enable ip igmp allgroup**

Modified Command

show ip igmp

Syntax `SHOW IP IGMP`

Description This command displays information about IGMP, and multicast group membership for each IP interface.

Table 17: New parameter in the output of the **show ip igmp** command

Parameter	Meaning
Disabled All-groups ports	A list of ports that are not allowed to act as all-groups ports

Figure 18: New parameter in the output of the **show ip igmp** command

```
IGMP Protocol
--------------------------------------------------------------------------------
Status ........................... Enabled
Default Query Interval ........... 125 secs
Default Timeout Interval ......... 270 secs
Disabled All-groups ports ........ 1,5,7

Interface Name .......... vlan2              (DR)
IGMP Proxy .............. Off
Group List ..............

  Group. 238.0.1.2          Last Adv. 172.50.2.1         Refresh time 34 secs
  Ports  11,23

  Group. 224.1.1.2          Last Adv. 172.50.2.1         Refresh time 130 secs
  Ports  11,23

  All Groups                Last Adv. 172.50.1.1         Refresh time 45 secs
  Ports  11,23


Interface Name .......... vlan4              (DR)
IGMP Proxy .............. Off
Group List ..............

  No group memberships.

--------------------------------------------------------------------------------
```

Static IGMP

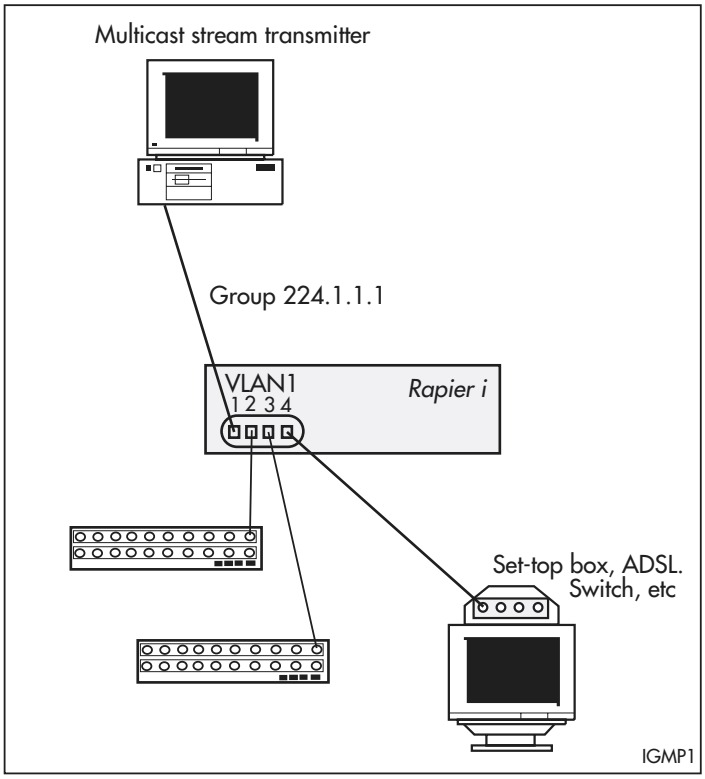
Static IGMP configures the switch to forward multicast data over specified interfaces and ports. It is an alternative to dynamic IGMP, and is useful for network segments that either have no multicast group members or have hosts that are unable to report group membership with IGMP. A dynamic IGMP configuration will not send multicast traffic to such network segments.

Figure 19 illustrates a switch forwarding the multicast stream to a set-top box after a user specifies that group 224.1.1.1 multicast data should be forwarded out of port 4 of VLAN1.

Unlike conventional IGMP membership, this user-specified *static membership* never times out.

You can also filter some IGMP debug messages by source IP address and group destination address.

Figure 19: Forwarding multicast data over a specified interface and port.



To configure a static IGMP association:

1. Enable IGMP on the switch, using the command:

   ```
   ENABLE IP IGMP
   ```

2. Enable IGMP on the required interface, using the command:

   ```
   ENABLE IP IGMP INTERFACE=interface
   ```

3. Create the static IGMP association, using the command:

   ```
   CREATE IP IGMP DESTINATION=ipaddress INTERFACE=interface
      [PORT={ALL|port-list}]
   ```

   The multicast data for the group specified by the **destination** parameter will be forwarded over the ports specified by the **port** parameter. Any of the four octets of the IP address may be replaced by an asterisk (*) to enable wildcard matches. If the **port** parameter is not entered, the association will default to all ports belonging to the interface.

To display information about the static IGMP association, use the command:

```
SHOW IP IGMP [COUNTER] [INTERFACE=interface]
   [DESTINATION=ipaddress]
```

To add more ports to an association, use the command:

```
ADD IP IGMP DESTINATION=ipaddress INTERFACE=interface
   PORT={ALL|port-list}
```

Unlike dynamic IGMP group membership information, static IGMP associations never time out. If the network configuration changes, they must be manually modified. To delete ports from an association, use the command:

```
DELETE IP IGMP DESTINATION=ipaddress INTERFACE=interface
   PORT={ALL|port-list}
```

To remove an association from the switch, use the command:

```
DESTROY IP IGMP DESTINATION=ipaddress INTERFACE=interface
```

To enable or disable IGMP debugging of destination and source IP addresses, use the commands:

```
ENABLE IP IGMP DEBUG [DESTINATION={ALL|ipaddress}]
   [SOURCEIPADDRESS={ALL|ipaddress2}]

DISABLE IP IGMP DEBUG
```

where:

- *ipaddress* is an IGMP group destination address.
- *ipaddress2* is the IP address of a host that responds to IGMP queries.

Debugging is disabled by default. To display which debugging options are set, use the command:

```
SHOW IP IGMP DEBUG
```

Specifying Router Multicast Addresses for IGMP Snooping

You can now specify the mode of operation when IGMP Snooping is enabled with the command:

```
set igmpsnooping routermode=[all|default|ip|multicastrouter|
    none]
```

If **all** is specified, all reserved multicast addresses (i.e. 224.0.0.1 to 224.0.0.255) are treated as router multicast addresses.

If **default** is specified, the following addresses are treated as router multicast addresses:

- IGMP Query, 224.0.0.1
- All routers on this subnet, 224.0.0.2
- DVMRP Routers, 224.0.0.4
- OSPFIGP all routers, 224.0.0.5
- OSPFIGP designated routers, 224.0.0.6
- RIP2 routers, 224.0.0.9
- All PIM routers, 224.0.0.13
- All CBT routers, 224.0.0.15

If **ip** is specified, you specify addresses treated as router multicast addresses using the **add igmpsnooping routeraddress** and the **delete igmpsnooping routeraddress** commands. When in this mode, your router or switch retains previous addresses that have already been specified.

If **multicastrouter** is specified, the following addresses are treated as router multicast addresses:

- DVMRP Routers, 224.0.0.4
- All PIM routers, 224.0.0.13

If **none** is specified, the router or switch does not create router ports at all.

To add and delete reserved IP multicast addresses to and from the list of router multicast addresses specified by the s**et igmpsnooping routermode** command when the **ip** parameter is selected, use the commands:

```
add igmpsnooping routeraddress
```

```
delete igmpsnooping routeraddress
```

The IP addresses specified must be from 224.0.0.1 to 224.0.0.255.

To display information about the current list of configured IP multicast router addresses configured on your router or switch, use the command:

```
show igmpsnooping routeraddress
```

Logging and SNMP Traps for PIM-SM

PIM-SM can be configured to produce log messages in response to status changes and errors, and SNMP traps. This feature does not apply to PIM-DM.

Status log messages

Events that trigger a status-change log message are:

- PIM interface is disabled
- PIM interface is enabled
- PIM neighbour adjacency has timed out
- PIM neighbour generation ID has changed
- PIM neighbour has changed port
- PIM RP has changed
- PIM DR has changed
- PIM BSR has changed

Error log messages

Errors that trigger a log message are:

- Invalid PIM packet
- Invalid destination address
- Fragmentation reassembly
- Packet too short
- Bad group address encoding
- Bad source address encoding
- Missing option
- Internal error
- Receive packet - a range of errors that mean the packet was received but cannot be forwarded.

SNMP traps

The following traps are sent:

- PimInterfaceUpTrap - generated when a PIM interfaces comes up and is active
- PimInterfaceDownTrap - generated when a PIM interfaces goes down and is in-active
- PimNeighbourLossTrap - generated when a known PIM neighbour has loss adjacency or has timed-out. This trap is part of the experimental PIM MIBs group
- PimNeighbourAddedTrap - generated when a PIM neighbour is added
- PimNeighbourDeletedTrap - generated when a PIM neighbour is deleted
- PimErrorTrap - generated when any one of the PIM error counters is incremented or when a log message of subtype LOG_STY_PIM_ERROR is generated (see list of errors above)

To specify the type of log messages and SNMP traps that the router or switch generates, use the command:

```
set pim log={none|status|error|all}
   [trap={none|status|error|all}]
```

To display the specified options, use the command:

```
show pim debug
```

PIM4 Hash Mask Length Support

This enhancement enables you to specify the number of bits of the group number that are significant when selecting a rendezvous point (RP) candidate, if this router or switch becomes the BSR. Prior to this enhancement, a single RP was selected for all groups that match its prefix. With this enhancement, the RP selection will be spread across candidate RPs based on the hash mask length chosen.

Note that previous release versions did not correctly support the PIM hash mask length option. As a result, the RP selection calculation differs between old and new release versions. If a network contains routers or switches running a mixture of versions, this leads to incorrect forwarding behaviour. To avoid this issue, either ensure that all devices on the network correctly support the hash mask length option (recommended), or ensure that the following **both** hold:

- The hash mask length option on all BSR candidates is configured to 4 bits. This implies that all BSR candidates must be running a new release.
- All RP candidates use a common prefix of 224.0.0.0/240.0.0.0. This will have the side effect of collapsing all groups to use a single PIM RP.

Modified Commands

To configure the hash mask length, use the highlighted new parameters in the following existing commands.

add pim bsrcandidate

Syntax `ADD PIM BSRCANDIDATE` **`[HASHMASKLENGTH=0..32]`**
`[`*`other-options`*`]`

The **hashmasklength** parameter specifies the number of bits of the group number to use when selecting a rendezvous point (RP) candidate if this router or switch becomes the BSR. A higher number increases the spread of groups across RPs. The default is 30.

Examples To add the router or switch as a Bootstrap Router Candidate to a PIM domain, with a preference of 10 to become the bootstrap router in the domain and a hash mask length of 0, use the command:

```
add pim bsrcandidate preference=10 hashmasklength=0
```

set pim bsrcandidate

Syntax `SET PIM BSRCANDIDATE` **`[HASHMASKLENGTH=0..32]`**
`[`*`other-options`*`]`

The **hashmasklength** parameter specifies the number of bits of the group number to use when selecting a rendezvous point (RP) candidate if this router or switch becomes the BSR. A higher number increases the spread of groups across RPs. The default is 30.

Specifying the MLD Query Version

This enhancement changes the parameter that specifies the MLD Query version on an interface. For software versions since 2.6.4, including this software version, the **v1compatible** parameter for the command has been replaced with the **queryversion** parameter. The new syntax is:

```
enable ipv6 mld interface=interface [queryversion={1|2}]
set ipv6 mld interface=interface [queryversion={1|2}]
```

The **queryversion** parameter specifies the version of MLD Query to use on the interface. It is a more accurate way to specify interoperability between MLDv2 and MLDv1. The default is 2.

To avoid unnecessary error messages, we recommend that users replace **v1compatible** with **queryversion** along with their related values in scripts currently being used. For more information about Multicast Listener Discovery, see the *IPv6 Multicasting* chapter of the *Software Reference*.

ICMP Router Discovery Advertisements

This release supports all of *RFC 1256, ICMP Router Discovery Messages, 1991* as it applies to routers. If this feature is configured, the router or switch sends router advertisements periodically and in response to router solicitations. It does not support the Host Specification section of this RFC.

Before an IP host can send an IP packet, it has to know the IP address of a neighbouring router that can forward it to its destination. ICMP Router Discovery messages allow routers to automatically advertise themselves to hosts. Other methods either require someone to manually keep these addresses up to date, or require DHCP to send the router address, or require the hosts to be able to eavesdrop on whatever routing protocol messages are being used on the LAN.

Router Discovery Process

For a summary of the processes that occur when Router Discovery advertisements are enabled for interfaces on the router or switch see Table 18.

Table 18: Router Discovery Process

When ...	Then ...
Router Discovery advertising starts on a router or switch interface because: - the router or switch starts up, or - advertisements are enabled on the router or switch or on an interface	the router or switch multicasts a router advertisement and continues to multicast them periodically until router advertising is disabled.
a host starts up	the host may send a router solicitation message.
the router or switch receives a router solicitation	the router or switch multicasts an early router advertisement on the multicast interface on which it received the router solicitation.
a host receives a router advertisement	the host stores the IP address and preference level for the advertisement lifetime.
the lifetime of all existing router advertisements on a host expires	the host sends a router solicitation.
a host does not receive a router advertisement after sending a small number of router solicitations	the host waits for the next unsolicited router advertisement
a host needs a default router address	the host uses the IP address of the router or L3 switch with the highest preference level.
Router Discovery advertising is deleted from the physical interface (**delete ip advertise** command), or the logical interface has **advertise** set to **no** (**set ip interface** command)	the router or switch multicasts a router advertisement with the IP address(es) that stopped advertising, and a lifetime of zero (0). It continues to periodically multicast router advertisements for other interfaces.
the router or switch receives a router advertisement from another router	the router or switch does nothing but silently discards the message.

Router Advertisement Messages

A *router advertisement* is an ICMP (type 10) message containing:

- In the destination address field of the IP header, the interface's configured advertisement address, either 224.0.0.1 (ALL) or 255.255.255.255 (LIMITED).
- In the lifetime field, the interface's configured advertisement lifetime.
- In the Router Address and Preference Level fields, the addresses and preference levels of all the logical interfaces that are set to advertise.

Router Solicitation Messages

A *router solicitation* is an ICMP (type 10) message containing:

- Source Address: an IP address belonging to the interface from which the message is sent
- Destination Address: the configured Solicitation Address, and
- Time-to-Live: 1 if the Destination Address is an IP multicast address; at least 1 otherwise.

Router Advertisement Interval

The router advertisement *interval* is the time between router advertisements. For the first few advertisements sent from an interface (up to 3), the router or switch sends the router advertisements at intervals of at most 16 seconds. After these initial transmissions, it sends router advertisements at random intervals between the minimum and maximum intervals that the user configures, to reduce the probability of synchronization with the advertisements from other routers on the same link. By default the minimum is 450 seconds (7.5 minutes), and the maximum is 600 seconds (10 minutes).

Preference Level

The *preference level* is the preference of the advertised address as a default router address relative to other router addresses on the same subnet. By default, all routers and layer 3 switches have the same preference level, zero (0). While it is entered as a decimal in the range `-2147483648..2147483647`, it is encoded in router advertisements as a twos-complement hex integer in the range 0x8000000 to 0x7fffffff. A higher PREFERENCELEVEL is preferred over a lower value.

Lifetime

The *lifetime* of a router advertisement is how long the information in the advertisement is valid. By default, the lifetime of all advertisements is 1800 seconds (30 minutes).

Configuration Example

By default, the router or switch does not send router advertisements.

To configure the router to send router advertisements:

1. Set the physical interface to advertise.

For each physical interface that is to send advertisements, add the interface. In most cases the default advertising parameters will work well, but you can change them if required. By default, the router or switch sends router advertisements every 7.5 to 10 minutes, with a lifetime of 30 minutes. These settings are likely to work well in most situations, and will not cause a large amount of extra traffic, even if there are several routers on the LAN. If you change these settings, keep these proportions:

```
lifetime=3 x maxadvertisementinterval
```

```
minadvertisementinterval=0.75 x maxadvertisementinterval
```

To change these settings, use one of the commands:

```
add ip advertise interface=interface
    [advertisementaddress={all|limited}]
    [maxadvertisementinterval=4..1800]
    [minadvertisementinterval=3..maxadvertisementinterval]
    [lifetime=maxadvertisementinterval..9000]
```

```
set ip advertise interface=interface
    [advertisementaddress={all|limited}]
    [maxadvertisementinterval=4..1800]
    [minadvertisementinterval=3..maxadvertisementinterval]
    [lifetime=maxadvertisementinterval..9000]
```

2. Stop advertising on other logical interfaces.

By default, logical interfaces are set to advertise if their physical interface is set to advertise. If the physical interface has more than one logical interface (IP multihoming), and you only want some of them to advertise, set the other logical interfaces not to advertise by using one of the commands:

```
add ip interface=interface ipaddress={ipadd|dhcp}
   advertise=no [other-parameters...]
```

```
set ip interface=interface advertise=no
    [other-parameters...]
```

3. Set preference levels.

By default, every logical interface has the same preference for becoming a default router (mid range, 0). To give a logical interface a higher preference, increase the **preferencelevel**. To give it a lower preference, decrease this value. If it should never be used as a default router, set it to **notdefault**.

```
add ip interface=interface ipaddress={ipadd|dhcp}
   preferencelevel={-2147483648..2147483647|notdefault}
    [other-parameters...]
```

```
set ip interface=interface
    [preferencelevel={-2147483648..2147483647|notdefault}]
    [other-parameters...]
```

4. Enable advertising.

To enable router advertisements on all configured advertising interfaces, use the command:

```
enable ip advertise
```

5. Check advertise settings.

To check the router advertisement settings, use the command:

```
show ip advertise
```

Adopting the VRRP IP Address

Benefits of VRRP IP Address Adoption

The VRRP master router can *adopt* the IP address of the virtual router (VR), and respond to the following packets destined for the VR IP address, even if it does not own this IP address on any of its interfaces:

- ICMP echo requests (pings)
- Telnet and SSH connection requests
- HTTP and SSL GUI management requests
- SNMP requests, and
- DNS relay requests

VRRP IP Address Adoption allows continuous accessibility of the VR IP address even as the VR master changes. Using this feature:

- You can easily tell whether the VR is functioning, by pinging the single VR IP address.
- You can easily monitor the performance of the VR, regardless of which participating router is acting as master.
- DNS relay can continue functioning via the same IP address at all times.

Risks of VRRP IP Address Adoption

When VRRP IP Address Adoption is used, the master router accepts packets destined for the virtual router, even though it may not own this IP address. This does not conform to RFC 2338. Because the same IP address refers to different devices at different times, there is a risk of confusion arising. This risk can be reduced by a suitable network management policy.

Recommendations

Before using VR IP address adoption, consider the following guidelines to avoid confusion:

- Ensure that the VR has an IP address that is different from the interface IP addresses of any of the individual routers in the VR.
- Ensure that all routers in the virtual router use VRRP IP Address Adoption (or that none do).
- Use the VRRP IP address to monitor the VR master. Be aware that this does not give information about one particular participating router, but about the current VR master, whichever participating router is acting as the master at the time.
- When changing the configuration of the participating routers using Telnet, GUI or SNMP, configure each device individually by pointing to their individual IP addresses.
- When changing the configuration of the participating routers, do not use the VR IP address. Only one device, the VR master, is responding to this IP address, and you may not know which device it is.

Configuration of VR IP Address Adoption

To configure VRRP IP Address Adoption, use the new parameter, **adoptvrip**, that has been added to the **create vrrp** and **set vrrp** commands:

```
create vrrp=vr-identifier over=physical-interface
   ipaddress=ipadd [adoptvrip={on|off}] [other-parameters...]

set vrrp=vr-identifier [adoptvrip={on|off}]
   [other-parameters]
```

The **adoptvrip** parameter specifies that when the router or switch is acting as the VRRP master it should respond to requests directed at any IP address that it is backing up, even if it does not own that address. If it does not own the address the access requests that the router or switch will permit are limited to: ICMP echo requests (pings), Telnet, SSH, HTTP and SSL GUI, SNMP and DNS relay. All other types of access to the address will be ignored. The default is OFF.

Caution: If you set **adoptvrip** to on, give the VR an IP address that is different from the interface IP addresses of any of the individual routers in the VR, and only use the VR IP address to monitor the VR, not to configure any of its participating routers. Otherwise, you risk confusion when you monitor or configure individual routers. See *Synchronising Time Across Stacks* in the Stacking Chapter for more about risks and recommendations.

Caution: Configure all the routers or switches in a virtual router with the same values for the VRRP virtual router identifier, IP address, adopt VR IP address mode, advertisement interval, preempt mode, authentication type and password. Inconsistent configuration will cause advertisement packets to be rejected and the virtual router will not perform properly.

To display the value of the new parameter, use the **show vrrp** command.

Table 19: New parameter displayed in the output of the **show vrrp** command

Parameter	Meaning
Adopt VR IP Address(es)	Whether or not the router or switch should respond to ICMP echo, Telnet, GUI, SNMP and DNS relay service requests targeted at the VR IP address(es) associated with the virtual router, even if it does not own those address(es).

Values for IPv6 Router Advertisement *AdvRetransTimer*

The value you enter for the AdvRetrans timer is now rounded up to the nearest 100 milliseconds (for example, 301 becomes 400). The AdvRetrans timer is the interval between repeats of each Router Advertisement message sent by the router or switch, and is specified by using the **retrans** parameter in the command:

```
set ipv6 nd interface=interface retrans=0..4294967295
   [other-parameters]
```

The default is 0, which indicates that this timer is not specified.

Defence Against SYN Flood Attacks

The firewall maintains a queue of unestablished TCP (Transmission Control Protocol) connections. The management of this queue has been enhanced to protect against SYN Flood Attacks. SYN Flood Attacks send TCP requests faster than a machine can process them.

This enhancement supports an automated mechanism for aging unestablished TCP sessions. Depending on the number of suspicious hosts detected, any, or if necessary, all unestablished TCP sessions are aggressively aged, until the number of suspicious hosts decreases again to an acceptable level.

The output of the **show firewall** command now includes the parameter TCP Handshake Timeout Mode, which will specify one of the following values:

- **Normal**
  All unestablished TCP sessions are aged normally
- **Semi Aggressive**
  Any unestablished TCP sessions that involve suspicious hosts are being aged aggressively
- **Aggressive**
  All unestablished TCP sessions are being aged aggressively.

Debugging and Displaying Firewall ARP Requests

A new option has been added to the **disable firewall policy** and **enable firewall policy** commands. **Arp** can now be specified as a **debug** parameter. This option enables or disables the display of all ARP requests that have passed through the firewall.

To specify **arp**, use the commands:

```
enable firewall policy=name [debug={all|arp|http|packet|
   pkt|process|proxy|smtp}] [other-parameters...]
```

```
disable firewall policy=name [debug={all|arp|http|packet|
   pkt|process|proxy|smtp}] [other-parameters...]
```

A router or switch that is also acting as a NAT device will now respond to ARP requests for any of its global IP addresses.

A new command, **show firewall arp**, displays information about IP addresses specified in Firewall NAT configurations for which ARP responses from the router or switch may be required. To display this information, use the command:

```
show firewall arp [policy=name]
```

The **policy** parameter specifies a firewall policy and displays IP addresses for NAT configurations with that policy. If this parameter is not specified, IP addresses are displayed for all policies.

An example output and the parameter descriptions for the **show firewall arp** are shown below.

Figure 20: Example output from the **show firewall arp** command

```
IP                ARP Interfaces  NAT Type    Int        Gbl Int    Rule
(range)           Policy
-------------------------------------------------------------------------------
172.20.8.50       Public          Int based   eth0-0     eth1-0     -
                  Office
172.20.8.57       All Public      Rule        eth0-1     -          1
 -172.20.8.62     LAN
-------------------------------------------------------------------------------
```

Table 20: Parameters in the output of the **show firewall arp** command

Parameter	Meaning
IP (range)	An IP address or range for which the router or switch may be required to send ARP responses.
Policy	The name of the policy whose NAT configuration the IP address (range) belongs to.
ARP Interfaces	Interfaces in the policy on which ARP requests are permitted: Public - ARP requests are permitted on the public interface specified by the Gbl Int parameter All Public - ARP requests are permitted on all of the policy's public interfaces Private - ARP requests are permitted on the private interface specified by the Int parameter All Private - ARP requests are permitted on all of the policy's private interfaces An address in an ARP request must match the subnet of the interface on which the ARP request is received.
NAT Type	The type of NAT configuration associated with the IP address: Int Based - The address (range) was specified by an interface-based NAT configured with the **add firewall policy nat** command Rule - The address (range) was specified by a NAT rule configured by the **add firewall policy rule** command, where the ACTION parameter was specified as NAT
Int	The private interface associated with the NAT configuration. If the NAT Type is Int based, this is the private interface specified by the INTERFACE parameter in the **add firewall policy nat** command. If the NAT Type is Rule, this is the interface to which the rule is attached. If this is a private interface, a dash indicates that the rule is attached to a public interface (see the Gbl Int parameter).
Gbl Int	The public interface associated with the NAT configuration. If the NAT Type is Int based, this is the public interface specified by the GBLINTERFACE parameter in the **add firewall policy nat** command. If the NAT Type is Rule, this is the interface to which the rule is attached. if this is a public interface, a dash indicates that the rule is attached to a private interface (see the Int parameter).
Rule	The number of the rule associated with this entry. When the NAT Type is Int based, no value is displayed.

New IPsec Log Messages

The new log messages *Inbound packet discarded* and *Outbound packet discarded* are now available.

For information about how to configure logging, see the *Logging Facility* chapter in the *Software Reference*.

Inbound packet discarded

Message

```
Inbound ESP packet discarded: src <source-ip-address>
dst <destination-ip-address>
[spi <security-parameter-index>]
[seqNum <sequence-number>] <reasons-for-discard>

Inbound AH packet discarded: src <source-ip-address>
dst <destination-ip-address>
[spi <security-parameter-index>]
[seqNum <sequence-number>] <reasons-for-discard>

Inbound IPCOMP packet discarded: src <source-ip-address>
dst <destination-ip-address>
[cpi <compression-parameter-index>] <reasons-for-discard>
```

Severity	Module	Type	Subtype
4/NOTICE	81/IPSEC	042/IPSC	001/MSG

Explanation This message indicates that during IPSEC processing an inbound ESP, AH or IPCOMP packet was discarded. Detailed information of the packet and reasons for discard are logged.

Recommended Action Investigate the given reasons.

Outbound packet discarded

Message

```
Outbound ESP packet discarded: src <source-ip-address>
dst <destination-ip-address>
[spi <security-parameter-index>]
[seqNum <sequence-number>] <reasons-for-discard>

Outbound AH packet discarded: src <source-ip-address>
dst <destination-ip-address>
[spi <security-parameter-index>]
[seqNum <sequence-number>] <reasons-for-discard>

Outbound IPCOMP packet discarded: src <source-ip-address>
dst <destination-ip-address>
[cpi <compression-parameter-index>] <reasons-for-discard>
```

Severity	Module	Type	Subtype
4/NOTICE	81/IPSEC	042/IPSC	001/MSG

Explanation This message indicates that during IPSEC processing an outbound ESP, AH or IPCOMP packet was discarded. Detailed information of the packet and reasons for discard are logged.

Recommended Action Investigate the given reasons.

ISAKMP Counters and Log Message

New ISAKMP counters have been added to the output of the **show isakmp** commands. New counters are shown in bold. The log message *Retransmission received after completion* has also been added.

Figure 21: New counters in the output of the **show isakmp counters=quick** command

```
Quick Mode Counters:

General Counters:
        Initiator:
    startExchange         0     exchangeGood           0
    hashSaNonceSent       0     hashSent               0
    hashSaNonceReceived   0     hashSaNonceRcvdGood    0
    connectedReceived     1     connectedReceivedGood  1
    natOaSent             0     natOaReceived          0
    hashSaNonceExpKByte   0     connectedExpKByte      0

        Responder:
    startExchange         0     exchangeGood           0
    hashSaNonceSent       0
    hashSaNonceReceived   0     hashSaNonceRcvdGood    0
    hashReceived          0     hashReceivedGood       0
    hashSaNonceExpKByte   0     hashExpKByte           0
    natOaSent             0     natOaReceived          0

 Error Counters:

        Initiator: General errors:
    initHash2Fail         0     initDHGenFail          0
    initStartCBFailed     0     initStartCBNoXchg      0
    initProc1CBFailed     0     initProc1CBNoXchg      0
    initHash4Fail         1

                  Receive Hash SA Nonce message errors:
    invalidPayloadType    0     hashPayMissing         0
    hashUnexpectedLen     0
    saNoMatch             0     saBadLen               0
    saBadDoi              0     saBadSituation         0
    saLenTooShort         0     saPayMissing           0
    saPropInconsistLen    0
    propNoMatch           0     propBadLen             0
    propTranInconsistLen  0     propBadNextPayload     0
    propBadRsv            0     propMultipleTrans      0
    propTooManyProps      0
    tranBadLen            0     tranBadNextPayload     0
    tranBadRsv            0     attrBad                0
    nonceBadLen           0     noncePayMissing        0
    nonceSeenTwice        0     idsSeenTwice           0
    idciBad               0     idcrBad                0
    idcrPayMissing        0     keSeenTwice            0
    badNatOa              0

                  Receive Connected message errors:
    hashPayMissing        0     invalidPayloadType     0
    hashBadLen            0
```

```
     Responder: General errors:
respAcquireNoPolicy  0     respRemotePropNoMatch  0
respHash1Fail        0     respHash3Fail          0
respProc2CBFailed    0     respProc2CBNoXchg      0
respProc3CBFailed    0     respProc3CBNoXchg      0

             Receive Hash SA Nonce message errors:
invalidPayloadType   0     hashPayMissing         0
hashUnexpectedLen    0     noncePayMissing        0
nonceSeenTwice       0     nonceBadLen            0
saPayMissing         0     saPayBad               0
saLenTooLong         0     saLenTooShort          0
saBadDoi             0     saBadSituation         0
saPropInconsistLen   0
propLenTooLong       0     propLenTooShort        0
propBadLen           0     propBadRsv             0
propBadNextPayload   0     propBadNumber          0
propBadProtid        0     propBadSpiSize         0
propNoTransforms     0     propTranInconsistLen   0
tranBadLen           0     tranBadNextPayload     0
tranBadNumber        0     tranBadRsv2            0
tranTransId          0     attrBad                0
idsSeenTwice         0
idciBadType          0     idcrBadType            0
idcrPayMissing       0     keSeenTwice            0
badNatOa             0

          Receive Hash message errors:

hashPayMissing       0     invalidPayloadType     0
hashBadLen           0
```

Table 21: New Parameters in the output of the **show isakmp counter=quick** command.

Parameter	Meaning
General counters for this router or switch as initiator or responder	
connectedReceived	The number of times a Connected message was received in Quick mode.
connectedReceivedGood	The number of times a valid Connected message was received in Quick mode and processed successfully.
connectedExpKByte	The number times a Connected message was received in Quick mode and caused the SA to expire due to reaching its Kbyte limit.
Error counters for initiators and responders	
initHash4Fail	The number of times the fourth hash check of an exchange failed.

New **deletedelay** parameters have been added to the output of the **show isakmp counter=general** and **show isakmp sa** commands.

Figure 22: Example output from the **show isakmp counter=general** command

```
other parameters...

Heartbeat Mode Counters:
  startXchgInitiator     0    startXchgResponder     0
  initXchgComplete       0    respXchgComplete       0
  txMsg                  0    rxMsg                  0
  startXchgInitNotCfgd   0    startXchgRespNotCfgd   0
  rxUnexpectedPayload    0    rxInvalidSeqno         0
  rxHashUnexpectedLen    0    rxNotifyPayInvalid     0
  rxBadHash              0    rxExtraPayloads        0

Delete Delay Counters:
  deleteDelayStarted 18346    deleteDelayNotUsed     2
  deleteDelayPktsRxd     0    deleteDelayRetrySent   0
  deleteDelayConSent     0    deleteDelayConNoSA     0
  deleteDelayTxExceeded  0    deleteDelayPktDiff     0
```

Table 22: New parameters in the output of the **show isakmp counter=general** command

Delete Delay Counters	Counters about ISAMP Delete Delay
deleteDelayStarted	The number of ISAKMP exchanges where a deletedelay period was started.
deleteDelayNotUsed	The number of ISAKMP exchanges where a deletedelay period was not used. This may occur if the deletedelay is zero, if the device did not send the last message in the exchange, or for informational exchanges, where the peer may not send a message in the exchange.
deleteDelayPktsRxd	The number of retransmissions received from ISAKMP peers during the deletedelay periods.
deleteDelayRetrySent	The number of retransmitted messages the device has sent during the deletedelay periods.
deleteDelayConSent	The number of retransmitted informational connected messages the device has sent during the deletedelay periods.
deleteDelayConNoSa	The number of informational connected messages the device failed to retransmit due to lack of a suitable ISAKMP SA.
deleteDelayTxExceeded	The number of deletedelay periods where the retries sent exceeds the ISAKMP policy's msgretrylimit.
deleteDelayPktDiff	The number of retransmissions received that have a different unencrypted length or next payload type compared to the last received packet.

Figure 23: Example output from the **show isakmp sa** command for a specific SA.

```
 SA Id ................................ 1
  Initiator Cookie ..................... e418dba372510e53
  Responder Cookie ..................... 80c30ff4f2cb3f29
  DOI .................................. IPSEC
  Policy name .......................... main
  State ................................ ACTIVE
  Local address ........................ 202.36.163.161
  Remote Address ....................... 202.36.163.201
  Time of establishment ...............
  Commit bit set ....................... FALSE
  Send notifies ........................ TRUE
  Send deletes ......................... FALSE
  Message Retry Limit .................. 5
  Initial Message Retry Timeout (s) ... 20
  Exchange Delete Delay (s) ........... 30
  Do Xauth ............................. FALSE
    Xauth Finished ..................... TRUE
  Expiry Limit (bytes) ................. 1024000
  Soft Expiry Limit (bytes) ............ 896000
  Bytes seen ........................... 304
  Expiry Limit (seconds) ............... 86400
  Soft Expiry Limit (seconds) .......... 75600
  Seconds since creation ............... 2117
  Number of Phase 2 exchanges allowed . 4294967295
  Number of acquires queued ............ 0

...

other output
```

Table 23: New parameter in the output of the **show isakmp sa** command for a specified SA

Parameter	Meaning
Exchange Delete Delay (s)	The delay period, in seconds, between the completion and the deletion of ISAKMP exchanges over this SA.

Figure 24: Example output from the **show isakmp policy** command for a specific policy.

```
ISAKMP Policy
  Name ................................ my_isakmp_policy
  Peer Address ........................ 202.36.163.201
  Phase1 Mode ......................... IDPROT
  Authentication Type ................. PRESHARED
  Extended Authentication ............. NONE
  Extended Authentication Type ........ -
  Extended Authentication User Name ..... -
  Extended Authentication Password ...... -
  Key Id .............................. 30
  Local RSA key ....................... -
  Peer Certificate Id ................. -
  Phase 2 Exchanges Limit ............. NONE
  PreNegotiate ........................ TRUE
  DOI ................................. IPSEC
  Send Notify Messages ................ TRUE
  Send Delete Messages ................ FALSE
  Always Send ID Messages ............. FALSE
  Commit Bit .......................... FALSE
  Message Retry Limit ................. 5
  Message Time Out .................... 20
  Exchange Delete Delay ................ 30

...

   other parameters
```

Table 24: New parameter in the output of the **show isakmp policy** command for a specific policy

Parameter	Meaning
Exchange Delete Delay (s)	The delay period, in seconds, between the completion and the deletion of ISAKMP exchanges notified for this policy.

Figure 25: Example output from the **show isakmp exchange** command for a specific exchange in Main mode

```
ISAKMP Exchange

  Id ..................................... 4
  Type ................................... MAIN
  State .................................. SASENT
  Phase .................................. 1
  Initiator .............................. TRUE
  DOI .................................... IPSEC
  Policy name ............................ main
  SA ..................................... 1
  Peer IP Address ........................ 202.36.163.201
  Local IP Address ....................... 202.36.163.161
  Encrypted .............................. FALSE
  Expecting message ...................... TRUE
  Has SA ................................. TRUE
  Initiator Cookie ....................... d464cc30b348efa7
  Responder Cookie ....................... 0000000000000000
  Message Id ............................. 00000000
  Set Commit bit ......................... FALSE
  Commit bit received .................... FALSE
  Send notifies .......................... TRUE
  Send deletes ........................... FALSE
  Message Retry Limit .................... 5
  Packet Retry Counter ................... 5
  Delete Delay Time (s) ................. 30
  Delete Delay Timer (time left (s)) .... 8
  Initial Message Retry Timeout (s) ..... 20
  Packet Retry Timer (time left(s)) ..... 10

...

other parameters
```

Table 25: Parameters in the output of the **show isakmp exchange** command.

Parameter	Meaning
State	... other parameters ....
	For informational exchanges, the State is IDLE.
	For completed exchanges awaiting deletion, the State is DELETEDELAY.
Delete Delay Time (s)	The number of seconds between the completion of the exchange and its deletion.
Delete Delay Timer (time left (s))	The number of seconds left before a completed exchange is deleted.

The following new log message has been added to the Log Reference.

Retransmission received after completion

Message `Retransmission received after completion`

Severity	Module	Type	Subtype
3/INFO	82/ISAK	043/IKMP	004/MSG

Explanation This message is logged when a retransmission of the last message from the ISAKMP peer is received after the ISAKMP exchange has completed, and the exchange is in the DELETEDELAY state. This indicates that the peer did not receive the last message that the device transmitted in the ISAKMP exchange.

Recommended Action This could indicate that the last message that the device transmitted was lost due to network congestion. The device will resend its last message again, in an attempt to recover the ISAKMP exchange.

Compare the ID of the IKE Packet

To increase security during IKE negotiation, the subject value in the certification received from the IKE authorisation message is now validated against the ID value in the same message.

Increase Number of IKE Key Exchanges

The maximum number of simultaneous IKE exchanges has been increased.

PPTP Pass Through

This enhancement enables multiple Point-To-Point Tunnelling Protocol (PPTP) tunnels to be initiated from the private side of the firewall by default. This will allow private users to terminate their PPTP tunnels on their respective corporate or private networks. From the public side of the firewall, PPTP tunnels are blocked by default.

PPTP is a tunnelled mechanism to transfer Point-to-Point Protocol (PPP) frames across an intermediate network. PPTP connection is a method to create secure connections across public networks, such as the Internet, for both remote access and router-to-router virtual private network (VPN) connections, utilising the authentication, encryption, and protocol configuration mechanisms of PPP.

PPTP uses a TCP connection for tunnel management and Generic Routing Encapsulation (GRE) is used to encapsulate PPP frames for tunnelled data. In this implementation of PPTP, GRE tunnels are created and closed automatically when PPTP is enabled and disabled. The payloads of the encapsulated PPP frames can be either encrypted or compressed, or both encrypted and compressed.

PPTP has been added to the list of pre-defined service names you can specify when adding or modifying a firewall policy rule.

Table 26: New pre-defined IP protocol service name

Service Name	Port Number	Standard Protocol
PPTP	1723	TCP

When PPTP clients are on the private side of the firewall, to add a rule **denying** PPTP tunnel traffic when adding or modifying a firewall policy rule, use the commands:

```
add firewall policy=policy-name rule=rule-id action=deny
   interface=interface protocol=tcp port=pptp
   [other-parameters]

set firewall policy=policy-name rule=rule-id action=deny
   interface=interface protocol=tcp port=pptp
   [other-parameters]
```

When PPTP clients are on the public side of the firewall, to add a rule **allowing** PPTP tunnel traffic when adding or modifying a firewall policy rule, use the commands:

```
add firewall policy=policy-name rule=rule-id action=allow
   interface=interface ip=ipadd[-ipadd] protocol=tcp
   port=pptp gblip=ipadd gblport=pptp [other-parameters]

set firewall policy=policy-name rule=rule-id action=allow
   interface=interface ip=ipadd[-ipadd] protocol=tcp
   port=pptp gblip=ipadd gblport=pptp [other-parameters]
```

The **port** parameter specifies a port number, a range of port numbers, or a predefined service name to match. If dynamic NAT is active on the interface, it is possible to re-map a global port number to a different internal port number. For rules applied to a private interface, PORT is the destination port on the public network. For rules applied to a public interface, PORT is either the

destination port on the private network or, in the case of NAT being applied, the destination port on the private network where traffic will be mapped.

The **gblport** parameter specifies the port number, service name, or range of port numbers that apply to the rule if NAT is active on an interface.

PPTP has been added to the list of pre-defined service names you can specify when adding a local private IP network to the address translation table used by NAT.

Table 27: New service name for use with Network Address Translation (NAT)

Service Names	Value
PPTP	1723

To specify PPTP when adding a local private IP network to the address translation table used by NAT, use the command:

```
add ip nat ip=ipadd gblport=pptp port=pptp protocol=tcp
    [mask=ipadd] [gblip=ipadd] [gblmask=ipadd]
    [gblinterface=interface]
```

The **port** parameter specifies the port number or service name for the port used on the private IP host when specifying a static ENAT entry.

The **gblport** parameter specifies the port number or service name for the port available to global Internet access, when creating a static ENAT.

IPsec NAT-Traversal

IPsec NAT-T is an enhancement to IPsec and ISAKMP protocols that lets Virtual Private Network (VPN) clients communicate through NAT gateways over the Internet. For example, business travellers commonly use IPsec on their laptops to gain remote VPN access to the central office. When working off-site, these users sometimes need to connect to the Internet through a NAT gateway such as from a hotel. Network Address Translation (NAT) gateways are often part of a company's firewall and let its Local Area Network (LAN) appear as one IP address to the world. For more information about NAT gateways, refer to RFC 1631 and to the *Network Address Translation* section in the *Internet Protocol* chapter of the *Reference Manual*.

Problems arise with NAT gateways for a number of reasons. A key one is that when they handle IPsec packets, they cannot access encrypted UDP or TCP headers. Therefore, NAT gateways cannot identify traffic for different private devices and cannot properly track individual sessions.

NAT-T is not on the NAT gateway and is not an "IPsec pass-through". NAT-T lets IPsec/ISAKMP peers send traffic through NAT gateways by putting packets inside UDP packets. This solution enables remote VPN users to communicate successfully when NAT gateways are part of the connection.

Basic NAT-T Operations

Using NAT-D (discovery) messages, NAT-T negotiates with a peer to determine if NAT gateways are present and at which end of the network. Each peer sends at least two NAT-D messages as part of the ISAKMP phase 1 negotiation. The first message contains a hash of a destination IP address; subsequent messages contain source addresses. A NAT gateway is detected when address messages from the peer have incorrect hash values, which indicates that a NAT gateway changed IP addresses.

Also during phase 1, NAT-T determines whether a peer has NAT-T capabilities by detecting a vendor ID. Vendor IDs tell what version of NAT-T the peer supports. When a NAT gateway is not detected or a peer does not support NAT-T, normal IPsec negotiations and protection occur.

When an ISAKMP initiator detects a NAT gateway during an exchange, communication changes from UDP port 500 to port 4500. Log messages inform users that the UDP port has changed. Main or Aggressive mode packets received on the old port are discarded and a separate log is created.

Because IPsec traffic can also be received on port 4500, ISAKMP adds and removes the non-ESP marker at the start of the ISAKMP message so that messages can be detected and passed to the ISAKMP module. ISAKMP drops packets when it receives them on port 4500 without a non-ESP marker.

NAT-T inserts a UDP header between the outer IP and ESP headers thereby encapsulating the ESP data (Figure 26). NAT-T encapsulates IPsec traffic only when a NAT gateway is detected.

Figure 26: UDP Encapsulation for NAT-T

IPsec intercepts UDP-encapsulated ESP packets before they are passed to UDP.

A peer behind a NAT gateway sends keepalive messages to ensure that port mappings in the device remain active between peers. Keepalive intervals are not configurable. The purpose of keepalive messages is different from heartbeat messages controlled by the **heartbeatmode** parameter in ISAKMP Policy commands, which detect an IKE peer. IKE heartbeat messages and NAT-T keepalive messages do not affect each other.

NAT-T on the Router or Switch

This NAT-T implementation supports interoperability with the following VPN clients:

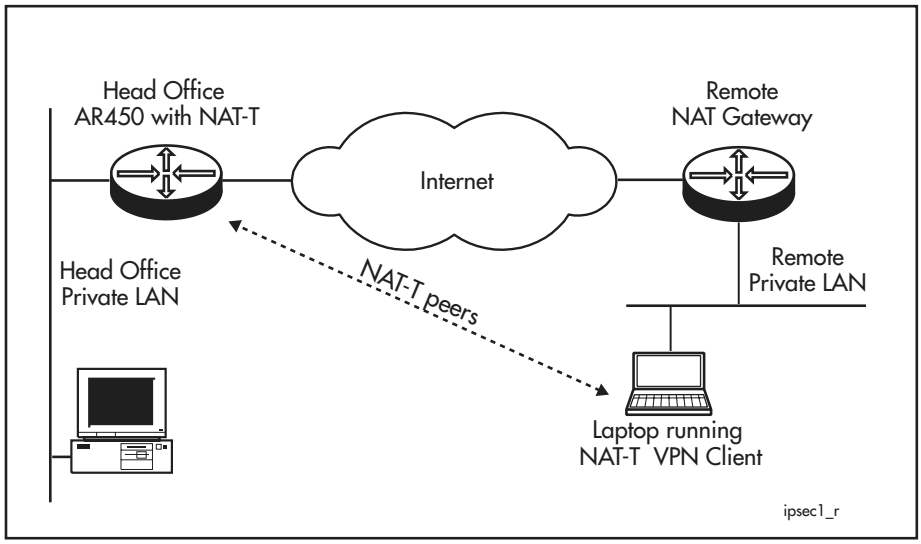
- SafeNet SoftRemote®
- Microsoft Windows 2000®
- Microsoft Windows XP®

NAT-T can also be implemented router-to-router for offices with their own IPsec router behind a NAT gateway. For router-to-VPN examples, see Configuration Notes for interoperability with SafeNet SoftRemote clients, and Microsoft Windows 2000 and XP VPN clients at www.alliedtelesyn.co.nz/solutions/solutions.html.

Versions 02 and 08 of the following NAT-T IETF drafts have been implemented:

- *Negotiation of NAT-Traversal in the IKE*, draft-ietf-ipsec-nat-t-ike-02, which describes the modifications to IKE to support NAT detection and UDP tunnel negotiation
- *UDP Encapsulation of IPsec Packets*, draft-ietf-ipsec-udp-encaps-02, which defines the method of UDP encapsulation of IPSec packets
- *Negotiation of NAT-Traversal in the IKE*, draft-ietf-ipsec-nat-t-ike-08, which describes the modifications to IKE to support NAT detection and UDP tunnel negotiation
- *UDP Encapsulation of IPsec Packets*, draft-ietf-ipsec-udp-encaps-08, which defines the method of UDP encapsulation of IPSec packets

Figure 27: IPsec NAT-T peers negotiate traffic through a NAT gateway device



NAT-T is enabled by default, and is enabled or disabled in the ISAKMP policy. We recommend that users carefully read security considerations in the IETF drafts to fully understand the implications of using NAT-T. When users create an ISAKMP policy with the **create isakmp policy** command, they define how peers respond during an ISAKMP exchange, and can use the **nattraversal** parameter to disable NAT-T if they prefer. They can later use the same parameter with the **set isakmp policy** command to enable NAT-T again.

Users should configure an IPsec policy to allow ISAKMP traffic on UDP port 4500 so that it flows through the IPsec layer to ISAKMP. Refer to the ISAKMP policy commands in this chapter to change or add a policy.

Peers send their original, untranslated addresses to each other, which they store in the ISAKMP SA. Recipients use original addresses (OAs) to correct the checksums in the UDP or TCP headers in the IPsec payload.

NAT-T is implemented for IPv4 for both transport and tunnel modes. We recommend transport mode for MS clients. For SafeNet and router-to-router connections, we recommend tunnel mode and specifying unique IP addresses for remote peers. Refer to the **set ipsec saspecification** command to set modes.

Commands Modified for NAT-T

- set isakmp policy
- **show ipsec sa**
- **show ipsec counter=main**
- **show isakmp policy**
- **show isakmp policy**
- **show isakmp counter**=general
- **show isakmp counter**=aggressive
- **show isakmp counter**=main
- **show isakmp counter**=network
- **show isakmp counter**=quick

ISAKMP policy commands

The existing **create isakmp policy** and **set isakmp policy** commands have been modified to add a **nattraversal** parameter. This parameter enables or disables NAT-T to let peers negotiate a UDP-encapsulated mode so that IPsec traffic can flow through a NAT gateway. The default is enabled.

Syntax

```
CREate ISAkmp POLICY=name PEer={ipv4add|ipv6add|ANy}
   [NATTraversal={ON|OFf|TRue|FAlse}]
   [other-isakmp-parameters]

SET ISAkmp POLicy=name [NATTraversal={ON|OFf|TRue|FAlse}]
   [other-isakmp-parameters]
```

show ipsec sa

Output for the existing **show ipsec sa** command has been modified to include the following parameters that are specific to NAT-T.

Parameter	Meaning
SA id	The identification number for the SA.
Role	Whether this peer acted as the initiator or responder in order to create this SA.
Mode	The IPsec operational mode for this SA: TUNNEL TRANSPORT UDP_ENCAPSULATED_TUNNEL UDP_ENCAPSULATED_TRANSPORT
NAT-Traversal NAT-OA	Information about original IP addresses.
Peer original source IP address	Source IP address that the remote peer uses when sending packets to this peer. UDP-encapsulated transport mode only.
Peer original destination IP address	Destination IP address that the remote peer uses when sending packets to this peer. UDP-encapsulated transport mode, and IETF draft v08, *Negotiation of NAT-T in the IKE*.
Filters	Information about the packet selections for this SA.
NAPT remote port number	Network Address Port Translation number. Multiple clients in UDP-encapsulated transport mode appear to come from the same source. Therefore, NAT-T changes the source port to this value to maintain a distinction.

show ipsec counter=main

Parameter	Meaning
IPsec main packet processing counters	
inProcessPktKeepalive	The number of NAT keepalive packets received.

show isakmp policy

Parameter	Meaning
NAT Traversal	Whether NAT-T is enabled or disabled.

show isakmp sa

Parameter	Meaning
NAT-Traversal Information	Information about NAT-T capability.
NAT-T enabled	Whether NAT-T is enabled on the router or switch.
Peer NAT-T capable	Whether the remote peer sent a valid NAT-T vendor ID.
NAT discovered	Whether a NAT gateway has been detected between peers: No - Not detected Remote - Detected at the remote site Local - Detected at this site Both - Detected at local and remote sites Unknown - Peer is not NAT-T capable or the NAT discovery process was incomplete

show isakmp counter

- main
- network
- quick

Table 28: NAT-T parameters in the output of the **show isakmp counter=general** command

Parameter	Meaning
msgRxBadPortIpChange	The number of ISAKMP packets received with an unexpected source IP address or source port and discarded.
msgRxFailOldPort	The number of ISAKMP packets discarded because they were received on port 500 after NAT-T had moved the ISAKMP traffic to port 4500.

Table 29: NAT-T parameters in the output of the **show isakmp counter=aggressive** command

Parameter	Meaning
initSendNatD	The number of NAT-D messages sent by the initiator.
respSendNatD	The number of NAT-D messages sent by the responder.
initRecvNatD	The number of NAT-D messages received by the initiator.
respRecvNatD	The number of NAT-D messages received by the responder.

Table 30: NAT-T parameters in the output of the **show isakmp counter=main** command

Parameter	Meaning
initSendNatD	The number of NAT-D messages sent by the initiator.
initRecvNatD	The number of NAT-D messages received by the initiator.
respSendNatD	The number of NAT-D messages sent by the responder.
respRecvNatD	The number of NAT-D messages received by the responder.

Table 31: NAT-T parameter in the output of the **show isakmp counter=network** command

Parameter	Meaning
rxFailNoNonEspMarker	The number of packets ISAKMP dropped because they were received on NAT-T port 4500 without a non-ESP marker.

Table 32: NAT-T parameters in the output of the **show isakmp counter=quick** command

Parameter	Meaning
natOaSent	The number of NAT originating messages sent.
natOaReceived	The number of NAT originating messages received.
badNatOa	The number of non-conforming NAT-OA (originating address) messages received such as unknown type.

NAT-T Configuration Example

This is a basic router-to-router solution with NAT-Traversal for a Virtual Private Network (VPN) that shows:

- NAT gateways at both ends of the VPN link
- a firewall configuration at both ends

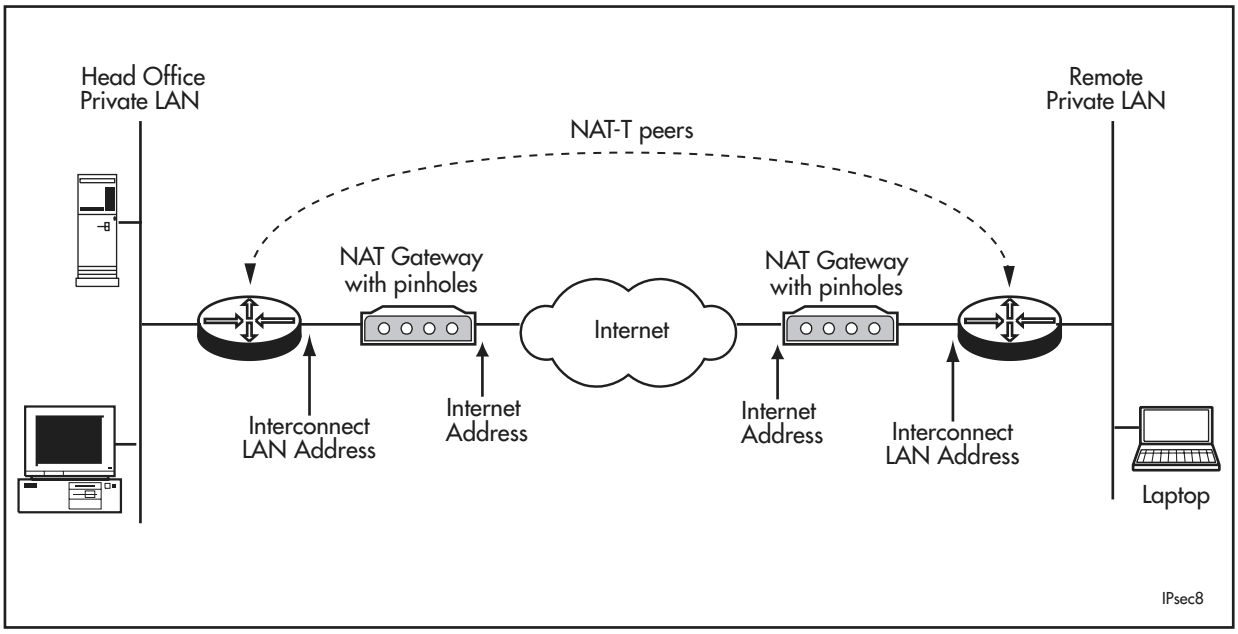
For solutions that enable office-to-office VPN access through the Internet, as well as VPN access for travellers with VPN clients, see www.alliedtelesyn.co.nz/solutions/solutions.html for interoperability with Microsoft Windows 2000 and XP VPN clients, and with SafeNet SoftRemote clients.

General Considerations

- This example also works with a NAT gateway at the initiator end, responder end, or neither end.
- IPsec and ISAKMP policies must refer to valid Internet peer addresses. When a peer router is directly connected to the Internet, this address is on its local WAN interface. When a peer accesses the Internet through a NAT gateway, this address is on the NAT gateway.
- If you have a NAT gateway at the responder end, it must be configured to allow traffic (*pinholes*) for UDP ports 500 and 4500.
- One end of the link must have a fixed Internet address. To enable both peers to initiate an IPsec link, both ends must have fixed addresses and both NAT gateways must have pinholes for UDP ports 500 and 4500.

  Many ISPs assign dynamic addresses that may change periodically, and you may need to ask for a fixed address.
- ISAKMP peers behind NAT gateways must identify themselves using a name string.

Figure 28: NAT-T router-to-router solution in IPsec tunnel mode



The router must be in secure mode and you must log on as a user with security officer privilege. The router must also already have pre-shared keys for ISAKMP to use. See the IP Security chapter in the Software Reference for details.

Figure 29: Head Office Router

```
set system name="IPsec Head Office"
set user securedelay=600
add user=secoff pass=secoff priv=sec login=yes telnet=yes
del user=manager

# The IPsec peer will be authenticated by the following name
# and password
add user=remote password=friend

# IP configuration
# Smaller MRU/MTU settings are needed for IPsec tunnel
# mode so that larger payload packets successfully pass
# through the IPsec tunnel
enable ip
set int=eth0 mtu=1300
add ip int=eth0 ip=interconnect-LAN-address frag=yes
add ip int=vlan1 ip=head-office-LAN-address
add ip rou=0.0.0.0 mask=0.0.0.0 int=eth0
  next=NAT-gateway-address

enable fire
create fire poli=main
add fire poli=main int=vlan1 type=private
add fire poli=main int=eth0 type=public
add fire poli=main nat=enhanced int=vlan1 gblint=eth0
add fire poli=main rule=1 int=eth0 action=allow
  ip=interconnect-LAN-address prot=udp port=500
  gblip=interconnect-LAN-address gblpo=500
add fire poli=main rule=2 int=eth0 action=allow
  ip=interconnect-LAN-address prot=udp port=4500
  gblip=interconnect-LAN-address gblpo=4500
add fire poli=main rule=3 int=eth0 action=nonat prot=all
   ip=head-office-LAN-ip-range x.x.x.x - x.x.x.x encap=ipsec

# Rule 4 for internally initiated VPN traffic to the
# remote site
add firewall poli=main ru=4 ac=nonat int=vlan1 prot=all
  ip=head-office-LAN-ip-range x.x.x.x - x.x.x.x
  poli=main ru=4 remoteip=remote-LAN-ip-range
  x.x.x.x - x.x.x.x

# IPsec configuration
create ipsec sas=1 key=isakmp prot=esp enc=des hasha=sha
create ipsec bund=1 key=isakmp string="1"
create ipsec pol=isakmp int=eth0 ac=permit
set ipsec pol=isakmp lp=500
create ipsec pol=natt_udp int=eth0 ac=permit
set ipsec pol=natt_udp lp=4500

# The following peer's internet address may be on its
# NAT gateway or its router. If the remote site has a
# dynamically assigned address, use the keyword "dynamic".
create ipsec pol=remote_site int=eth0 ac=ipsec key=isakmp
  bund=1 peer=remote-internet-address isa=to_remote
```

Figure 29: Head Office Router (Continued)

```
set ipsec pol=remote_site
  lad=head-office-LAN-ip-subnet-address
  lmask=head-office-LAN-ip-subnet-mask
  rad=remote-LAN-ip-subnet-address
  rmask=remote-LAN-ip-subnet-mask

# If you need both VPN and internet-browsing access, use
# the following internet policy. Do not use this policy
# for VPN only.
create ipsec pol=internet int=eth0 ac=permit
enable ipsec

# ISAKMP configuration
# The following peer's internet address may be on its
# NAT gateway or its router. If the remote site has a
# dynamically assigned address, use the keyword "any".
create isakmp pol=to_remote
  peer=remote-internet-address key=1

# The following heartbeat parameter is optional and lets
# the office delete inactive SAs if the remote office
# connection to the Internet drops. Heartbeats can be set
# to send, receive or both. The receiver expects to receive
# heartbeats; when three are missing, it deletes the
# associated SA to avoid SA "out of step" fault conditions.
set isakmp pol=to_remote localid=head_office
  heartbeat=receive
set isakmp pol=to_remote sendd=true setc=true

# For XAUTH to authenticate the IPsec peer
set isakmp pol=to_remote xauth=server xauthtype=generic
enable isakmp
```

Figure 30: Remote Site Router

```
set system name="IPsec Remote Site"
set user securedelay=600
add user=secoff pass=secoff priv=sec login=yes telnet=yes
del user=manager

# IP configuration
# Smaller MRU/MTU settings are needed for IPsec tunnel
# mode so that larger payload packets successfully pass
# through the IPsec tunnel
enable ip
set int=eth0 mtu=1300
add ip int=eth0 ip=interconnect-LAN-address frag=yes
add ip int=vlan1 ip=remote-LAN-address
add ip rou=0.0.0.0 mask=0.0.0.0 int=eth0
  next=NAT-gateway-address

enable fire
create fire poli=main
add fire poli=main int=vlan1 type=private
add fire poli=main int=eth0 type=public
add fire poli=main nat=enhanced int=vlan1 gblint=eth0
add fire poli=main rule=1 int=eth0 action=allow
   ip=interconnect-LAN-address prot=udp port=500
   gblip=interconnect-LAN-address gblpo=500
add fire poli=main rule=2 int=eth0 action=allow
   ip=interconnect-LAN-address prot=udp port=4500
   gblip=interconnect-LAN-address gblpo=4500
add fire poli=main rule=3 int=eth0 action=nonat prot=all
  ip=remote-LAN-ip-range x.x.x.x - x.x.x.x encap=ipsec

# Rule 4 for internally initiated VPN traffic to the remote
# site
add firewall poli=main ru=4 ac=nonat int=vlan1 prot=all
   ip=remote-LAN-ip-range x.x.x.x - x.x.x.x
   poli=main ru=4 remoteip=head-office-LAN-ip-range
   x.x.x.x - x.x.x.x

# IPsec configuration
create ipsec sas=1 key=isakmp prot=esp enc=des hasha=sha
create ipsec bund=1 key=isakmp string="1"
create ipsec pol=isakmp int=eth0 ac=permit
set ipsec pol=isakmp lp=500 rp=500
create ipsec pol=natt_udp int=eth0 ac=permit
set ipsec pol=natt_udp lp=4500 rp=4500

# The following peer's internet address may be on its
# NAT gateway or its router. This example assumes the head
# office has a fixed address.
create ipsec pol=head_office int=eth0 ac=ipsec
   key=isakmp bund=1 peer=head-office-internet-address
   isa=to_office
set ipsec pol=head_office
   lad=remote-LAN-ip-subnet-address
   lmask=remote-LAN-ip-subnet-mask
   rad=head-office-LAN-ip-subnet-address
   rmask=head-office-LAN-ip-subnet-mask
```

Figure 30: Remote Site Router (Continued)

```
# If you need both VPN and internet-browsing access, use
# the following internet policy. Do not use this policy
# for VPN only.
create ipsec pol=internet int=eth0 ac=permit
enable ipsec

# ISAKMP configuration
# The following peer's internet address may be on its
# NAT gateway or its router. This example assumes the head
# office has a fixed address.
create isakmp pol=to_office
  peer=head-office-internet-address key=1

# The following heartbeat parameter is optional and lets
# the office delete inactive SAs if the remote office
# connection to the Internet drops. Heartbeats can be set
# to send, receive or both. The receiver expects to receive
# heartbeats; when three are missing, it deletes the
# associated SA to avoid SA "out of step" fault conditions.
set isakmp pol=to_office localid=remote_site heartbeat=send
set isakmp pol=to_office sendd=true setc=true

# For XAUTH to authenticate the IPsec peer
set isakmp pol=to_office xauth=client xauthname=remote
  xauthpass=friend
enable isakmp
```

Email Relaying

How the SMTP application gateway protects against third party relaying of email has been enhanced.

A third-party mail relay occurs when a mail server processes a mail message where neither the sender or the recipient is a local user. The mail server is an entirely unrelated party to mail processing. If an email originates from the public side of the firewall, the firewall SMTP proxy will reject the email if the address in the "RCPT TO" field has a different domain name to a mail server on the private side of the firewall. If an email originates from the private side of the firewall, the firewall SMTP proxy will reject the email if either:

- domain name in the "MAIL FROM" field is different to domain name specified by the **set firewall policy smtpdomain** command, or
- domain name in the "RCPT TO" field is not consistent with IP address of the IP packet.

Note that for the latter to occur, a DNS server must be must be setup by using the **add ip dns** command, *Internet Protocol (IP)* chapter. If a DNS server is not configured, the proxy will only check the email based on the "MAIL FROM" field.

The firewall SMTP proxy can relay any email that originates from the private side of the firewall. This happens when the IP packets for the email are only destined to the private interface of the firewall. The proxy will forward the email to the final destination specified in the "RCPT TO" field. Note that the relaying function requires that a DNS server is setup using the **add ip dns** command, *Internet Protocol (IP)* chapter.

The behaviour of the **disable firewall policy smtprelay** command has been enhanced. This command disables emails that intend to use the third party relaying mechanism for delivery from passing through the SMTP proxy.

The behaviour of the **enable firewall policy smtprelay** command has been enhanced. This command enables emails which intend to use the third party relaying mechanism to pass through the SMTP proxy.

The behaviour of the **set firewall policy smtpdomain** command has been enhanced. This command sets a domain name for the SMTP proxy. The domain name is normally the same as the SMTP server that is located on the private side of the firewall. The specified domain name is used to compare with either:

- the domains of the destination addresses of all SMTP sessions that originate from the public side of the firewall, or
- the domains of the source addresses of all SMTP sessions that originate from the private side of the firewall.

If the domain name does not match, the firewall concludes that the email is trying to use the third party relay mechanism for delivery. If SMTP relaying is disabled then the session is terminated.

For more information about SMTP proxy and the full syntax of these commands, see the Firewall chapter in the Software Reference.

Chapter 1

Multiple Spanning Tree Protocol (MSTP)

Multiple Spanning Tree Protocol (MSTP) 1-2
Multiple Spanning Tree Regions 1-2
Bridge Protocol Data Units (BPDUs) 1-3
Compatibility with Previous Spanning Tree Protocols 1-5
Configuring MSTP 1-6
Common and Internal Spanning Tree (CIST) 1-12
The Relationship between Spanning Trees and Trunks 1-16
Command Reference 1-16
activate mstp migrationcheck port 1-17
add mstp msti vlan 1-18
create mstp msti 1-19
delete mstp msti vlan 1-21
destroy mstp msti 1-21
disable mstp 1-22
disable mstp cist port 1-22
disable mstp debug msti 1-23
disable mstp msti port 1-24
enable mstp 1-25
enable mstp cist port 1-26
enable mstp debug 1-27
enable mstp msti port 1-29
purge mstp 1-30
reset mstp counter port 1-30
set mstp 1-31
set mstp cist 1-33
set mstp cist port 1-34
set mstp msti 1-36
set mstp msti port 1-37
show mstp 1-39
show mstp cist 1-41
show mstp cist port 1-43
show mstp counter port 1-45
show mstp debug msti 1-46
show mstp msti 1-47
show mstp msti port 1-49

Multiple Spanning Tree Protocol (MSTP)

Multiple Spanning Tree Protocol (MSTP) was developed to address limitations in the existing protocols, Spanning Tree Protocol (STP) and Rapid Spanning Tree Protocol (RSTP). These limitations apply mainly to networks that use multiple VLANS with topologies employing alternative physical links. MSTP is defined in IEEE Standard 802.1Q 2003. MSTP builds on and remains compatible with the following standards:

- IEEE Standard 802.1w, which defines the rapid spanning tree protocol (RSTP)
- IEEE Standard 802.1D/D4 2003, which defines a draft standard for local and metropolitan area networks

Multiple Spanning Tree Regions

Conceptually, MSTP views the total bridged network as one that comprises a number of *Multiple Spanning Tree Regions* (MSTRs), where each region can contain up to 64 spanning trees that operate locally, called *Multiple Spanning Tree Instances* (MSTIs). The task of assigning each bridge to a particular region is achieved by the member bridges each comparing their bridge protocol configuration identifiers. More information on configuration identifiers is in Table 1-1 on page 1-3, but for the moment a *bridge protocol configuration identifier* can simply be thought of as an identifier that represents the mapping of VLANs to MSTIs within each bridge. Therefore, bridges with identical VLAN-to-MSTI mapping tables have identical bridge protocol identifiers.

While each MSTI can contain up to 4096 VLANs, each VLAN can be associated with only one MSTI. Once these associations have been made, the bridges in each region can transmit their spanning tree algorithms and advertise their MSTIs. This in turn establishes the active data paths between the bridges for each group of VLANS (i.e. for each MSTI) and block any duplicate paths. A particular advantage of this enhancement applies where a large number of VLANS share a few internetwork paths. In this situation there need only be as many Multiple Spanning Tree Instances (MSTIs) as there are source, and destination bridge pairs, remembering that a pair of bridges probably has multiple paths between them.

To ensure that each bridge in a region maintains the same configuration information (particularly their VID to MSTI mappings) and to ensure each bridge's membership of a particular region, the bridges exchange configuration information in the form of *MST configuration identifiers*. Table 1-1 on page 1-3 explains an MST configuration identifier. A detailed explanation of bridge configuration identifiers can be found in Section 13.7 of the IEEE Standard 802.1Q-2003.

Table 1-1: MST configuration identifiers

Field Name	Description
Format Selector	Single octet field whose value of 0 indicates MSTP operation.
Configuration Name	Name (up to 32 characters long) that identifies a particular MST region. The configuration name is defined with the SET MSTP command.
Revision Level	Number representing the region's revision level. This value is normally set to 0.
Configuration Digest	16-octet (HMAC-MD5 based) signature created from the MST configuration table.

Bridge Protocol Data Units (BPDUs)

The main function of bridge protocol data units is to enable MSTP to select its root bridges for the CIST and each MSTP. MSTP is compatible with earlier spanning tree versions; its Bridge Protocol Data Unit (BPDU) formats build on earlier versions; see Compatibility with Previous Spanning Tree Protocols on page 5. Table 1-2 on page 3 shows the standardised format for MSTP BPDU messages. The general format of the BPDUs comprise a common generic portion—octets 1 to 36—that are based on those defined in IEEE Standard 802.1D, 1998, followed by components that are specific to CIST—octets 37 to 102. Components specific to each MSTI are added to this BPDU data block. These are shown in Table 1-3 on page 1-5.

Table 1-2: MST Bridge Protocol Data Units (BPDUs)

Field Name	Octets	Description
Protocol Identifier	1–2	Protocol being used. The value 0000 0000 0000 0000 identifies the Spanning Tree algorithm and protocol.
Protocol Version Identifier	3	Protocol version and has the value 0000 0000.
BPDU Type	4	Value 0000 0000 specifies a configuration BPDU.
CIST Flags	5	Bit1 is the topology change flag. Bit 2 conveys the CIST Proposal flag in RST and MST BPDUs - unused in STP. Bits 3 & 4 convey the CIST Port Role in RST, and MST BPDUs - unused in STP. Bit 5 conveys the CIST Learning flag in RST and MST BPDUs - unused in STP. Bit 6 conveys the CIST Forwarding flag in RST and MST BPDUs - unused in STP. Bit 7 conveys the CIST Agreement flag in RST and MST BPDUs - unused in STP. Bit 8 conveys the Topology Change Acknowledge Flag in STP Configuration BPDUs - unused in STP.
CIST Root Identifier	6–13	Value of the designated CIST root parameter held by the bridge. This has the same value as the designated bridge and the CIST bridge.
CIST External Path Cost	14–17	Cost of the path from the bridge to the root.
CIST Regional Root Identifier	18–25	ID of the current CIST regional root bridge.

Table 1-2: MST Bridge Protocol Data Units (BPDUs)

CIST Port Identifier	26–27	CIST port identifier of the transmitting bridge port.
Message Age	28–29	Message age timer value.
Max Age	30–31	Timeout value to be used by all bridges in the bridged network. This value is set by the root. Some implementations of MSTP may choose not to use this value.
Hello Time	32–33	Time interval between configuration BPDUs by the root.
Forward Delay	34–35	Timeout value to ensure forward delay timer consistency when transferring a port to the forwarding state. Also used for ageing filtering database dynamic entries following changes in the active topology.
Version 1 Length	0 36	Version 1 Length. It is always transmitted as 0.
Version 3 Length	37–38	Version 3 length. Is an integral number, from 0 to 64 inclusive, of MSTI configuration messages.
MST Configuration Identifier	39–89	Name for the revision level, plus a summary of the mapping of VLANs to Spanning Trees.
CIST Internal Root Path Cost	90–93	Path cost to the CIST regional root.
CIST Bridge Identifier	94–101	CIST bridge identifier of the transmitting bridge.
CIST Remaining Hops	102	Remaining hops which limits the propagation and longevity of received spanning tree information for the CIST.
MST Configuration Identifier	39–89	See **Multiple Spanning Tree Regions** command on page 1-2.
CIST Internal Root Path Cost	90–93	Path cost to the CIST regional root.
CIST Bridge Identifier	94–101	CIST bridge identifier of the transmitting bridge. This value is the same as the values for the root identifier and the designated bridge identifier.
CIST Remaining Hops	102	Number of CIST hops remaining.
MSTI Configuration Messages (may be absent)	103–39 plus Version 3 Length	See Table 1-3 on page 1-5.

Table 1-3: MSTI configuration messages

Field Name	Octets	Description
MSTI Flags	1	Bits 1 through 8, convey the Topology Change flag, Proposal flag, Port Role (two bits), Learning flag, Forwarding flag, Agreement flag, and Master flag for this MSTI.
MSTI Regional Root Identifier	2–9	This includes the value of the MSTID for this configuration message encoded in bits 4 through 1 of Octet 1, and bits 8 through 1 of Octet 2.
MSTI Internal Root Path Cost	10-13	Internal Root Path Cost.
MSTI Bridge Priority	14	Bits 5 through 8 convey the value of the bridge identifier priority for this MSTI. Bits 1 through 4 of Octet 14 are transmitted as 0, and ignored on receipt.
MSTI Port Priority	15	Bits 5 through 8 are used to convey the value of the port identifier priority for this MSTI. Bits 1 through 4 are transmitted as 0, and ignored on receipt.
MSTI Remaining Hops	16	Value of remaining hops for this MSTI.

Compatibility with Previous Spanning Tree Protocols

MSTP is compatible with older spanning tree protocols in several ways. In addition to the MSTR described in the previous section, the protocol provides for single spanning tree systems by employing the Common and Internal Spanning Tree (CIST) protocol. The CIST applies a common spanning tree protocol to the whole bridged network and is a direct equivalent the Internal Spanning Tree (IST) protocol of earlier versions.

As with legacy spanning tree systems, the CIST protocol first determines its root bridge from all the bridges on the network. This is the bridge that contains the lowest bridge identifier. The protocol then selects a regional root bridge for each MSTR. This is the bridge that provides the best path to the CIST root. After MSTR root bridges are chosen, they act on the region's behalf in such a way that the region appears to the CIST as a virtual bridge. So in addition the having multiple MSTIs, each region **must** operate a CIST.

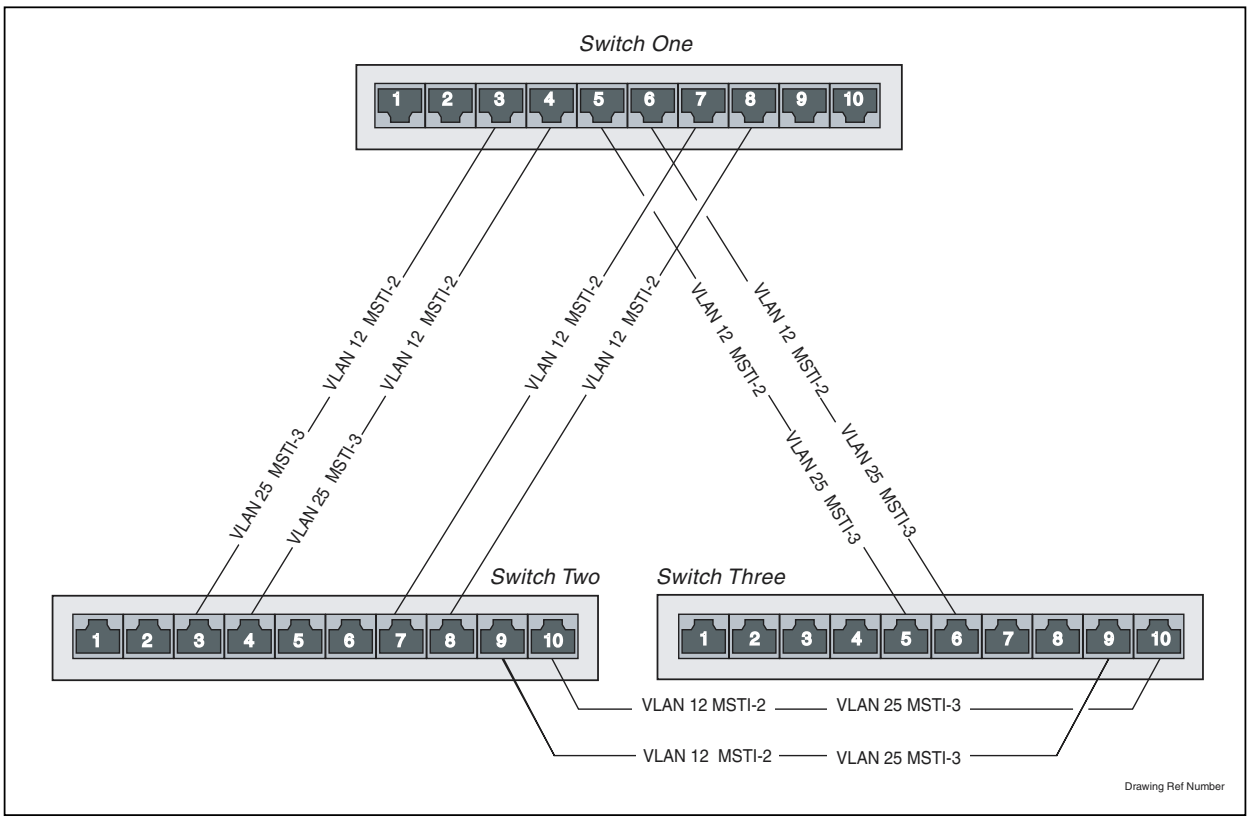
Configuring MSTP

The configuration examples in this section are based on the network shown in Figure 1-1 on page 1-6. This simple network comprises three LAN bridges connected in a multi-linked mesh configuration.

The network is configured as a single MSTP region, called a MSTR, and given the name, Head Office. Two spanning tree instances (MSTIs) are created within this region called MSTI-2 and MSTI-3. For simplicity only two VLANs are configured VLAN 12 and VLAN 25; however, a typical MSTI network could have many more VLANs.

Two MSTIs are created (MSTI-2 and MSTI-3). MSTI-2 is assigned to VLAN12 and MSTI-3 is assigned to VLAN25. The network has several alternative links. By using MSTP each VLAN can be configured to use its own preferred set of links

Figure 1-1: Example configuration using MSTP



Configure Switch 1

1. **Name the system and set manager port.**

```
set system name=switch-one
set manager asyn=0
```

2. **Create VLAN 12 and assign it a VID of 12.**

```
create vlan=vlan12 vid=12
```

3. **Create VLAN 25 and assign it a VID of 25.**

```
create vlan=vlan25 vid=25
```

4. **Add VLAN 12 to the required ports, as tagged ports.**

```
add vlan=12 po=3,4,5,6,7,8 frame=tagged
```

5. **Add VLAN 25 to the required ports, as tagged ports.**

```
add vlan=25 po=3,4,5,6 frame=tagged
```

6. **Create MSTP on Switch 1 Name the region Head Office and assign it a revision level of 0 (the value recommended in the IEEE standard).**

```
set mstp configname=headoffice revision=0
```

7. **Enable static VLAN support on MSTP.**

```
set mstp staticvlans=on
```

8. **Create the MSTIs 2 and 3.**

```
create mstp msti=2
create mstp msti=3
```

9. **Add MSTI-2 to VLAN 12, and MSTI-3 to VLAN 25.**

```
add mstp msti=2 vlan=12
add mstp msti=3 vlan=25
```

10. **Assign priorities to each MSTI. These values are compared with those set on the other switches in order to determine the root bridge for each MSTI.**

```
set mstp msti=2 prio=8192
set mstp msti=3 prio=8192
```

11. **Enable MSTP on the switch.**

```
ena mstp
```

Configure Switch 2

1. **Name the system and set manager port.**

```
set system name=switch-two
set manager asyn=0
```

2. **Create VLAN 12 and assign it a VID of 12.**

```
create vlan=vlan12 vid=12
```

3. **Create VLAN 25 and assign it a VID of 25.**

```
create vlan=vlan25 vid=25
```

4. **Add VLAN 12 to the required ports, as tagged ports.**

```
add vlan=12 po=3,4,5,6,7,8,9,10 frame=tagged
```

5. **Add VLAN 25 to the required ports, as tagged ports.**

```
add vlan=25 po=3,4,9,10 frame=tagged
```

6. **Create MSTP on Switch2. Name the region Head Office and assign it a revision level of 0 (the value recommended in the IEEE standard).**

```
set mstp configname=headoffice revision=0
```

7. **Enable static VLAN support on MSTP.**

```
set mstp staticvlans=on
```

8. **Create the MSTIs 2 and 3.**

```
create mstp msti=2
create mstp msti=3
```

9. **Add MSTI-2 to VLAN 12, and MSTI-3 to VLAN 25.**

```
add mstp msti=2 vlan=12
add mstp msti=3 vlan=25
```

10. **Assign priorities to each MSTI. These values are compared with those set on the other switches in order to determine the root bridge for each MSTI. Set MSTI 2 to 8192 and set MSTI 3 to 4096.**

```
set mstp msti=2 prio=8192
set mstp msti=3 prio=4096
```

11. **Enable MSTP on the switch.**

```
ena mst
```

Configure Switch 3

1. **Name the system and set manager port.**

```
set system name=switch-two
set manager asyn=0
```

2. **Create VLAN 12 and assign it a VID of 12.**

```
create vlan=vlan12 vid=12
```

3. **Create VLAN 25 and assign it a VID of 25.**

```
create vlan=vlan25 vid=25
```

4. **Add VLAN 12 to the required ports, as tagged ports.**

```
add vlan=12 po=5,6,9,10 frame=tagged
```

5. **Add VLAN 25 to the required ports, as tagged ports.**

```
add vlan=25 po=5,6,9,10 frame=tagged
```

6. **Create MSTP on Switch 2. Name the region Head Office and assign it a revision level of 0 (the value recommended in the IEEE standard).**

```
set mstp configname=headoffice revision=0
```

7. **Enable static VLAN support on MSTP.**

```
set mstp staticvlans=on
```

8. **Create the MSTIs 2 and 3.**

```
create mstp msti=2
create mstp msti=3
```

9. **Add MSTI-2 to VLAN 12, and MSTI-3 to VLAN 25.**

```
add mstp msti=2 vlan=12
add mstp msti=3 vlan=25
```

10. **Assign priorities to each MSTI. These values are compared with those set on the other switches in order to determine the root bridge for each MSTI. Set MSTI 2 to 4096 and let MSTI 3 take the default of 8192.**

```
set mstp msti=2 prio=4096
set mstp msti=3 prio=8192
```

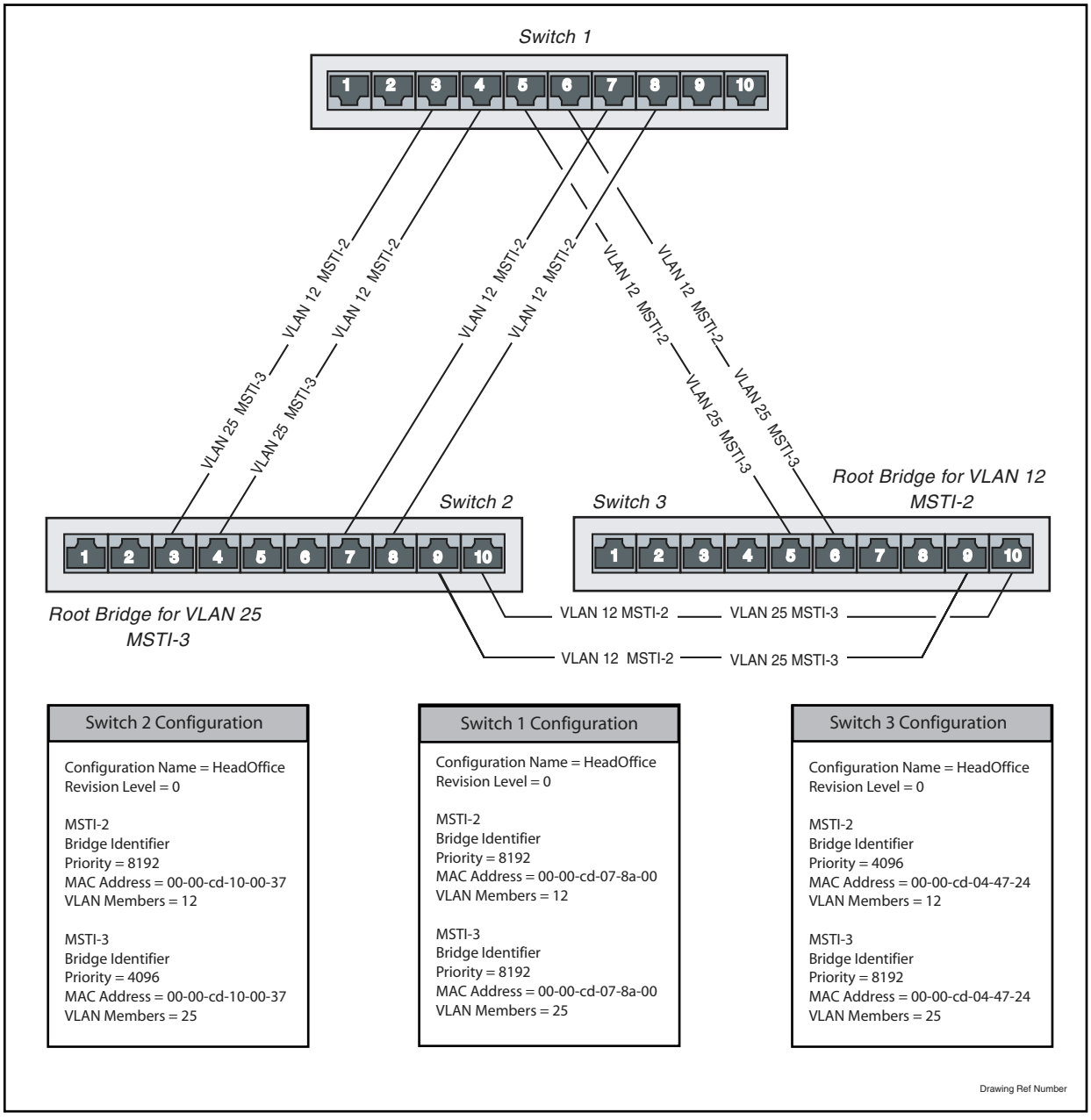
11. **Enable MSTP on the switch.**

```
ena mst
```

Root bridge selection for MSTP MSTIs

MSTP selects its root bridges for each MSTI. It does this by selecting, for each MSTI, the bridge that contains (numerically) the lowest bridge identifier. This is shown in Figure 1-2 on page 1-10.

Figure 1-2: Example MSTP MSTI configuration



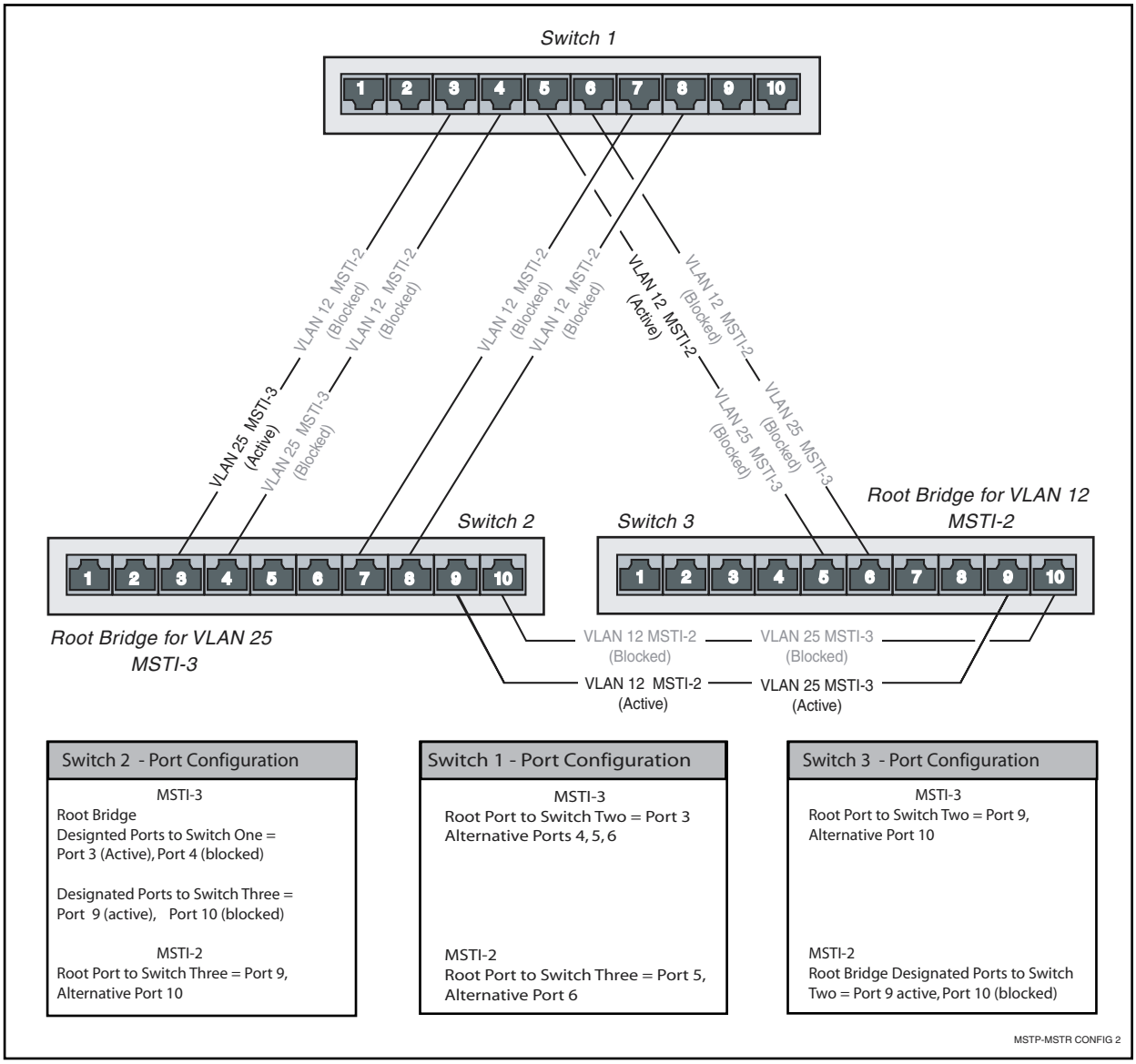
Notice that the root bridges are different for the two MSTIs. The root bridge for MSTI-2 is Switch 3 and the root bridge for MSTI-3 is Switch 2. This is because for MSTI-2 switch 3 it has been given the lowest MSTI priority value, 4096, compared with 8192 for Switches One and Two.

Similarly, the root bridge for MSTI-3 is Switch 2 because its MSTI priority value has been set to 4096, compared with the value 8192 set for Switches One and Two. If all three bridges were configured with the same priority value for a particular MSTI, then Switch 3 would become the root bridge for that MSTI, because it has the lowest MAC address of the three switches.

Path selection for MSTP MSTIs

Once the protocol has selected its root bridge for each MSTI, it selects which are to be the active and blocked paths for each MSTI. The port with the best path to the root bridge is selected as the root port and becomes active. Other ports that also lead to the root bridge but via a path that is better than the path back through the switch, are selected as alternate ports and are blocked to prevent loops. Ports that are connected to another port on the same switch where that port has a better priority value, are backup ports and are blocked to prevent a loop. Ports that are not disabled are selected as designated ports and are eventually made active. Figure 1-3 on page 1-11 shows which paths have been selected.

Figure 1-3: Example MSTP MSTI path configuration



For MSTI-3
Between switches One and Two there are two paths available, port 3 to port 3, and port 4 to port 4. Since no port priority has been explicitly applied, all port configurations have their defaults. Since all ports have the same speed (100 MBPS), each port has a Port Path Cost of 200,000. Since port 3 is numerically lower than port 4, the active path is the one between switch 1 port 3, and the other path is blocked. Similarly, the active path between switches Two and Three, is between port 9 on each switch.

For MSTI-2
Between switches One and Three there are two paths available, port 5 to port 5, and port 6 to port 6. Since no port priority has been explicitly applied, all port configurations have their defaults. Since all ports have the same speed (100 MBPS) each port has a Port Path Cost of 200,000. Since port 5 is numerically lower than port 6, the active path is the one between Switch One port 5 and Switch 2 port 5, and the other path is blocked. Similarly, the active path between switches Two and Three, is between port 9 on each switch.

If you want to make a particular path the active one, use the SET MSTP MSTI PORT command.

Example To balance the load between switches Two and Three, set the active path for MSTI-2 to be between ports 10 and 10 of each switch. Use the following command to set the port path cost less than the present default of 200,000:

For Switch 2

```
set mstp msti=2 port=10 pathcost=1000
```

For Switch 3

```
set mstp msti=2 port=10 pathcost=1000
```

Configuration Check

To check the status of the paths and to see which are forwarding and which are blocked, run the SHOW MSTP MSTI PORT command for a specific MSTI and port. From the output, note whether the port is a root and whether its status is forwarding or blocking. If the port is a root port and is in the forwarding state, then its path is active.

Common and Internal Spanning Tree (CIST)

In addition to the individual MSTIs within each MSTR region, the MSTI contains a network-wide spanning tree called the Common and Internal Spanning Tree (CIST). Conceptually, each region represents a virtual bridge. Internal and external bridge connectivity are two independent functions.

Frames with VIDs allocated to the CIST are subject to the rules and path costs of the complete bridged LAN as determined by the CIST's vectors. Frames other than these are subject to the CIST when travelling outside their region, and subject to its particular MSTI inside the region.

The following operational rules apply:

- Each bridge can be a member of only one region.
- A data frame is associated with a single VID.
- Data frames with a given VID are associated with either the CIST or their particular MSTI, but not both.

The configuration examples in this section are based on the network shown in Figure 1-4 on page 1-15. This simple network comprises six LAN bridges and is basically two networks of the type used in the previous examples, that are connected back to back.

Configuring the CIST Example

Configuring this network involves the same basic steps used in the previous examples. Note that the only VLAN that is common to both regions is VLAN 12, which uses MSTI 3. These must be explicitly configured to ports 1 and 10 of switches Three and Four.

For Switch 3

1. **Add VLAN 12 to the required ports, as tagged ports.**

```
add vlan=12 po=1,10 frame=tagged
set mstp msti=2 port=10 pathcost=1000
```

For Switch 4

1. **Add VLAN 12 to the required ports, as tagged ports.**

```
add vlan=12 po=1,10 frame=tagged
set mstp msti=2 port=10 pathcost=1000
```

If you configured the network using the steps in the previous example, and added the shared VLANs to the connecting ports as shown above, the network now has two regions: Region One representing a company's Head Office; and Region Two, representing the company's Manufacturing Plant. Note that although each network region is separate, with each of its MSTIs only having local significance within the region, the data itself still flows between the two networks and the VLANs in each are still recognised across MSTR boundaries.

The task of preventing loops within the wider network, is the role of CIST. By inspecting the example network, it is clear that there is a potential loop between the two regions that CIST must handle.

CIST first allocates root and designated bridges by selecting the bridge with the lowest identifier as the root. As far as the physical topology is concerned a good choice for the root bridge would be either of switches Three or Four. The network has been designed to force Switch 3 to become the root by assigning it the lowest priority identifier in the network (12288), and of course it is also the root bridge for Region One. Similarly, assigning Switch Four the priority identifier of 20480 ensures that this bridge becomes the root bridge for Region Two (because its priority identifier of 20480 is lower than any other bridge in its region). Switch Four is also the CIST regional bridge since it offers the lowest path cost from Region Two to Switch 3 (the CIST root bridge).

Note that the bridge identifier comprises two parts: a bridge priority part (more significant), and a bridge address part (less significant). The multiple spanning tree algorithm uses the bridge identifier when determining the role of a switch within each spanning tree. The switch with a lower priority is considered to have better bridge identifier, and is therefore more likely to be chosen as the root bridge. You can set the CIST bridge priority by using the command:

```
set mstp cist priority=20480
```

CIST vectors

Having selected the CIST root and designated bridge, CIST deals with loops between the regions. It does this by considering the following entities, called *vectors*, in the following order:

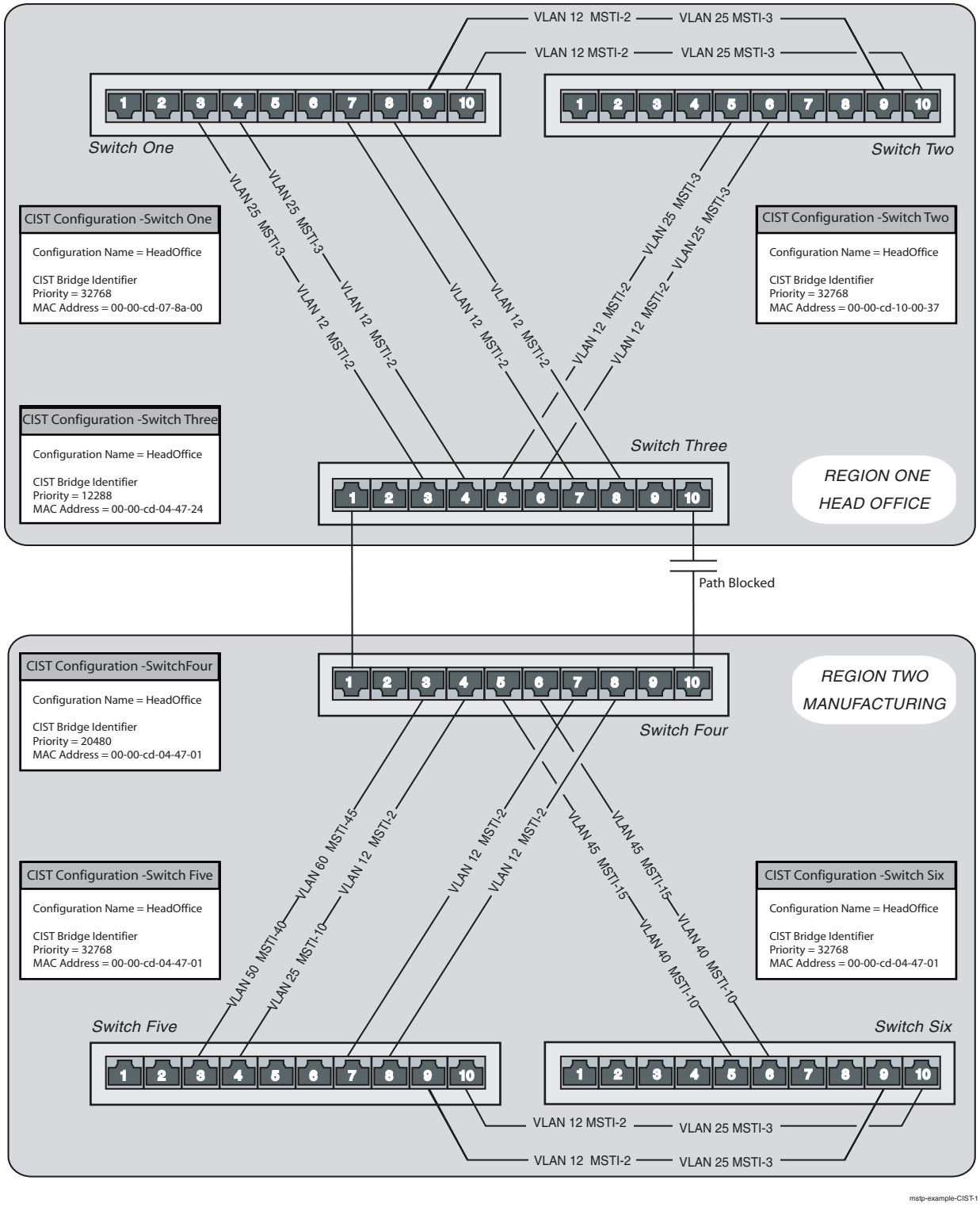
1. CIST External Root Path Cost
2. CIST Regional Root Identifier
3. CIST Internal Root Path Cost
4. CIST Designated Bridge Identifier
5. CIST Designated Port Identifier
6. CIST Receiving Port Identifier

Since there is clearly a loop condition between Switches Three and Four, CIST inspects each vector. Assuming the two links from the same bridge have equal path costs, the active link is selected as the one from the port with the lowest port number. Hence the path between ports 10 on each switch is blocked.

Note the situation if the connections on Switch Four were reversed, i.e. port 1 of Switch 3 being connected to port 10 of switch Four, and port 1 of switch Four being connected to port 10 of switch 3.

In the above situation, metric 5 above would apply (since metrics 1 through 4 have the same value). The designated ports would be 1 and 10 on switch 3, and since port 1 has the lower (numeric) value, this port provides the active link, and the path from its port 10 is blocked.

Figure 1-4: MSTP - CIST configuration example



The Relationship between Spanning Trees and Trunks

If multiple links are trunked together, either manually or automatically (by using a feature such as LACP), the spanning tree application is notified and considers the links to be a single logical path. Consequently, the spanning tree broadcast messages (BPDUs) only traverse the master trunk path.

Whether trunking offers a better solution depends on the individual network configuration. We recommend that both trunking and MSTP be considered so that the best option is selected based on requirements of the network.

Command Reference

This section describes the commands available to configure and manage switching functions on the switch.

The shortest valid command is denoted by capital letters in the Syntax section. See “Conventions” on page -cxxv of , Preface in the front of this manual for details of the conventions used to describe command syntax. See Appendix A, Messages for a complete list of messages and their meanings.

activate mstp migrationcheck port

Syntax `ACTivate MSTp MIGRationcheck POrt={`*`port-list`*`|ALL}`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description If an MSTP bridge detects the presence of STP data on one of its ports (from a legacy bridge) it automatically migrates the port to the STP protocol. Other MSTP and RSTP bridges connected to this port do the same. Thus all bridges that connect to this port revert to the STP protocol. However, this condition remains even after the original STP bridge has been removed.

Activating a migration check (mcheck) on such a port forces the bridge to migrate back to MSTP (or RSTP) and to transmit either MSTP (or RSTP) messages. After receiving these messages, other RSTP/MSTP bridges follow the same procedure. If no further STP bridge messages are received within a preset time period, then all the connected bridges remain in MSTP mode. The bridge decides whether to use RSTP or MSTP mode based on the setting of the PROTOCOLVERSION parameter of the MSTP command.

The PORT parameter specifies ports that are to have an mcheck applied to them. If ALL is specified, all ports in the switch are forced to the mcheck message. If the command would succeed on a subset of the ports specified, but cause an error on the others, then the command as a whole fails and has no effect.

The PORT parameter specifies the ports to transmit the mcheck messages. If ALL is specified, then all ports in the switch have an mcheck applied to them.

Example To transmit mcheck messages to all ports on the switch, use the command:

```
act mst po=all
```

Related Commands **show mstp**

add mstp msti vlan

Syntax `ADD MSTp MSTI=`*`instance`* `VLAN={1..4094|ALL}`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command maps one or more VLANs to a specific Multiple Spanning Tree Instance (MSTI). The MSTP algorithm provides multiple spanning tree topologies within one MSTR so different VLANs can be forwarded in different paths.

All of the VLANs are mapped to the Common and Internal Spanning Tree (CIST) by default. After a VLAN is mapped to a specific MSTI, it is removed from the CIST.

A VLAN can be mapped to only one MSTI or the CIST. One VLAN cannot be mapped to multiple spanning trees. A VLAN must be removed from one MSTI before it can be mapped to another. VLANs follow the CIST when operating between regions.

The MSTI parameter specifies the instance number of the spanning tree. The MSTI must already exist before a VLAN can be mapped to it. The **create mstp msti** command on page 1-19 creates an MSTI.

The VLAN parameter specifies a VLAN to be mapped to a specific MSTI. If ALL is specified, then all VLANs are mapped to the MSTI. If a VLAN is already mapped to an MSTI other than the one specified in the command, then the command fails.

Examples To map a VLAN with VID of 1 to MSTI5, use the command:

```
add mst msti=5 vlan=1
```

Related Commands
delete mstp msti vlan
show mstp
show mstp msti

create mstp msti

Syntax `CREate MSTp MSTI=`*`instance`* `[PRIOrity=0..65535]`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command creates a new multiple spanning tree instance (MSTI) on the switch. The multiple spanning tree algorithm enables a collection of VLANs to be associated with a particular spanning tree instance. Within this instance, frames belonging to this VLAN group are forwarded over the active topology established by that particular instance's spanning tree. Frames for VLAN groups belonging to other instances each have their own active topologies.

Once an MSTI has been successfully created, VLANs can be added to it by using the command ADD MSTP MSTI VLAN.

Within each MST region, the MSTP maintains multiple spanning tree instances (MSTIs). A unique instance number identifies each single MSTI.

The MSTI parameter specifies the instance number of the multiple spanning tree instance (MSTI) being created. Although numbers can be assigned within the range 1 to 4094, the maximum number of MSTIs within each region, or switch, is 64. Instance number 0 is reserved for the common internal spanning tree (CIST) instance.

The MSTI number is very useful because it identifies a particular instance within an MST region.

The PRIORITY parameter sets the value of the priority field contained in the bridge identifier. The bridge identifier comprises two parts: a bridge priority part (more significant), and a bridge address part (less significant). The multiple spanning tree algorithm uses the bridge identifier when determining the role of a switch within each spanning tree. The switch with a lower priority is considered to have better bridge identifier, and is therefore more likely to be chosen as the root bridge. The CIST and each MSTI have their own individual PRIORITY parameter, so the roles of the same switch could be different in the CIST and each MSTI by tuning the bridge priority. The priority value operates in multiples of 4096. If you specify a value that is not a multiple of 4096, it is rounded down to the nearest multiple of 4096 (Table 1-4 on page 1-20).The default is 32768.

Table 1-4: Rounding scheme for ranges of bridge priority parameter values .

Lower Boundary	Upper Boundary	Rounded Bridge Value
0	4095	0
4096	8191	4096
8192	12287	8192
12288	16383	12288
16384	20479	16384
20480	24575	20480
24576	28671	24576
28672	32767	28672
32768	36863	32768
36864	40959	36864
40960	45055	40960
45056	49151	45056
49152	53247	49152
53248	57343	53248
57344	61439	57344
61440	65535	61440

Example To create a new MSTI 5 with a priority of 8192, use the command:

```
cre mst msti=5 prio=8192
```

Related Commands

destroy mstp msti
show mstp
show mstp msti

delete mstp msti vlan

Syntax `DELete MSTp MSTI=`*`instance`* `VLAN={1..4094|ALL}`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command removes VLANs from a specific MSTI. The removed VLANs are mapped to the CIST.

Once a VLAN is unmapped from a specified MSTI, the frames belonging to that VLAN are not longer forwarded along the spanning tree associate with that instance. The frames are forwarded along the CIST spanning tree.

The MSTI parameter specifies the instance number of the specified Multiple Spanning Tree Instance. Any VLANs that are not explicitly assigned to a specific MSTI are mapped to the CIST by default. There is no command to remove VLANs from the CIST.

The VLAN parameter specifies the VLAN mapped to a specific MSTI. To unmap a VLAN from an MSTI it must have previously been mapped to the MSTI. If ALL is specified, all VLANs mapped to the MSTI are unmapped and remapped to the CIST.

Examples To delete the mapping of all VLANs from MSTI5, use the command:

```
del mst msti=5 vlan=all
```

Related Commands
add mstp msti vlan
show mstp
show mstp msti

destroy mstp msti

Syntax `DESTroy MSTp MSTI=`*`instance`* `[PRIOrity=0..65535]`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command destroys a specific multiple spanning tree instance (MSTI) on the switch. An MSTI cannot be destroyed if it still has VLANs mapped to it. Use the **delete mstp msti vlan=all** command to remove all VLANs from the specified MSTI.

Example To destroy an existing MSTI5, use the command:

```
dest mst msti=5
```

Related Commands
delete mstp msti vlan
show mstp
show mstp msti

disable mstp

Syntax `DISable MSTp`

Description This command disables the multiple spanning tree operation on the switch. By default, MSTP is disabled on switch start-up. This command overrides the following commands:

enable mstp cist port
disable mstp cist port
enable mstp msti port
enable mstp msti port

Once MSTP has been disabled, no port for the CIST or MSTIs can be enabled or disabled. MSTP must be disabled before STP instances can be enabled.

Examples To enable MSTP, use the commands:

```
dis mst
```

Related Commands
enable mstp
show mstp
disable mstp

disable mstp cist port

Syntax `DISable MSTp CIST POrt={`*`port-list`*`|ALL}`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description This command disables operation of the Multiple Spanning Tree algorithm on specific ports or all ports for the Common Internal Spanning Tree. Disabled ports are placed in a discarding state and cannot forward frames. All ports are enabled for the CIST by default.

The MSTP module must be enabled first before any port for the CIST can be enabled or disabled.

The PORT parameter specifies a list of ports to be disabled for the CIST. If **all** is specified, all ports on the switch are disabled for the CIST. If the command would succeed on a subset of the ports specified, but cause an error on the others, then the command as a whole fails and has no effect.

Example To disable port 2 in the CIST, use the command:

```
dis mst cist po=2
```

Related Commands
show mstp cist
show mstp cist port

disable mstp debug msti

Syntax

```
DISable MSTp DEBug={Msg|Pkt|State|ALL} MSTI={CIST|
   instance|ALL} [POrt={port-list|ALL}]
```

where:

- *instance* is an instance number from 1 to 4094 for a specific MSTI.
- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description

This command disables MSTP debugging for a specified MST instance (MSTI) or all instances, or on specific ports.

The MSTI parameter specifies the instance for which the debugging mode is disabled. If CIST is specified, then debug is disabled on the CIST. If an instance is specified, then debug is disabled on the MSTI. If ports are specified using the PORT parameter, then debug is disabled on the specified port on the specified instance. If ALL is specified and the ports are specified using the PORT parameter, then debug mode is disabled on all the instances for the listed ports.

The DEBUG parameter specifies which debugging modes are to be disabled. If ALL is specified, then all debugging modes for the instances or ports are disabled. The other modes can be disabled independently of each other.

The PORT parameter specifies the ports where the debug mode is disabled.

Example

To disable debugging on all ports in MSTI5, use the command:

```
dis mst msti=5 po=all
```

Related Commands

show mstp msti
show mstp msti port

disable mstp msti port

Syntax `DISable MSTp MSTI=`*`instance`* `POrt={`*`port-list`*`|ALL}`

where:

- *instance* is an instance number from 1 to 4094 for a specific MSTI.
- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports

Description This command disables operation of the Multiple Spanning Tree algorithm on specific ports or on all ports for a specific Multiple Spanning Tree Instance. Disabled ports are placed in a discovery state and cannot forward frames.

MSTP must be enabled before ports for a specific MSTI can be enabled or disabled.

The MSTI parameter specifies the instance number for the specified MSTI.

The PORT parameter specifies a list of ports to be disabled for a specific MSTI. If **all** is specified, all ports on the switch are disabled for the specified MSTI.

Example To disable port 2 in MSTI5, use the command:

```
dis mat msti=5 po=2
```

Related Commands **show mstp msti**
show mstp msti port

enable mstp

Syntax `ENAble MSTp`

Description This command enables the operation of the multiple spanning tree algorithm on the switch. Multiple spanning tree protocol (MSTP) enables a number of VLANs to each use separate active topologies throughout a virtual bridged LAN. By default, MSTP is disabled at switch start-up. MSTP must be enabled before the following commands can be used:

enable mstp cist port
disable mstp cist port
enable mstp msti port
disable mstp msti port

After MSTP is enabled, any port for the CIST and the existing MSTIs can be enabled or disabled. Enabling MSTP initialises the status for the switch and all its ports. MSTP cannot be enabled while STP instances are enabled; all STP instances must be disabled.

Examples To enable MSTP, use the command:

```
ena mst
```

Related Commands **disable mstp**
show mstp

enable mstp cist port

Syntax `ENAble MSTp CIST POrt={`*`port-list`*`|ALL}`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports

Description This command enables operation of the MSTP algorithm on a specific port or all ports for the CIST. This command fails when a port is a member of a trunk group but is not the master port,

MSTP must be enabled before ports for the CIST can be enabled or disabled.

The PORT parameter specifies a list of ports to be enabled for the CIST. If **all** is specified, all ports on the switch are enabled. If the command would succeed on a subset of the ports specified, but cause an error on the others, then the command as a whole fails and has no effect.

If a port has been disabled with the **disable switch port** command, or has a link status of down, and the port is enabled, a message is displayed indicating the condition.

All ports are enabled for the CIST by default.

Example To enable all ports in the CIST, use the command:

```
ena mst cist po=all
```

Related Commands **show mstp cist**
show mstp cist port

enable mstp debug

Syntax

```
ENAble MSTp DEBug={Msg|Pkt|State|All} MSTI={CIST|instance|
   ALL} [POrt={port-list|ALL}] [Statemachine={PTM|PRX|PPM|
   PIM|PTX|PRS|PRT|PST|TCM|ALL}] [Output=Console]
   [Timeout=1..4000000000|NONE]
```

where:

- *instance* is an instance number from 1 to 4094 for a specific MSTI.
- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description

This command enables MSTP debugging for a specific MSTP instance or all instances, or on a specific port or all ports.

The MSTI parameter specifies the spanning tree instance that is to have its debugging mode enabled. If CIST is specified, then debugging is enabled on the CIST. If an MSTI is specified, then debugging is enabled on the specific MSTI. If ports are specified with the PORT parameter, then the debug is enabled on the specified port on the specified instance. If ALL is specified and the ports are specified with the PORT parameter, then debugging mode for the listed ports is enabled on all the instances with the listed ports.

The instance number is for the specified MSTI.

The DEBUG parameter specifies which debugging modes are to be enabled. If ALL is specified, then all debugging modes for the instances or ports are enabled. Other modes can be enabled independently of each other. The following table lists options for this parameter.

Option	Description
MSG	Decoded display of BPDUs received and transmitted by MSTP
PKT	Raw ASCII display of BPDUs received and transmitted BY mstp
STATE	Port state transitions. For MSTP states for state machines specified by the STATEMACHINE parameter are displayed
ALL	All debug options

Setting the OUTPUT parameter to CONSOLE instructs the bridge to send the debugging information to the console. By default, the debugging data is sent to the port that received the ENABLE MSTP DEBUG command. This option should be selected if the ENABLE MSTP DEBUG command is used in a script, because a script is not received on a port.

The PORT parameter specifies which ports on the bridge are to have the debug mode enabled. If port value is not entered, the parameter defaults to ALL.

The STATEMACHINE parameter specifies which state machines are to have debugging enabled This parameter is valid only when the debug mode is STATE. The default is ALL. The following table lists options for this parameter and the corresponding state machine.

Option	Description
PTM	Port timer state machine
PRX	Port receive state machine
PPM	Port protocol migration state machine
PIM	Port information state machine
PTX	Port transmit state machine
PRS	Port role selection state machine
PRT	Port role transitions state machine
PST	Port state transition state machine
TCM	Topology change state machine

This parameter is cleared only when the **disable mstp debug** command specifies the DEBUG parameter as either STATE or ALL—when debug mode is neither of these, the STATEMACHINE parameter is not cleared.

The TIMEOUT parameter specifies the time period in seconds during which debugging is enabled on the specified ports. Limiting the debugging time period reduces the risk of the switch and the display being overloaded with debugging information. Note that this value overrides previous MSTP debugging timeout values for these ports, even if they were specified for other debugging modes. If a value is not specified, the default is NONE. When the timeout expires, the following events occur:

- OUTPUT is redirected to the console
- DEBUG is disabled for all modes
- STATEMACHINE modes are all disabled
- TIMEOUT is set to NONE

Example To enable debugging on all ports in MSTI5, use the command:

```
ena mst msti=5 po=all
```

Related Commands

disable mstp debug msti
show mstp debug msti

enable mstp msti port

Syntax `ENAble MSTp MSTI=`*`instance`* `POrt={`*`port-list`*`|ALL}`

where:

- *instance* is an instance number from 1 to 4094 for a specific MSTI.
- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description This command enables operation of the Multiple Spanning Tree algorithm on the specified ports or all ports for the specified Multiple Spanning Tree Instance. If a port is a member of a trunk group but is not the master port, then this command fails.

The MSTP module must be enabled first before any port for the specified MSTI can be enabled or disabled.

The MSTI parameter specifies the instance number for the specified MSTI.

The PORT parameter specifies a list of ports to be enabled for a specific MSTI. If ALL is specified, all ports on the switch are enabled it. If the command would succeed on a subset of the ports specified, but cause an error on the others, then the command as a whole fails.

If a port is disabled with the **disable switch port** command, or has a link status of *down* and the port is enabled, a message is displayed indicating the condition.

All ports are enabled for the specified MSTI by default.

Example To enable all port in MSTI5, use the command:

```
ena mst msti=5 po=all
```

Related Commands **show mstp msti**
show mstp msti port

purge mstp

Syntax `PURge MSTp`

Description This command purges all configuration information relating to MSTP. All user-created MSTIs are destroyed. All VLANs are mapped to the CIST. It restores the defaults to all the configurable parameters. This command returns the MSTP module to its status when the switch was first powered on.

After the MSTP configuration is purged, MSTP is disabled and returned to the initialised status.

Use with extreme caution because all current configurations will be lost.

Example To purge the MSTP configuration, use the command:

```
pur mst
```

Related Commands **show mstp**
show mstp msti

reset mstp counter port

Syntax `RESET MSTp COUnter POrt={`*`port-list`*`|ALL}`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description This command resets counters for specific ports.

The PORT parameter specifies the ports. If ALL is specified, all port counters in the switch are reset. The default is ALL.

Example To enable all port in MSTI5, use the command:

```
reset mst cou po=1
```

Related Commands **reset mstp counter port**

set mstp

Syntax

```
SET MSTp [CONFigname=name] [REVIsionlevel=level]
   [MAXHops=1..40] [MAxage=6..40] [HEllotime=1..10]
   [FOrwarddelay=4..30] [PROTocolversion={STP|RSTp|MSTp}]
   [STATicvlans={YES|NO|ON|OFF|True|False}]
```

where:

- *name* is the MST configuration name. It is a string of up to 32 characters, valid characters being uppercase and lowercase letters, digits, the underscore, and the hyphen.
- *level* is the MST configuration revision level with a range from 0 to 65535.

Description

This command sets the MST configuration identifier values and the state machine performance parameters. The configuration identifier contains:

- the configuration name
- the revision number
- a digest of the VLAN to MSTI configuration table

The state machine performance parameters are constants used by the CIST and MSTI state machines.

When the MST algorithm calculates the active topology, it does not consider the VLAN membership of the ports. It does not need to because IEEE Standard 802.1Q-2003 assumes that the active topology is determined first and that the VLANs are configured dynamically over the active topology, via GVRP. GVRP configures the VLAN memberships of ports so that frames belonging to a VLAN can traverse the spanning tree (CIST or MSTI) to which the VLAN is assigned.

The process is reversed when statically configured VLANs are used. The VLAN memberships of ports are configured statically and then the active topology is calculated. However, the MST algorithm does not consider the VLAN memberships when calculating the active topology. It may choose a port that is not a member of any of the VLANs assigned to the spanning tree to be the root port, even though an alternate port that is a member of the VLANs may exist. This would partition the network, preventing frames belonging to a VLAN assigned to the spanning tree from traversing the network. In this situation it is desirable that the algorithm considers the VLAN memberships of ports and prevents partitioning where possible. It should choose the root port from the ports that are members of the VLANs assigned to the spanning tree.

When using statically configured VLANs, each VLAN assigned to a given spanning tree should have the same port membership; otherwise, partitioning may occur.

The MST configuration identifier determines the MST region where a switch belongs. The MST configuration identifier is conveyed in the MSTP BPDUs so the switch can check whether it is allocating VIDs to the same spanning tree instance as a neighbouring switch. If the configuration identification of two switches matches, they are from the same MST region.

MSTP assigns the switch a default MST configuration identification consisting of a unique default configuration name and a default revision level.

The CONFIGNAME parameter specifies the name for the MST region. Switches in the same MST region have the same configuration name. If the configuration name is not set explicitly by the command, the default name for the MST region is the switch's MAC address presented as a text string. By default all switches are in their own MST region because MAC addresses are unique.

The REVISIONLEVEL parameter specifies the revision level in the MST region. All the switches in the same MST region have the same revision number. If the revision level is not set explicitly by the command, the default is 0.

The FORWARDDELAY parameter sets the number of seconds that a port waits before changing its spanning tree state towards the forwarding state. Its purpose is to allow sufficient time for other ports to receive their spanning tree information. The delay determines the maximum time taken to transition from discarding to learning and from learning to forwarding. This value is used only when the switch is acting as the root bridge. Any switch not acting as the root bridge uses a dynamic value for the FORWARDDELAY set by the root bridge. The FORWARDDELAY, MAXAGE, and HELLOTIME parameters are interrelated (see formulas below). The default is 15 seconds.

The HELLOTIME parameter sets the time period, in seconds, between the transmissions of spanning tree configuration messages. These messages are transmitted by ports with the 'designated port' role of the spanning tree, or are trying to become the root bridge. The default is 2 seconds.

The MAXAGE parameter sets the maximum age, in seconds, that dynamic MSTP configuration information stored in the switch may reach before it is discarded. The default is 20 seconds.

The FORWARDDELAY, MAXAGE, and HELLOTIME parameters should be set according to the following formulae, as specified in IEEE Standard 802.1d:

2 x (FORWARDDELAY - 1.0 seconds) >= MAXAGE

MAXAGE >= 2 x (HELLOTIME + 1.0 seconds)

The MAXHOP parameter specifies the maximum hop count in transmitting information within an MST region. This is in order to ensure that old information does not endlessly circulate through redundant paths in the network, thus preventing the effective propagation of the new information. The hop count is decremented by each receiving port. Received information is discarded and the port is made a designated port if the hop count reaches 0. The default for MAXHOP is 20.

The PROTOCOLVERSION parameter specifies which version of the spanning tree protocol the switch uses.If MSTP is specified, the switch uses the full Multiple Spanning Tree protocol and sends MSTI BPDUs. If RSTP is specified, the switch uses the Rapid Spanning Tree protocol and sends RST BPDUs. The switch operates as though it is in a region of its own. If STP is specified, the switch emulates the Spanning Tree Protocol and transmits STP configuration BPDUs. Rapid port state transitions are disabled, and the switch operates as if in a region of its own.

The STATICVLANS parameter should be turned on when the ports that link to other switches have static VLAN memberships. In simple static VLAN configurations it may be possible to operate with this option turned off provided that redundant links between any pair of switches have the same VLAN memberships. If VLANs are being configured dynamically with GVRP,

the STATICVLANS parameter should be set to OFF (NO, or FALSE). The default is OFF.

Example To set MST configuration name to mstRegion1 and the revision level to 10, use the command:

```
set mst conf=mstRegion1 revi=10
```

To set forward delay time to 20 seconds and max hop count to be 25, use the command:

```
set mst fo=20 maxh=25
```

To set STATICVLANS to be true, use the command:

```
set mst stat=t
```

To set hello time to be 2 seconds and max message age to be 30 seconds, use the commands:

```
set mst he=2 ma=30
```

Related Commands **show mstp**

set mstp cist

Syntax `SET MSTp CIST [PRIOrity=0..65535]`

Description This command sets parameters used by the MSTP algorithm to calculate the Common and Internal Spanning Tree (CIST). The bridge level parameters of the CIST can be modified to force the spanning tree configuration or to tune its topology.

The PRIORITY parameter sets the priority of the switch to become the root bridge in the CIST. The lower the value, the better the bridge identifier and the more likely it will be selected as the root. Although any value between 0 and 65,535 can be specified, the protocol requires the priority to be multiples of 4096; therefore, values are rounded down to the nearest multiple of 4096 (Table 1-4 on page 1-20). The default is 32768.

Example To set CIST to a priority of 8192, use the command:

```
set mst cist prio=8192
```

Related Commands **show mstp cist**
show mstp

set mstp cist port

Syntax

```
SET MSTp CIST POrt={port-list|ALL} [PRIOrity=0..255]
   [EXTPathcost=extPathCost] [INTPathcost=intPathCost]
   [EDGEport={YES|NO|ON|OFF|True|False}]
   [POINttopoint={YES|NO|ON|OFF|True|False|Auto}]
```

where:

- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.
- *extPathcost* is a value from 1 to 200,000,000.
- *intPathcost* is a value from 1 to 200,000,000.

Description

This command sets the Common and Internal Spanning Tree (CIST) tuning parameters for specific ports. Modifying parameters for a switch port forces the CIST port rules to be recalculated.

Parameters assigned to specific ports affect the network topology of only the CIST, and do not affect the topology of other spanning tree instances on the switch.

The PORT parameter specifies a list of ports to be configured for the CIST. If ALL is specified, then all ports are configured according to the new parameters for the CIST.

The PRIORITY parameter sets the value of the priority field contained in the port identifier. The MST algorithm uses the port priority when determining the root port for the switch in the CIST. The port with the lowest value is considered to have the highest priority and is chosen as root port over a port—equivalent in all other aspects—but with a higher priority value. Any value from 0 to 255 can be entered, but the switch rounds it down to the nearest multiple of 16 (for example, if 17 is entered, 16 is used). The default is 128. See Table 1-5 on page 1-37.

The EXTPATHCOST parameter sets the external path cost for the ports. This parameter specifies a port's contribution to the cost of a path to the region containing the CIST root via that port. It applies when the port is a root port.

The INTPATHCOST parameter sets the internal path cost for the ports. This parameter specifies a port's contribution to the cost of a path to the CIST regional root via that port. It applies when the port is a root port.

If the EXTPATHCOST or INTPATHCOST is not explicitly set, or the defaults have been restored to the port, then the default for the port varies as the speed of the port varies, See Table 1-6 on page 1-38. However, deleting an existing EXTPATHCOST or INTPATHCOST value does not re-apply the "no value" condition; to re-apply the condition, enter the word "default."

The EDGEPORT parameter specifies whether the port is an edge port. An edge port is a one that attaches to a LAN that is known to have no other bridges attached. If NO is specified, then the port is not an edge port. The values NO, OFF and FALSE are equivalent. If YES is specified, then the port is an edge port. The values YES, ON, and TRUE are equivalent.

If EDGEPORT is set to YES and an MSTP BPDU is received on the port, indicating that another bridge is connected to the LAN, then the port is no longer treated as an edge port. The default is NO. Edge ports are permitted to make rapid transitions to the forwarding state because they are not connected to another bridge and therefore cannot form part of a network loop. Edge ports that are not configured as such must make slow transitions to the forwarding state. For optimal convergence, all edge ports should be identified and have EDGEPORT set to Yes. A port should be set to edge port only when it connects to a single end station.

The POINTTOPOINT parameter specifies whether the port has a point-to-point connection to another bridge. If AUTO is specified, then the status of point-to-point link is determined automatically by the switch. If YES is specified, then the port is treated as a point-to-point LAN segment. The values YES, ON and TRUE are equivalent. If NO is specified, then the port is not be treated as a point-to-point LAN segment. The values NO, OFF and FALSE are equivalent. If the port is considered as a point-to-point port, then it is permitted to make rapid transitions to the forwarding state, providing it receives an agreement message from the bridge at the other end of the segment. A port should be set to point-to-point only when it connects exactly one other bridge. The default is AUTO.

Example To set port priority of 16 for port 2 in the CIST, use the command:

```
set mst cist po=2 prio=16
```

To set external port path cost of 120 for port 2 in the CIST, use the command:

```
set mst cist po=2 extp=120
```

To set internal port path cost of 200 for port 2 in the CIST, use the command:

```
set mst cist po=2 intp=200
```

To set port 2 in the CIST as edge port, use the command:

```
set mst cist po=2 edge=yes
```

To set port 2 in the CIST as point to point link, use the command:

```
set mst cist po=2 poin=yes
```

Related Commands **show mstp cist**
set mstp cist port

set mstp msti

Syntax `SET MSTp MSTI=`*`instance`* `[PRIOrity=0..65535]`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command sets parameters used by the Multiple Spanning Tree algorithm to calculate the spanning tree for a specific MSTI. The bridge level parameters of the MSTI can be modified to tune the spanning tree topology.

The MSTI parameter specifies the instance number for the specified Multiple Spanning Tree Instance.

The PRIORITY parameter sets the priority of the switch to become the root bridge in a specific MSTI. The lower the value, the better the bridge identifier and the more likely the bridge will be selected as a root bridge. Although any value between 0 and 65535 can be specified, the switch processes values that are multiples of 4096; therefore, values are rounded down to the nearest multiple of 4096 (Table 1-4 on page 1-20). The default is 32768.

Example To set the priority to 8192 for MSTI5, use the command:

```
set mst msti=5 prio=8192
```

Related Commands **show mstp**
show mstp msti

set mstp msti port

Syntax

```
SET MSTp MSTI=instance POrt={port-list|ALL}
   [PRIOrity=0..255] [PAthcost=pathCost]
```

where:

- *instance* is an instance number from 1 to 4094 for a specific MSTI.
- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.
- *pathcost* is a value from 1 to 200000000.

Description

This command sets tuning parameters for a specific port or all ports for a specific Multiple Spanning Tree Instance (MSTI). Modifying parameters for a port forces a recalculation of the port roles for the specified MSTI.

Parameters assigned for a specific port affect only the network topology of the specified MSTI, and not other spanning tree instances on the switch.

The MSTI parameter specifies the instance number for the selected MSTI.

The PORT parameter specifies a list of ports to be configured for a specific MSTI. If ALL is specified, all ports are configured according to the new values for the specified MSTI.

The PRIORITY parameter sets the value of the priority field contained in the port identifier. The MSTP algorithm uses the port priority when determining the root port for the switch in the specified MSTI. The port with the lowest value is considered to have the highest priority and is chosen as root port over a port—equivalent in all other aspects—but with a higher priority value. Any value from 0 to 255 can be entered, but the switch rounds it down to the nearest multiple of 16 (for example, if 17 is entered, 16 is used). The default is 128 (Table 1-5 on page 1-37).

Table 1-5: Rounding scheme for ranges of port **priority** parameter values

Lower Boundary	Upper Boundary	Rounded Port Priority Value
0 0	15	0
16	31	16
32	47	32
48	63	48
64	79	64
80	95	80
96	127	96
128	143	128
144	159	144
160	175	160
176	191	176
192	207	192
208	223	208
224	239	224
240	254	240

The PATHCOST parameter sets the internal path cost for the each port. This parameter specifies a port's contribution to the cost of a path to the MSTI regional root via that port. It applies when the port is a root port. The PATHCOST for a LAN port should be from 1 to 200000000. The defaults and range of recommended values depend on the port speed.

If the PATHCOST of a port has not been explicitly set by the user, or the defaults have been restored to the port, then the default PATHCOST for the port varies as the speed of the port varies. However, deleting an existing PATHCOST value does not reapply the "no value" condition; to reapply the condition, enter the word "default".

Table 1-6: Path cost values and port speed

Port Speed	Default PATHCOST	Recommended PATHCOST range
Less than 100 Kb/s	200,000,000	20,000,000-200,000,000
1Mbps	20,000,000	2,000,000-20,000,000
10Mbps	2,000,000	200,000-2,000,000
100 Mbps	200,000	20,000-200,000
1 Gbps	20,000	2,000-20,000
10 Gbps	2,000	200-2, 000
100 Gbps	200	20-200
1Tbps	20	2-200
10 Tbps	2	2-20

Example To set port priority of 120 for port 2 in MSTI5, use the command:

```
set mst msti=5 po=2 prio=120
```

To set port path cost of 200 for port 2 in MSTI5, use the command:

```
set mst msti=5 po=2 pa=120
```

Related Commands **show mstp msti**
enable mstp msti port

show mstp

Syntax `SHow MSTp [CONfigid] [TAble]`

Description This command displays information about MSTP (Figure 1-5 on page 1-39, and Table 1-7 on page 1-39).

If the CONFIGID parameter is specified, the MST Configuration Identification is displayed as shown in Figure 1-6 on page 1-40, and Table 1-8 on page 1-40.

If the TABLE parameter is specified, the MST Configuration Table that contains the map between MSTIs and VLANs is displayed as shown in Figure 1-7 on page 1-40, and Table 1-9 on page 1-40.

Figure 1-5: Example output from the **show mstp** command

```
MSTP Information
------------------------------------------------------------
MSTP Status ............................. Enabled
MST Configuration Name .................. mstRegion1
MST Revision Level ...................... 0
Number of MSTIs ......................... 10
Hello Time .............................. 2
Forward Delay ........................... 15
Max Message Age ......................... 100
Max Hops ................................ 5
Protocol Version ........................ MSTP
Support Static VLANs .................... Enabled
Transmission Limit ...................... 3
Migrate Time ............................ 8
------------------------------------------------------------
```

Table 1-7: Parameters in output of the **show mstp** command

Parameter	Description
MSTP Status	Whether MSTP is enabled.
MST Configuration Name	Name of the MST region.
MST Revision Level	Revision level of the MST region.
Number of MSTIs	Number of Multiple Spanning Tree instances.
Protocol Version	Spanning Tree Protocol version: STP, RSTP, or MSTP.
Max Hops	Maximum hop count in transmitting information within an MST region
Transmission Limit	Number of bridge protocol messages (BPDUs) that may be transmitted in the interval specified by Hello Time
Migrate Time	A constant timer value used as the initial value of the migration delay. The value of Migrate Time is 3 seconds
Hello Time	The seconds between transmissions of spanning tree configuration information (BPDUs)
Forward Delay	Number of seconds that controls how fast a port changes its spanning tree state when moving towards the forwarding state
Max Message Age	Maximum age of received bridge protocol message (BPDU) information before it is discarded
Support Static VLAN	Whether a supporting static VLAN configuration is enabled.

Figure 1-6: Example output from the **show mstp configid** command

```
MST Configuration Identification
-------------------------------------------------------------
Configuration Name ..... mstRegion1
Format Selector ......... 0
Revision Level .......... 12
Configuration Digest .... AC36177F50283CD4B83821D8AB26D8AB
-------------------------------------------------------------
```

Table 1-8: Parameters in output of the **show mstp configid** command

Parameter	Description
Configuration Name	Name of the MST region
MST Configuration Name	A Configuration Identifier Format Selector
MST Revision Level	Revision level of the MST region
Configuration digest	A 16-octet signature of type HMAC-MID5 created from the MST Configuration Table

Figure 1-7: Example output from the **show mstp table** command

```
MST Configuration Table
-------------------------------------------------------
Multiple Spanning Tree Instance        VLAN Members
-------------------------------------------------------
CIST                                   15-19,31-4094
MSTI 1                                 1,2,10,20-30
MSTI 2                                 3-9
MSTI 3                                 11-14
-------------------------------------------------------
```

Table 1-9: Parameters in output of the **show mstp cist** command

Parameter	Description
Multiple Spanning Tree Instance	Whether the instance of a spanning tree is CIST or MSTI.
VLAN Members	A list of the VLANs that are mapped to a specified MSTI.

Example To show the information of MSTP, use the command:

```
sh mst
```

Related Commands

enable mstp
disable mstp
destroy mstp msti
add mstp msti vlan
delete mstp msti vlan
set mstp
set mstp cist
set mstp msti
show mstp cist

show mstp cist

Syntax `SHow MSTp CIST`

Description This command displays the information about the Common Internal Spanning Tree (Table 1-8 on page 1-41).

Figure 1-8: Example output from the **show mstp cist** command

```
Common Internal Spanning Tree
--------------------------------------------------------------
Bridge Identifier.................32768 : 00-00-cd-05-19-28
Bridge Role.......................Root Bridge
VLAN Members......................1, 2-10, 20
CIST Root Bridge.....  .....32768 : 00-00-cd-05-19-28
CIST Regional Root Bridge.........32768 : 00-00-cd-05-19-28
Designated Bridge.................32768 : 00-00-cd-05-19-28
Root Port.........................N/A
External Root Path Cost...........0
Internal Root Path Cost...........0
Performance:
  Max Age.........................20
  Hello Time......................2
  Forward Delay...................20
  Max Hops........................5
  Bridge Max Age..................20
  Bridge Hello Time...............20
  Bridge Forward Delay............20
  Bridge Max Hops.................20
  Transmission Limit..............3
Topology Changes:
  Time Since Topology Change......100
  Topology Change Count...........3
  Topology Change.................FALSE
--------------------------------------------------------------
```

Table 1-10: Example output from the **show mstp cist** command

Parameter	Meaning
Bridge Identifier	The unique bridge identifier of the switch. This parameter consists of two parts, one part is derived from the switch's unique MAC Address, and the other part is the priority value entered for the switch.
Bridge Role	Whether the role of the bridge in the CIST is root bridge, regional root bridge, or designated bridge.
VLAN Members	A list of the VLANs that are mapped to the Multiple Spanning Tree Instance specified.
CIST Root Bridge	Bridge identifier of the CIST Root of the bridged local area network.
CIST Regional Root Bridge	Bridge identifier of the root bridge for the CIST in an MST region (MSTR).
Designated Bridge	Bridge identifier of the bridge through which the root bridge may be reached from this device.

Table 1-10: Example output from the **show mstp cist** command (Continued)

Parameter	Meaning
Root Port	Port number of the root port for the switch. This parameter is not valid when the switch is the root bridge and output is shown as N/A.
External Root Path Cost	Path cost to the region containing the CIST root from this region.
Internal Root Path Cost	Path cost to the CIST Regional Root.
Max Age	Maximum age of received bridge protocol message (BPDU) information before it is discarded.
Hello Time	Seconds between transmissions of spanning tree configuration information (BPDUs)
Forward Delay	Maximum time taken to transition from the discarding state to the learning state, and from the learning state to the forwarding state.
Max Hops	Maximum hop count within an MST region for CIST information transmitted from this switch.
Bridge Max Age	Value of the max age parameter when the switch is either the root or is attempting to become the root. This parameter is set by the MAXAGE parameter in the SET MSTP command.
Bridge Hello Time	Value of the Hello Time parameter when the switch is the root or is attempting to become the root. This parameter is set by the HELLOTIME parameter in the SET MSTP command.
Bridge Forward Delay	Value of the Forward Delay parameter when this switch is the root or is attempting to become the root. This parameter is set by the FORWARDDELAY parameter in the SET MSTP command.
Bridge Max Hops	Value of the Max Hops parameter when the switch is either the root or is attempting to become the root. This parameter is set by the MAXHOPS parameter in the SET MSTP command.
Transmission Limit	Number of BPDUs that may be transmitted in the interval specified by Hello Time parameter. The value of this fixed parameter is 3.
Time Since Topology Change	The count in seconds of the time elapsed since the last topology changed.
Topology Change Count	Number of times the topology has changed since the bridge was powered or initialised.
Topology Change	Whether the topology is in the middle of changing.

Example To display the current CIST information, use the command:

```
sh mst cist
```

Related Commands

enable mstp
disable mstp
set mstp cist
set mstp cist port
enable mstp cist port
disable mstp cist port

show mstp cist port

Syntax `SHow MSTp CIST POrt[={`*`port-list`*`|ALL}]`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description This command displays the port information about the common internal spanning tree (CIST) (Table 1-9 on page 1-43, and Table 1-11 on page 1-43).

The PORT parameter specifies the ports to display. If ALL is specified, all ports in the switch are displayed.

Figure 1-9: Example output from the **show mstp cist port** command

```
CIST Port Information
---------------------------------------------------------
Port Number.........................1
  Port Identifier...................127:1
  Port Role.........................Designated Port
  Port State........................Forwarding

Port Number.........................2
  Port Identifier...................127:2
  Port Role.........................Designated Port
  Port State........................Forwarding

Port Number.........................3
  Port Identifier...................127:3
  Port Role.........................Designated Port
  Port State........................Forwarding
---------------------------------------------------------
```

Table 1-11: Parameters in output of the **show mstp cist port** command

Parameter	Meaning
Port Number	Number of the port in the switch.
Port Identifier	Unique identifier of the port. This parameter consists of two parts; one part is the port number, and the other is the priority configured for the port.
Port Role	Whether the role of the port is Disabled, Alternate, Backup, Designated, or Root.
Port State	Whether the port is Disabled, Discarding, Learning, or Forwarding.

Figure 1-10: Example output from the **show mstp cist port** command

```
CIST Port Information
-------------------------------------------------------------
Port Number.....................1
  Port Identifier.................128:1
  Port Role.......................Disabled Port
  Port State......................Discarding
  Switch Port State...............Enabled
  Link Status.....................Down
  Port Path Cost..................200000
  External Port Path Cost.........200000
  Designated Bridge...............32768 : 00-00-cd-08-35-e0
  Designated Port.................128:1
  Regional Root Path Cost.........0
  External Root Path Cost.........0
  Edge Port.......................No
  Point to Point Link.............Yes (Auto)
  Boundary Port...................Yes
-------------------------------------------------------------
```

Table 1-12: Parameters in output of the **show mstp cist port** command

Parameter	Meaning
Port Number	Number of the port in the switch.
Port Identifier	Unique identifier of the port. This parameter consists of two parts, one part is the port number, and the other is the priority configured for the port.
Port Role	Whether the role of the port is Disabled, Alternate, Backup, Designated, or Root.
Port State	Whether the state of the port is Disabled, Discarding, Learning, or Forwarding.
Switch Port State	Whether the port is enabled.
Link Status	Whether the link is up or down.
Port Path Cost	Path cost of the port within the region.
External Port Path Cost	Path cost of the port outside the region when the port is a boundary port
Edge Port	Whether this is an edge port that attaches to a LAN and known to have no other bridges attached.
Point to Point Link	Whether the port has a point-to-point connection with another bridge.
Boundary Port	Whether the port is a boundary port in the MST region.

Example To display port 1 information in the CIST, use the command:

```
sh mst cist po=1
```

Related Commands

enable mstp
disable mstp

set mstp cist
enable mstp cist port
show mstp
set mstp cist

show mstp counter port

Syntax `SHow MSTp COUnter POrt={`*`port-list`*`|ALL}`

where *port-list* is a port number, a range of port numbers (specified as *n-m*), or a comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description This command displays counter information for a specified port or ports. See (Figure 14, Table 14).

The PORT parameter specifies the ports to display. If ALL is specified, all ports on the switch are displayed.

Figure 1-11: Example output from the **show mstp counter port** command

```
MSTP Port Counters
------------------------------------------------------------
Port Number    1
Receive:                        Transmit:
  Total BPDUs             0       Total BPDUs           0
  MSTP BPDUs              0       MSTP BPDUs            0
  RSTP BPDUs              0       RSTP BPDUs            0
  STP BPDUs               0       STP BPDUs             0
  Invalid BPDUs           0

Discarded:
  Port Disabled           0
  Invalid Protocol        0
  Invalid Type            0
  Invalid BPDU length     0
 ------------------------------------------------------------
```

Table 1-13: Parameters in output of the **show mstp counter port** command

Parameter	Meaning
Total BPDUs	Whether the role of the port is Disabled, Alternate, Backup, Designated, or Root.
MSTP BPDU	Number of received MSTP BPDUs
RSTP BPDUs	Number of received RSTP BPDUs
STP BPDUs	Number of received STP BPDUs
Invalid BPDUs	Number of received invalid BPDUs
Transmit	BPDUs transmitted
Total BPDUs	Total number of transmitted BPDUs.
MSTP BPDU	Number of transmitted MSTP BPDUs.
RSTP BPDUs	Number of transmitted RSTP BPDUs.
STP BPDUs	Number of transmitted STP BPDUs.
Discard	BPDUs discarded.
Port Disabled	Number of BPDUs discarded because the port that the BPDU was received on was disabled.

Table 1-13: Parameters in output of the **show mstp counter port** command

Parameter	Meaning
Invalid Protocol	Number of BPDUs that had an invalid Protocol Identifier field or invalid Protocol Version Identifier field.
Invalid Type	Number of BPDUs that had an invalid Type field.
Invalid Message Age	Number of BPDUs that had an invalid message age.
Invalid BPDU Length	Number of BPDUs that had an incorrect length.

Examples To display the counters for port 1 to 3, use the command:

```
sh mst po=1-3 cou
```

Related Commands
enable mstp
disable mstp
reset mstp counter port
set mstp cist

show mstp debug msti

Syntax `SHow MSTp DEBug MSTI={CIST|`*`instance`*`|ALL}`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command displays the MSTP debugging modes that are enabled on a specific MSTP instance or all instances.

Example To display the debug mode for all MSTIs, use the command:

```
sh mst deb msti=all
```

Figure 1-12: Example output from the **show mstp debug msti** command

```
MSTP Instance   Port  Debug Modes                Output          Timeout
                       State Machine Debug Modes
-------------------------------------------------------------------------
CIST            1     MSG, STATE                 Asyn 0 (16)     None
                      PTM, PIM, PST, PST
                2     PKT                        Asyn 0 (16)     1
                      All
                3     MSG, PKT, STATE            Asyn 0 (16)     2
                      PRX, PPM, PTX, PRS, PRT, PST
                4     MSG, STATE                 Asyn 0 (16)     3
                      PTM, PIM, PST, PST
-------------------------------------------------------------------------
```

Related Commands
enable mstp debug
disable mstp debug msti
enable mstp
disable mstp
disable mstp

show mstp msti

Syntax `SHow MSTp MSTI[={`*instance*`|All}]`

where *instance* is an instance number from 1 to 4094 for a specific MSTI.

Description This command displays information about a specific Multiple Spanning Tree Instance or all(Table 1-14 on page 1-47).

The MSTI parameter specifies the instance number for the specified Multiple Spanning Tree Instance to be displayed. If ALL is specified, all of the MSTIs are displayed. If no value is specified, summary information about all MSTIs is shown

Figure 1-13: Example output from the **show mstp msti** command

```
Multiple Spanning Tree Instances
------------------------------------------------------------
MSTI ........................... 1
  Bridge Identifier ............ 32768 : 00-00-cd-05-19-28
  Bridge Role .................. Designated Bridge
  VLAN Members ................. 1,3-5,7,9

MSTI ........................... 2
  Bridge Identifier ............ 32767 : 00-00-cd-05-19-28
  Bridge Role .................. Designated Bridge
  VLAN Members ................. 2,6,8,10-12

MSTI ........................... 3
  Bridge Identifier ............ 32766 : 00-00-cd-05-19-28
  Bridge Role .................. Designated Bridge
  VLAN Members ................. 13-20,22
------------------------------------------------------------
```

Table 1-14: Parameters in output of the **show mstp msti** command

Parameter	Meaning
MSTI	Instance number of the spanning tree.
Bridge Identifier	Unique bridge identifier of the switch. This parameter consists of two parts: one is derived from the switch's unique MAC Address, and the other is the priority value entered for the switch.
Bridge Role	Whether the role of the bridge in the spanning tree is root bridge or designated bridge.
VLAN Members	A list of the VLANs that are mapped to a specified multiple spanning tree instance.

Figure 1-14: Example output from the **show mstp msti=1** command

```
Multiple Spanning Tree Instance
------------------------------------------------------------
MSTI ............................ 1
  Bridge Identifier ............. 32768 : 00-00-cd-05-19-28
  Bridge Role ................... Root Bridge
  VLAN Members .................. vlan1, vlan2-vlan10, vlan20
  Regional Root Identifier ...... 32768 : 01-00-cd-05-19-28
  Designated Bridge ............. 32768 : 02-00-cd-05-19-28
  Root Path Cost ................ 32
  Root Port ..................... 2
  Topology Changes:
    Time Since Topology Change .. 100
    Topology Change Count ....... 3
    Topology Change ............. FALSE
------------------------------------------------------------
```

Table 1-15: Parameters in output of the **show mstp msti** command

Parameter	Meaning
MSTI	Instance number of the spanning tree.
Bridge Identifier	Unique Bridge Identifier of the switch. This parameter consists of two parts: one part is derived from the switch's unique MAC Address, and the other part is the priority value entered for the switch.
Bridge Role	Whether the role of the bridge in the spanning tree is root bridge or designated bridge.
VLAN Members	A list of the VLANs that are mapped to a specified multiple spanning tree instance.
Regional Root Identifier	Bridge identifier of the root bridge for the MSTI in an MST region.
Designated Bridge	Bridge identifier for the transmitting bridge for the spanning tree.
Root Path Cost	Path cost to the regional root.
Root Port	Port number of the root port for the switch. This parameter is not valid when the switch is the root bridge and n/a is displayed.
Time Since Topology Change	Seconds elapsed since the last topology change.
Topology Change Count	Number of times that the topology has changed since the bridge was powered on or initialised.
Topology Change	Whether the topology is in the middle of changing.

Example To display the information about a specified MSTI5, use the command:

```
sh mst msti=5
```

Related Commands

enable mstp
disable mstp
set mstp
set mstp msti port

show mstp msti port

Syntax `SHow MSTp MSTI=`*`instance`* `POrt={`*`port-list`*`|ALL}`

where:

- *instance* is an instance number from 1 to 4094 for a specific MSTI.
- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports.

Description This command displays port information of a specific multiple spanning tree instance (MSTI) (Figure 1-15 on page 1-49, and Table 1-16 on page 1-49).

The MSTI parameter specifies the instance number for a specific MSTI to be displayed.

The PORT parameter specifies the ports to display. If ALL is specified, all ports on the switch are displayed.

Figure 1-15: Example output from the **show mstp msti=1 port=1** command

```
MSTI 1 Port Information
------------------------------------------------------
Port Number ................. 1
  Port Identifier ........... 127:1
  Port Role ................. Designated Port
  Port State ................ Forwarding
  Link Status ................Forwarding
  Port Path Cost..............200,000
  Switch Port State ..........Enabled
  Port Path Cost ............ 200
  Designated Bridge...........4096 : 00-00-cd-10-00-37
  Designated Port.............128:3
  ------------------------------------------------------
```

Table 1-16: Parameters in output of the **show mstp msti port** command

Parameter	Meaning
Port Number	Number of the port in the switch.
Port Identifier	Unique identifier of the port. This parameter consists of two parts, one part is the port number, and the other is the priority configured for the port.
Port Role	Whether the he role of the port is Disabled, Alternate, Backup, Designated, or Root.
Port State	Whether the port is Disabled, Discarding, Learning, or Forwarding.
Switch Port State	Whether the switch port is enabled or disabled.
Link Status	Whether the link state of the port is up or down**.**
Port Path Cost	Path cost of the port.
Designated Bridge	Either the unique bridge Identifier of the switch, or the unique bridge identifier of the switch believed to be the designated bridge for the LAN to which the port is attached.

Table 1-16: Parameters in output of the **show mstp msti port** command (Continued)

Parameter	Meaning
Designated Port	Port identifier of the port on the designated bridge through which the designated bridge transmits configuration BPDU information stored by this port.

Example To display the information of port 1 for MSTI5, use the command:

```
sh mst msti=5 po=1
```

Related Commands

enable mstp
disable mstp
set mstp cist

Chapter 2

Generic Packet Classifier

Introduction 2-2
Configuration of Classifiers 2-2
Command Reference 2-3
create classifier 2-4
destroy classifier 2-18
set classifier 2-19
show classifier 2-28

Introduction

The classifier enables you to create packet matching rules—called *classifiers*—to sort packets into *data flows*. For example, you may want all packets with the same destination TCP/IP port to form a flow (e.g. all telnet or HTTP traffic). This chapter describes how to configure the classifier.

You can then configure the router to process all packets in a data flow in a given manner. You have two choices for acting on classified flows:

- Quality of Service (QoS).

  QoS prioritises packets and manages bandwidth. QoS is particularly useful for improving VoIP and video links, especially if your network is congested. Theory and configuration of QoS is described in the Software Quality of Service (QoS) chapter and, for Rapier i Series switches, the Quality of Service (QoS) on Switch Ports chapter.

- Packet filters.

  Filters forward or discard packets. They can also modify the packet's priority settings, send the packet to a mirror port, and do other advanced actions. Classifier-based filters are described in "Classifier-Based Packet Filters" in the Switching chapter.

Configuration of Classifiers

Configuring the classifier involves creating a set of packet matching rules, called *classifiers*, using the command:

```
create classifier=1..9999 [options]
```

These classifiers can identify any single packet based upon many criteria. Available criteria depend on the type of interface you use the classifier on, and include:

- Physical (layer 1) and layer 2 port or interface

  You can classify packets according to their ingress or egress port or ingress interface, and VLAN settings.

- Ethernet encapsulation type

  You can classify packets depending on the specific protocol type of each frame. Different values indicate how the packet is formatted. For example, a value of 802.2 indicates the packet is formatted according to IEEE standards 802.2 and 802.3 with a Destination Service Access Point/Source Service Access Point (DSAP/SSAP) value not equal to AAAA in hexadecimal; SAP encapsulation. A value of ETHII indicates the packet is formatted according to RFC 894; Ethernet II encapsulation. For more details on values see the ETHFORMAT parameter in the **create classifier** command on page 2-4.

- Source/Destination MAC address

  You can classify frames from a specific source or destination MAC address. This classification can be used for users on remote networks. You can also specify MAC type to distinguish unicast packets from broadcast or multicast packets.

- Frame relay, ATM and PPP settings

  You can classify according to DLCI, PPP index number or PPP protocol ID, and ATM VCI or VPI.

- Layer 3 protocols

  You can classify frames based on any value for Layer 3 protocols. Layer 3 protocol and Ethernet encapsulation types are interrelated, e.g. IPX Ethernet II encapsulated packets are different to IPX NETWARERAW encapsulated packets.

- DiffServ or IP TOS

  You can classify packets according to the value of the DSCP bits in the DiffServ field of the header, or the TOS precedence bits in the Type of Service (TOS). These fields are alternatives, so are mutually exclusive.

- Source/destination IP or IPv6 address and other IP settings

  You can classify packets based on an exact match of the source or destination IP address information within the IP or IPv6 header, and based on the presence of several other header fields.

- IPX settings

  You can classify packets based on their destination IPX address, packet type and source or destination socket.

- Layer 4 protocol (TCP/UDP, ICMP etc.)

  You can classify packets based on specific Layer 4 TCP or UDP destination and source port numbers contained within the IP or IPv6 header.

- Layer 4 source/destination port and other layer 4 settings

  You can classify packets based on a specific port number or a range of port numbers, and based on TCP flags, ICMP code and ICMP type.

- Up to three 16-bit words inside the first 64 bytes of a packet

  You can specify the bits to match, using the **match** parameter, and their position, using the **mask** and **offset** parameters.

Command Reference

This section describes the commands available to configure and manage the classifiers.

create classifier

Classifier parameters are sorted approximately in order of the OSI model, with layer 1 (physical) parameters first.

Syntax: hardware filters and QoS on switch ports

For classifiers to use with QoS on switch ports and hardware filters on switch ports:

```
CREate CLASSifier=1..9999 [EPOrt=port] [IPOrt=port]
   [VLAN={vlan-name|1..4094|ANY}]
   [ETHFormat={802.2|802.2-Tagged|802.2-Untagged|Ethii|
   ETHII-Tagged|ETHII-Untagged|Netwareraw|
   NETWARERAW-Tagged|NETWARERAW-Untagged|Snap|SNAP-Tagged|
   SNAP-Untagged|ANY}]
   [MACDaddr={macadd|ANY}] [MACSaddr={macadd|ANY}]
   [PROTocol={protocol-type|IP|IPX|NONIPIPX|ANY}]
   [IPDAddr={ipadd[/0..32]|ANY}] [IPSAddr={ipadd[/
   0..32]|ANY}]
   [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
   [IPPRotocol={TCP|UDP|ICMp|IGMp|NONTcpudp|
   ip-protocol-num|ANY}]
   [IPXDAddr={ipx-add|ANY}]
   [IPXPacket={NLSp|RIP|SAP|SPX|NCP|NETbios|
   ipx-packet-num|ANY}]
   [TCPFlags={{URG|ACK|RST|SYN|FIN}[,...]|ANY}]
   [TCPDport={port-id|ANY}] [TCPSport={port-id|ANY}]
   [UDPDport={port-id|ANY}] [UDPSport={port-id|ANY}]
   [IPXDSocket={NCP|SAP|RIP|NNB|DIAg|NLSp|IPXwan|
   ipx-socket-num|ANY}]
   [IPXSSocket={NCP|SAP|RIP|NNB|DIAg|NLSp|IPXwan|
   ipx-socket-num|ANY}] [MATCH1=hh MASK1=hh OFFSET1=0..62]
   [MATCH2=hh MASK2=hh OFFSET2=0..62]
   [MATCH3=hh MASK3=hh OFFSET3=0..62]
```

Syntax: software QoS on ingress

For classifiers to use with software QoS on ingress traffic over ETH ports; frame relay, ATM, and PPP interfaces; and the switch ports as a single instance on AR440S, AR441S, AR450S, and AR750S routers:

```
CREate CLASSifier=1..9999
   [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
   [VLANPriority={priority-list|ANY}]
```

Syntax: software QoS on egress

For classifiers to use with software QoS on egress traffic over ETH ports; frame relay, ATM, and PPP interfaces; and the switch ports as a single instance on AR440S, AR441S, AR450S, and AR750S routers:

```
CREate CLASSifier=1..9999 [IINTerface=interface|NONE]
   [EPOrt=port|ANY] [IPOrt=port|ANY]
   [SVlan={vlan-name|1..4094|ANY}]
   [DVlan={vlan-name|1..4094|ANY}]
   [VLANPriority={priority-list|ANY}]
   [ETHFormat={802.2|802.2-Tagged|802.2-Untagged|Ethii|
   ETHII-Tagged|ETHII-Untagged|Netwareraw|
   NETWARERAW-Tagged|NETWARERAW-Untagged|Snap|SNAP-Tagged|
   SNAP-Untagged|ANY}]
   [MACDaddr={macadd|ANY}] [MACSaddr={macadd|ANY}]
   [MACType={L2Ucast|L2BMcast|ANY}]
   [ATMVCI={vci-list|ANY}] [ATMVPI={vpi-list|ANY}]
   [DLCi={dlci-range|ANY}] [PPPIndex=0..1023]
```

```
    [PPPProtocolid={ppp-protocol-id|IP|IPv6|ANY}]
    [PROTocol={protocol-type|ARP|IP|IPV6|IPX|ANY}]
    [IPDAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
    [IPSAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
    [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
    [IPFRAg={YES|NO|ANY}] [IPOPtions={YES|NO|ANY}]
    [IPFLowlabel={0..1048575|ANY}]
    [IPPRotocol={TCP|UDP|ICMp|IGMp|OSPf|NONTcpudp|ANY|
    ip-protocol}]
    [ICmptype={Any|ECHORply|Unreachable|Quench|Redirect|
    ECHO|Advertisement|Solicitation|TImeexceed|Parameter|
    TSTAMP|TSTAMPRply|INFOREQ|INFOREP|ADDRREQ|ADDRREP|
    NAMEREq|NAMERPly|icmp-type}]
    [ICMPCode={Any|FIlter|FRAGMent|FRAGReassm|HOSTComm|
    HOSTIsolated|HOSTPrec|HOSTREdirect|HOSTRTos|HOSTTos|
    HOSTUNKnown|HOSTUNReach|NETComm|NETREdirect|NETRTos|
    NETTos|NETUNKnown|NETUNReach|NOptr|POrtunreach|
    PREcedent|PROtunreach|PTrproblem|Sourceroute|Ttl|
    icmp-code}]
    [TCPFlags={{URG|ACK|RST|SYN|FIN}[,...]|ANY}]
    [TCPDport={port-range|ANY}] [TCPSport={port-range|ANY}]
    [UDPDport={port-range|ANY}] [UDPSport={port-range|ANY}]
```

Syntax: software QoS on tunnels

For classifiers to use with software QoS on GRE, IPsec and 6-to-4 tunnels:

```
CREate CLASSifier=1..9999 [IINTerface=interface|NONE]
    [IPDAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
    [IPSAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
    [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
    [IPFRAg={YES|NO|ANY}] [IPOPtions={YES|NO|ANY}]
    [IPFLowlabel={flow-label-range|ANY}]
    [IPPRotocol={TCP|UDP|ICMp|IGMp|OSPf|NONTcpudp|ANY|
    ip-protocol}]
    [ICmptype={Any|ECHORply|Unreachable|Quench|Redirect|
    ECHO|Advertisement|Solicitation|TImeexceed|Parameter|
    TSTAMP|TSTAMPRply|INFOREQ|INFOREP|ADDRREQ|ADDRREP|
    NAMEREq|NAMERPly|icmp-type}]
    [ICMPCode={Any|FIlter|FRAGMent|FRAGReassm|HOSTComm|
    HOSTIsolated|HOSTPrec|HOSTREdirect|HOSTRTos|HOSTTos|
    HOSTUNKnown|HOSTUNReach|NETComm|NETREdirect|NETRTos|
    NETTos|NETUNKnown|NETUNReach|NOptr|POrtunreach|
    PREcedent|PROtunreach|PTrproblem|Sourceroute|Ttl|
    icmp-code}]
    [TCPFlags={{URG|ACK|RST|SYN|FIN}[,...]|ANY}]
    [TCPDport={port-range|ANY}] [TCPSport={port-range|ANY}]
    [UDPDport={port-range|ANY}] [UDPSport={port-range|ANY}]
```

Description

This command creates a classifier, to sort traffic into flows. Classifiers are packet matching rules that identify particular data flows. The data flows may be specific in nature (e.g. IP packets with a particular TCP destination port from a particular source IP address) or general (e.g. all ICMP packets).

For software QoS, you can use up to 16 classifiers per policy on AR725 and AR745 routers and 64 per policy on other routers and switches. Both static classifiers and the dynamic classifiers created by DAR objects count towards this limit.

The syntax above and Table 2-5 on page 2-16 both show whether parameters are valid in classifiers for switch ports or software QoS.

If a packet with an unknown destination port is to be transmitted, the packet is flooded to all ports in the VLAN. For such a packet, no hardware filters or switch port QoS will be applied to the packet. This also applies to any broadcast or multicast IP or IPX packet.

Parameter	Description
CLASSifier	The ID number of the new classifier. An integer in the range 1 to 9999. For classifiers for hardware filters and switch port QoS, the ID number only uniquely identifies the rule, it does not imply an ordering between rules. For software QoS, the traffic class ID number and classifier ID number together determine the rule matching order. Classifiers within each traffic class are checked in ascending order of ID number (lowest first). Default: no default
Layer 1 parameters	
EPOrt	The egress port—the Ethernet switch port through which the frame is destined to leave the switch. An integer in the range 1 to *n*, where *n* is the highest switch port. You can use classifers that contain **eport** in software QoS policies on egress interfaces, or hardware filters, or switch port QoS policies. If you use the classifier for switch port QoS on a 48-port switch, you can only apply the policy to a port in the same port block as the **eport** (port blocks are ports 1-24 and ports 25-48). Default: **any** (ignores egress port)
IPOrt	The ingress port—the Ethernet switch port through which the frame arrives at the switch. An integer in the range 1 to *n*, where *n* is the highest switch port. **Iport** and **iinterface** are mutually exclusive. Default: **any** (ignores ingress port)
IINTerface	The ingress interface—the interface through which the frame arrives at the switch. Valid entries are Layer 1 and 2 interfaces: ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● DS3 ● PPP (e.g. ppp0) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command. **Iport** and **iinterface** are mutually exclusive. **Iinterface** is only valid in classifiers for software QoS on egress interfaces or tunnels. Default: **none** (ignores ingress interface)
Layer 2 Ethernet parameters	
SVlan	The source VLAN—the VLAN associated with the frame when it arrives at the switch. Only valid in classifiers for software QoS on egress interfaces. Default: **any**
vlan-name	The name of the source VLAN. To see a list of current VLANs, use the **show vlan** command.
1..4094	The VLAN Identifier (VID) of the source VLAN.
ANY	The classifier ignores the source VLAN.

Layer 2 Ethernet parameters (Continued)		
DVlan	The destination VLAN—the VLAN that the frame will be transmitted to. Only valid in classifiers for software QoS on egress interfaces. Default: **any**	
	vlan-name	The name of the destination VLAN. To see a list of current VLANs, use the **show vlan** command.
	1..4094	The VLAN Identifier (VID) of the destination VLAN.
	ANY	The classifier ignores the destination VLAN.
VLAN	The destination VLAN—the VLAN that the frame will be transmitted to. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any**	
	vlan-name	The name of the destination VLAN. To see a list of current VLANs, use the **show vlan** command.
	1..4094	The VLAN Identifier (VID) of the destination VLAN.
	ANY	The classifier ignores the destination VLAN.
VLANPriority	The 802.1p VLAN priority value in the frame. An integer in the range 0 to 7; a range of integers separated by hyphens; or a comma separated list of integers and/or ranges (for example 0,2,4-6). Only valid in classifiers for software QoS on ingress and egress interfaces. Default: **any** (ignores VLAN priority)	
ETHFormat	The Ethernet encapsulation type of the frame. The **ethformat** and **protocol** must match. Table 2-1 on page 2-13 and Table 2-2 on page 2-14 show possible combinations and whether they are valid. You can distinguish between frames that are tagged and untagged at ingress. Default: **any**	
	802.2 802.2-Tagged 802.2-Untagged	Formatted according to IEEE Standards 802.2 and 802.3 with a DSAP/SSAP value not equal to hexadecimal AAAA. Encapsulation: SAP
	Ethii ETHII-Tagged ETHII-Untagged	Formatted according to RFC 894, *Standard for the transmission of IP datagrams over Ethernet networks.* Encapsulation: Ethernet II
	Netwareraw NETWARERAW-Tagged NETWARERAW-Untagged	Formatted as an IPX packet according to IEEE Standard 802.3. Encapsulation: NetWare Raw or Novell
	Snap SNAP-Tagged SNAP-Untagged	Formatted according to IEEE Standards 802.2 and 802.3 and RFC 1042, *Standard for the transmission of IP datagrams over IEEE 802 networks.* Encapsulation: SNAP
	ANY	The classifier ignores the Ethernet encapsulation.

Layer 2 Ethernet parameters (Continued)	
MACDaddr	The destination MAC address of the frame. An Ethernet six-octet MAC address, expressed as six pairs of hexadecimal digits delimited by hyphens. For classifiers attached to switch ports, it may not be possible to specify Layer 2 MAC address-based packet matching rules in conjunction with Layer 3 packet matching rules. If so, **macdaddr** and **macsaddr** will not be available. Default: **any** (ignores destination MAC address).
MACSaddr	The source MAC address of the frame. An Ethernet six-octet MAC address, expressed as six pairs of hexadecimal digits delimited by hyphens. For classifiers attached to switch ports, it may not be possible to specify Layer 2 MAC address-based packet matching rules in conjunction with Layer 3 packet matching rules. If so, **macdaddr** and **macsaddr** will not be available. Default: **any** (ignores source MAC address).
MACType	The type of destination MAC address on the frame. Only valid in classifiers for software QoS on egress interfaces. Default: **any**
L2Ucast	Layer 2 unicast addresses.
L2BMcast	Layer 2 broadcast or multicast addresses.
ANY	The classifier ignores the MAC address type.
PROTocol	The protocol, determined from the value of the following Ethernet field: ● for 802.2 (SAP encapsulation): the DSAP field, 1 byte hexadecimal ● for ETHII encapsulation: the ETYPE field, 2 bytes hexadecimal ● for NETWARERAW encapsulation: the IPX checksum field, 2 bytes hexadecimal with value FFFF ● for SNAP encapsulation: the ETYPE field, 5 bytes hexadecimal. The classifier matches on the last 2 bytes. The encapsulation type (**ethformat** parameter) and **protocol** must match. Table 2-1 on page 2-13 and Table 2-2 on page 2-14 show possible combinations and whether they are valid. Default: **any**, unless you also specify a TCP or UDP parameter (for example, **tcpsport**). Then the default is **IP**.
protocol-type	The protocol number or the predefined protocol name. Table 2-3 on page 2-15 shows predefined protocols, their numbers, and their encapsulations.
IP	Internet Protocol version 4. Valid with **ethformat** of **ethii** or **snap**.
IPV6	Internet Protocol version 6. Valid with **ethformat** of **ethii**. Only valid in classifiers for software QoS.
ARP	Address Resolution Protocol. Valid with **ethformat** of **ethii** or **snap**. Only valid in classifiers for software QoS.
IPX	IPX. Valid with **ethformat** of **802.2**, **ethii**, **netwareraw** or **snap**.
NONIPIPX	All protocols except for IP and IPX. Valid with **ethformat** of **802.2**, **ethii** or **snap**. Only valid in classifiers for hardware filters and switch port QoS.
ANY	The classifier ignores the protocol.

Layer 2 parameters (ATM, frame relay and PPP)	
ATMVCi	The Virtual Channel Identifier for an ATM connection. An integer in the range 0 to 255, or a range of integers separated by a hyphen (for example 0-3). Only valid in classifiers for software QoS on egress interfaces. Default: **any** (ignores ATM VCI).
ATMVPi	The Virtual Path Identifier for an ATM connection. An integer in the range 0 to 4095, or a range of integers separated by a hyphen (for example 0-3). Only valid in classifiers for software QoS on egress interfaces. Default: **any** (ignores ATM VPI).
DLCi	The identification number of a Frame Relay Data Link Connection (DLC). An integer in the range 0 to 1023, or a range of integers separated by a hyphen (for example 0-3). Only valid in classifiers for software QoS on egress interfaces. Default: **any** (ignores DLCI).
PPPIndex	The PPP interface number. For example, for ppp2, **pppindex=2**.
PPPProtocolid	The network layer protocol of the PPP encapsulated packet. Note that network and link control packets are processed by the software QoS policy's system traffic class. Examples of control packets include NCP, LCP, IPCP and PAP. Only valid in classifiers for software QoS on egress interfaces. Default: **any**, unless you also specify a TCP or UDP parameter. Then the default is **IP**.
ppp-protocol-id|	A 4 byte hexadecimal protocol number. Table 2-4 on page 2-16 shows valid protocols and numbers.
IP	Internet Protocol.
IPV6	Internet Protocol version 6.
ANY	The classifier ignores PPP protocol ID.
Layer 3 parameters	
IPDAddr	The destination IPv4 or IPv6 address of the packet. Default: **any**
ipadd[/0..32]	The destination IPv4 address, in dotted decimal notation. You can optionally specify a subnet by specifying a mask.
ipv6add[/0..128]	The destination IPv6 address, specified as eight pairs of hexadecimal octets separated by colons. You can optionally specify a prefix length. Default prefix length is 128—a single address. IPv6 addresses are only valid in classifiers for software QoS on egress or tunnel interfaces.
ANY	The classifier ignores destination IP or IPv6 address.

Layer 3 parameters (Continued)		
IPSAddr	The source IPv4 or IPv6 address of the packet. Default: **any**	
	ipadd[/0..32]	The source IPv4 address, in dotted decimal notation. You can optionally specify a subnet by specifying a mask.
	ipv6add[/0..128]	The source IPv6 address, specified as eight pairs of hexadecimal octets separated by colons. You can optionally specify a prefix length. Default prefix length is 128—a single address. IPv6 addresses are only valid in classifiers for software QoS on egress or tunnel interfaces.
	ANY	The classifier ignores source IPv4 or IPv6 address.
IPDScp	The DSCP value—the Code Point bits of the DiffServ field of an IPv4 or IPv6 packet. An integer in the range 0 to 63; a range of integers separated by hyphens; or a comma separated list of integers and/or ranges (for example 0,2,4-6). You can also specify EF, AF1, AF2, AF3 or AF4. **Ipdscp** and **Iptos** are mutually exclusive. Default: **any** (ignores DSCP).	
IPTOs	The TOS value—the value of the precedence field within the TOS byte of an IPv4 packet. An integer in the range 0 to 7. **Ipdscp** and **Iptos** are mutually exclusive. **Iptos** is only valid for IPv4 packets. Default: **any** (ignores TOS).	
IPFRAg	Whether the IPv4 packet is fragmented. Only valid in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores whether the packet is fragmented).	
IPOPtions	Whether the packet includes the IPv4 header options field. Only valid in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores whether the header options field is present or not).	
IPFLowlabel	The IPv6 flow label in an IPv6 packet, an integer in the range 0 to 1048575. Only valid for IPv6 packets in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores IPv6 flow label).	
IPXDAddr	The destination network address of an IPX packet, expressed as a 4 byte hexadecimal number. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores destination IPX address).	
IPXPacket	The value of the Packet Type field of an IPX packet. One of the options NLSp, RIP, SAP, SPX, NCP, or NETbios; or a 2 byte hexadecimal IPX packet number; or a recognised IPX packet type. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores IPX packet type).	

Layer 3 parameters (Continued)	
IPPRotocol	The Layer 4 IPv4 or IPv6 protocol of the packet. For IPv6 packets, **ipprotocol** matches against the Next Header field of the IPv6 packet header. You can use a total of 29 unique **ipprotocol** values, plus TCP and UDP, in total across all classifiers. Default: ● **tcp** if you also specify a TCP parameter (for example, **tcpsport**). ● **udp** if you also specify a UDP parameter (for example, **udpsport**). ● Otherwise, **any** (ignores IP protocol).
ip-protocol	A 1 byte decimal IPv4 or IPv6 protocol number or a well-known protocol name.
TCP	Transmission Control Protocol.
UDP	User Datagram Protocol.
NONTcpudp	Any IPv4 or IPv6 protocol except TCP or UDP.
ICMP	Internet Control Message Protocol.
IGMP	Internet Group Multicast Protocol.
OSPF	Open Shortest Path First. Only valid in classifiers for software QoS on egress or tunnel interfaces.
ANY	The classifier ignores the IP protocol value.

Layer 4 parameters	
ICmptype	The ICMP message type to match against the ICMP type field in an ICMP packet header. One of the list of options, or a decimal value in the range 0 to 65535. Only valid if **ipprotocol=icmp** in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores ICMP type).
ICMPCode	The ICMP message reason code to match against the ICMP code field in an ICMP packet header. One of the list of options, or a decimal value in the range 0 to 65535. Only valid if **ipprotocol=icmp** in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores ICMP code).
TCPFlags	The TCP flags of the TCP/IP packet. One or a comma-separated list of the options URG, ACK, RST, SYN and FIN. Default: **any** (ignores TCP flag).
TCPDport	The destination TCP port—the value in the TCP destination port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores destination TCP port).
TCPSport	The source TCP port—the value in the TCP source port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores source TCP port).
UDPDport	The destination UDP port—the value in the UDP destination port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores destination UDP port).

Layer 4 parameters (Continued)	
UDPSport	The source UDP port—the value in the UDP source port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores source UDP port).
IPXDSocket	The destination IPX socket number of an IPX packet. One of the options NCP, SAP, RIP, NNB, DIAg, NLSp or IPXwan; or a 2 byte hexadecimal IPX socket number; or a recognised IPX socket type. You can use a total of 7 unique **ipxdsocket** values across all classifiers, including any of the options. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores destination IPX socket).
IPXSSocket	The source IPX socket number of an IPX packet. One of the options NCP, SAP, RIP, NNB, DIAg, NLSp or IPXwan; or a 2 byte hexadecimal IPX socket number; or a recognised IPX socket type. You can use a total of 7 unique **ipxssocket** values across all classifiers, including any of the options. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores source IPX socket).
Bit matching parameters	
MATCH1 MATCH2 MATCH3	A general 16-bit word to match inside a packet, specified as a 2 byte hexadecimal number. **Match** specifies the actual data to match. You must specify all three of **match*x***, **mask*x*** and **offset*x*** together, where *x* is 1, 2 or 3. Only valid in classifiers for hardware filters and QoS on switch ports on Rapier i Series switches. Default: no default
MASK1 MASK2 MASK3	Whether the corresponding bit in **match** is "on" for a match or "don't care" for a match, specified as a 2 byte hexadecimal number. If the **mask** bit is set (on), the bit in **match** must be the same as the corresponding bit in the actual packet (so place binary "ones" in bit positions you want to match). If the **mask** bit is clear (don't care), the same bit in **match** will not be checked with the corresponding bit in the actual packet. Only valid in classifiers for hardware filters and QoS on switch ports on Rapier i Series switches. Default: no default
OFFSET1 OFFSET2 OFFSET3	The offset from the start of the packet, specified as an integer in the range 0 to 62. You must specify offsets in order (e.g. **offset1** before **offset2**). Only valid in classifiers for hardware filters and QoS on switch ports on Rapier i Series switches. Default: no default
* The shortest string you can enter is shown in capital letters.	

Table 2-1: Possible **ethformat** and **protocol** parameter combinations for AR400 and AR700 routers

ethformat	protocol	validity
ETHII	[not specified]	OK
	ANY	OK
	ARP	OK
	IP	OK (equivalent to protocol=0800)
	IPV6	OK
	IPX	OK (equivalent to protocol=8137)
	protocol-type	OK (see Table 2-3 for valid combinations)
NETWARERAW	[not specified]	OK (equivalent to protocol="IPX 802.3")
	ANY	OK (equivalent to protocol="IPX 802.3")
	ARP	Error
	IP	Error
	IPV6	Error
	IPX	OK (equivalent to protocol="IPX 802.3")
	"IPX 802.3"	OK
	protocol-type	Error
SNAP	[not specified]	OK
	ANY	OK
	ARP	OK
	IP	OK
	IPV6	Error
	IPX	OK
	protocol-type	OK (see Table 2-3 for valid combinations)
802.2	[not specified]	OK
	ANY	OK
	ARP	Error
	IP	Error
	IPV6	Error
	IPX	OK
	protocol-type	OK (see Table 2-3 for valid combinations)

Table 2-2: Possible **ethformat** and **protocol** parameter combinations for Rapier and Rapier i switches

ethformat	protocol	validity
ETHII	[not specified]	OK
	ANY	OK
	ARP	OK
	IP	OK (equivalent to protocol=0800)
	IPV6	OK
	IPX	OK (equivalent to protocol=8137)
	NONIPIPX	OK
	protocol-type	OK (see Table 2-3 for valid combinations)
NETWARERAW	[not specified]	OK (equivalent to protocol="IPX 802.3")
	ANY	OK (equivalent to protocol="IPX 802.3")
	ARP	Error
	IP	Error
	IPV6	Error
	IPX	OK (equivalent to protocol="IPX 802.3")
	"IPX 802.3"	OK
	NONIPIPX	Error
	protocol-type	Error
SNAP	[not specified]	OK
	ANY	OK
	ARP	OK
	IP	OK
	IPV6	Error
	IPX	OK
	NONIPIPX	OK
	protocol-type	OK (see Table 2-3 for valid combinations)
802.2	[not specified]	OK
	ANY	OK
	ARP	Error
	IP	Error
	IPV6	Error
	IPX	OK
	NONIPIPX	OK
	protocol-type	OK (see Table 2-3 for valid combinations)

Table 2-3: Predefined protocol types for use in the **protocol** parameter

Protocol Name	Protocol Number	Encapsulation	Min characters to enter
SNA Path Control	04	SAP	3
PROWAY-LAN	0E	SAP	7
EIA-RS	4E	SAP	3
PROWAY	8E	SAP	3
IPX 802.2	E0	SAP	9
NetBEUI	F0	SAP	3
ISO CLNS IS	FE	SAP	5
IP ETHII	0800	EthII	8
X.75 Internet	0801	EthII	4
NBS Internet	0802	EthII	3
ECMA Internet	0803	EthII	4
Chaosnet	0804	EthII	4
X.25 Level 3	0805	EthII	4
ARP	0806	EthII	3
XNS Compat	0807	EthII	3
Banyan Systems	0BAD	EthII	3
BBN Simnet	5208	EthII	3
DEC MOP Dump/Ld	6001	EthII	9
DEC MOP Rem Cons	6002	EthII	9
DEC DECNET	6003	EthII	7
DEC LAT	6004	EthII	7
DEC Diagnostic	6005	EthII	7
DEC Customer	6006	EthII	7
DEC LAVC	6007	EthII	7
RARP	8035	EthII	4
DEC LANBridge	8038	EthII	7
DEC Encryption	803D	EthII	7
AppleTalk	809B	EthII	3
IBM SNA	80D5	EthII	7
IPX EthII	8137	EthII	9
AppleTalk AARP	80F3	EthII	11
SNMP	814C	EthII	4
IPv6 ETHII	86DD	EthII	10
IPX 802.3	FFFF	NetWare 802.3 Raw	9
ETHERTALK 2	080007809B	SNAP	11
ETHERTALK 2 AARP	00000080F3	SNAP	13
IPX SNAP	0000008137	SNAP	8

Note: When you enter a protocol name that contains spaces, you must surround the name with double quotation marks. You can use lowercase or uppercase letters. For example, to specify ETHERTALK 2 AARP, enter **protocol="ethertalk 2 aarp"** or **protocol="ethertalk 2 a"**.

Table 2-4: PPP Network Layer protocol ID values for use in the **pppprotocolid** parameter

PPP Protocol	Number	Long Name
IP	0021	Internet Protocol
OSI	0023	OSI Network Layer
DEC	0027	Decnet Phase IV
APP	0029	Appletalk
IPX	002B	IPX
VJC	002D	Van Jacobson Compressed TCP/IP
VJU	002F	Van Jacobson Uncompressed TCP/IP
BRI	0031	Bridging PDU
MP	003D	Multilink Protocol
IP6HC	004F	IP6 Header Compression
ENC	0053	Encryption
IPV6	0057	Internet Protocol version 6
SINGLE	00FB	Single Link Compression in Multilink
Compressed	00FD	Compressed Datagram

Table 2-5: The classifier parameters that are valid for hardware filters and QoS on switch ports, and software QoS on ingress interfaces, egress interfaces, and tunnels

Parameter	Hardware filters and switch ports	Software QoS egress	Software QoS tunnels	Software QoS ingress
atmvci		✓		
atmvpi		✓		
dlci		✓		
dvlan		✓		
eport	✓	✓		
ethformat	✓	✓		
icmpcode		✓	✓	
icmptype		✓	✓	
iinterface		✓	✓	
ipdaddr	✓ (IPv4 only)	✓ (IPv4, IPv6)	✓ (IPv4, IPv6)	
ipdscp	✓	✓	✓	✓
ipflowlabel		✓	✓	
ipfrag		✓	✓	
ipoptions		✓	✓	
iport	✓	✓	✓	
ipprotocol	✓	✓	✓	
ipsaddr	✓ (IPv4 only)	✓ (IPv4, IPv6)	✓ (IPv4, IPv6)	
iptos	✓	✓	✓	✓
ipxdaddr	✓			

Table 2-5: The classifier parameters that are valid for hardware filters and QoS on switch ports, and software QoS on ingress interfaces, egress interfaces, and tunnels (Continued)

Parameter	Hardware filters and switch ports	Software QoS egress	Software QoS tunnels	Software QoS ingress
ipxdsocket	✓			
ipxpacket	✓			
ipxssocket	✓			
macdaddr	✓	✓		
macsaddr	✓	✓		
mactype		✓		
maskx	✓ (Rapier i)			
matchx	✓ (Rapier i)			
offsetx	✓ (Rapier i)			
pppindex		✓		
pppprotocolid		✓		
protocol	✓	✓		
svlan		✓		
tcpdport	✓ (single value)	✓ (range)	✓ (range)	
tcpflags	✓ (Rapier i)	✓	✓	
tcpsport	✓ (single value)	✓ (range)	✓ (range)	
udpdport	✓ (single value)	✓ (range)	✓ (range)	
udpsport	✓ (single value)	✓ (range)	✓ (range)	
vlan	✓			
vlanpriority		✓		✓

Examples

To create packet matching rule 1 so that it matches all IP packets from the IP subnet 192.168.100.2 (mask=255.255.255.0), with a destination TCP port of 23, use one of the commands:

```
cre class=1 ipsaddr=192.168.100.2/24 tcpdport=23
cre class=1 protocol=ip ipsaddr=202.36.164.2/24 tcpdport=23
```

To create packet matching rules to separate PPPoE interfaces 1 and 2 on an Ethernet interface, use the commands:

```
cre class=1 pppi=1
cre class=2 pppi=2
```

Related Commands

destroy classifier
set classifier
show classifier

destroy classifier

Syntax `DESTroy CLASSifier={`*`rule-list`*`|ALL}`

Description This command destroys one or more packet matching rules. You cannot destroy a classifier that is being used by a hardware filter or QoS.

The **classifier** parameter specifies the classifiers to destroy, and is the rule ID of an existing classifier, a comma-separated list of rule IDs, a range of rule IDs separated by a hyphen, or a combination (for example, 3,5,9-12). If you specify **all**, then all classifiers are destroyed.

Examples To destroy the packet matching rules with rule-ids 3, 5 and 9 to 12, use the command:

```
dest class=3,5,9-12
```

To destroy all packet matching rules, use the command:

```
dest class=all
```

Related Commands
create classifier
set classifier
show classifier

set classifier

Classifier parameters are sorted approximately in order of the OSI model, with layer 1 (physical) parameters first.

Syntax: hardware filters and QoS on switch ports

For classifiers to use with QoS on switch ports (on Rapier i Series switches only) and hardware filters on switch ports:

```
SET CLASSifier=1..9999 [EPOrt=port] [IPOrt=port]
   [VLAN={vlan-name|1..4094|ANY}]
   [ETHFormat={802.2|802.2-Tagged|802.2-Untagged|Ethii|
   ETHII-Tagged|ETHII-Untagged|Netwareraw|
   NETWARERAW-Tagged|NETWARERAW-Untagged|Snap|SNAP-Tagged|
   SNAP-Untagged|ANY}]
   [MACDaddr={macadd|ANY}] [MACSaddr={macadd|ANY}]
   [PROTocol={protocol-type|IP|IPX|NONIPIPX|ANY}]
   [IPDAddr={ipadd[/0..32]|ANY}] [IPSAddr={ipadd[/
   0..32]|ANY}]
   [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
   [IPPRotocol={TCP|UDP|ICMp|IGMp|NONTcpudp|
   ip-protocol-num|ANY}]
   [IPXDAddr={ipx-add|ANY}]
   [IPXPacket={NLSp|RIP|SAP|SPX|NCP|NETbios|
   ipx-packet-num|ANY}]
   [TCPFlags={{URG|ACK|RST|SYN|FIN}[,...]|ANY}]
   [TCPDport={port-id|ANY}] [TCPSport={port-id|ANY}]
   [UDPDport={port-id|ANY}] [UDPSport={port-id|ANY}]
   [IPXDSocket={NCP|SAP|RIP|NNB|DIAg|NLSp|IPXwan|
   ipx-socket-num|ANY}]
   [IPXSSocket={NCP|SAP|RIP|NNB|DIAg|NLSp|IPXwan|
   ipx-socket-num|ANY}] [MATCH1=hh MASK1=hh OFFSET1=0..62]
   [MATCH2=hh MASK2=hh OFFSET2=0..62]
   [MATCH3=hh MASK3=hh OFFSET3=0..62]
```

Syntax: software QoS on ingress

For classifiers to use with software QoS on ingress traffic over ETH ports; frame relay, ATM, and PPP interfaces; and the switch ports as a single instance on AR440S, AR441S, AR450S, and AR750S routers:

```
SET CLASSifier=1..9999
   [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
   [VLANPriority={priority-list|ANY}]
```

Syntax: software QoS on egress

For classifiers to use with software QoS on egress traffic over ETH ports; frame relay, ATM, and PPP interfaces; and the switch ports as a single instance on AR440S, AR441S, AR450S, and AR750S routers:

```
SET CLASSifier=1..9999 [IINTerface=interface|NONE]
   [EPOrt=port|ANY] [IPOrt=port|ANY]
   [SVlan={vlan-name|1..4094|ANY}]
   [DVlan={vlan-name|1..4094|ANY}]
   [VLANPriority={priority-list|ANY}]
   [ETHFormat={802.2|802.2-Tagged|802.2-Untagged|Ethii|
   ETHII-Tagged|ETHII-Untagged|Netwareraw|
   NETWARERAW-Tagged|NETWARERAW-Untagged|Snap|SNAP-Tagged|
   SNAP-Untagged|ANY}]
   [MACDaddr={macadd|ANY}] [MACSaddr={macadd|ANY}]
   [MACType={L2Ucast|L2BMcast|ANY}]
   [ATMVCI={vci-list|ANY}] [ATMVPI={vpi-list|ANY}]
   [DLCi={dlci-range|ANY}] [PPPIndex=0..1023]
```

```
   [PPPProtocolid={ppp-protocol-id|IP|IPv6|ANY}]
   [PROTocol={protocol-type|ARP|IP|IPV6|IPX|ANY}]
   [IPDAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
   [IPSAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
   [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
   [IPFRAg={YES|NO|ANY}] [IPOPtions={YES|NO|ANY}]
   [IPFLowlabel={0..1048575|ANY}]
   [IPPRotocol={TCP|UDP|ICMp|IGMp|OSPf|NONTcpudp|ANY|
   ip-protocol}]
   [ICmptype={Any|ECHORply|Unreachable|Quench|Redirect|
   ECHO|Advertisement|Solicitation|TImeexceed|Parameter|
   TSTAMP|TSTAMPRply|INFOREQ|INFOREP|ADDRREQ|ADDRREP|
   NAMEREq|NAMERPly|icmp-type}]
   [ICMPCode={Any|FIlter|FRAGMent|FRAGReassm|HOSTComm|
   HOSTIsolated|HOSTPrec|HOSTREdirect|HOSTRTos|HOSTTos|
   HOSTUNKnown|HOSTUNReach|NETComm|NETREdirect|NETRTos|
   NETTos|NETUNKnown|NETUNReach|NOptr|POrtunreach|
   PREcedent|PROtunreach|PTrproblem|Sourceroute|Ttl|
   icmp-code}]
   [TCPFlags={{URG|ACK|RST|SYN|FIN}[,...]|ANY}]
   [TCPDport={port-range|ANY}] [TCPSport={port-range|ANY}]
   [UDPDport={port-range|ANY}] [UDPSport={port-range|ANY}]
```

Syntax: software QoS on tunnels

For classifiers to use with software QoS on GRE, IPsec and 6-to-4 tunnels:

```
SET CLASSifier=1..9999 [IINTerface=interface|NONE]
   [IPDAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
   [IPSAddr={ipadd[/0..32]|ipv6add[/0..128]|ANY}]
   [IPDScp={dscp-list|ANY}] [IPTOs={0..7|ANY}]
   [IPFRAg={YES|NO|ANY}] [IPOPtions={YES|NO|ANY}]
   [IPFLowlabel={flow-label-range|ANY}]
   [IPPRotocol={TCP|UDP|ICMp|IGMp|OSPf|NONTcpudp|ANY|
   ip-protocol}]
   [ICmptype={Any|ECHORply|Unreachable|Quench|Redirect|
   ECHO|Advertisement|Solicitation|TImeexceed|Parameter|
   TSTAMP|TSTAMPRply|INFOREQ|INFOREP|ADDRREQ|ADDRREP|
   NAMEREq|NAMERPly|icmp-type}]
   [ICMPCode={Any|FIlter|FRAGMent|FRAGReassm|HOSTComm|
   HOSTIsolated|HOSTPrec|HOSTREdirect|HOSTRTos|HOSTTos|
   HOSTUNKnown|HOSTUNReach|NETComm|NETREdirect|NETRTos|
   NETTos|NETUNKnown|NETUNReach|NOptr|POrtunreach|
   PREcedent|PROtunreach|PTrproblem|Sourceroute|Ttl|
   icmp-code}]
   [TCPFlags={{URG|ACK|RST|SYN|FIN}[,...]|ANY}]
   [TCPDport={port-range|ANY}] [TCPSport={port-range|ANY}]
   [UDPDport={port-range|ANY}] [UDPSport={port-range|ANY}]
```

Description

This command modifies a classifier. Classifiers are packet matching rules that identify particular data flows. The data flows may be specific in nature (e.g. IP packets with a particular TCP destination port from a particular source IP address) or general (e.g. all ICMP packets).

The syntax above and Table 2-5 on page 2-16 both show whether parameters are valid in classifiers for switch ports or software QoS.

Parameter	Description
CLASSifier	The ID number of the classifier. For classifiers for hardware filters and switch port QoS, the ID number only uniquely identifies the rule, it does not imply an ordering between rules. For classifiers for software QoS, the ID number determines the rule matching order. Within each traffic class, the classifiers are checked in ascending order of ID number (lowest first). Default: no default
Layer 1 parameters	
EPOrt	The egress port—the Ethernet switch port through which the frame is destined to leave the switch. An integer in the range 1 to *n*, where *n* is the highest switch port. You can use classifers that contain **eport** in software QoS policies on egress interfaces, or hardware filters, or switch port QoS policies. If you use the classifier for switch port QoS on a 48-port switch, you can only apply the policy to a port in the same port block as the **eport** (port blocks are ports 1-24 and ports 25-48). Default: **any** (ignores egress port)
IPOrt	The ingress port—the Ethernet switch port through which the frame arrives at the switch. An integer in the range 1 to *n*, where *n* is the highest switch port. **Iport** and **iinterface** are mutually exclusive. Default: **any** (ignores ingress port)
IINTerface	The ingress interface—the interface through which the frame arrives at the switch. Valid entries are Layer 1 and 2 interfaces: ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● DS3 ● PPP (e.g. ppp0) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command. **Iport** and **iinterface** are mutually exclusive. **Iinterface** is only valid in classifiers for software QoS on egress interfaces or tunnels. Default: **none** (ignores ingress interface)
Layer 2 Ethernet parameters	
SVlan	The source VLAN—the VLAN associated with the frame when it arrives at the switch. Only valid in classifiers for software QoS on egress interfaces. Default: **any**
vlan-name	The name of the source VLAN. To see a list of current VLANs, use the **show vlan** command.
1..4094	The VLAN Identifier (VID) of the source VLAN.
ANY	The classifier ignores the source VLAN.
DVlan	The destination VLAN—the VLAN that the frame will be transmitted to. Only valid in classifiers for software QoS on egress interfaces. Default: **any**
vlan-name	The name of the destination VLAN. To see a list of current VLANs, use the **show vlan** command.
1..4094	The VLAN Identifier (VID) of the destination VLAN.
ANY	The classifier ignores the destination VLAN.

Layer 2 Ethernet parameters (Continued)		
VLAN	The destination VLAN—the VLAN that the frame will be transmitted to. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any**	
	vlan-name	The name of the destination VLAN. To see a list of current VLANs, use the **show vlan** command.
	1..4094	The VLAN Identifier (VID) of the destination VLAN.
	ANY	The classifier ignores the destination VLAN.
VLANPriority	The 802.1p VLAN priority value in the frame. An integer in the range 0 to 7; a range of integers separated by hyphens; or a comma separated list of integers and/or ranges (for example 0,2,4-6). Only valid in classifiers for software QoS on ingress and egress interfaces. Default: **any** (ignores VLAN priority)	
ETHFormat	The Ethernet encapsulation type of the frame. The **ethformat** and **protocol** must match. Table 2-1 on page 2-13 and Table 2-2 on page 2-14 show possible combinations and whether they are valid. You can distinguish between frames that are tagged and untagged at ingress. Default: **any**	
	802.2 802.2-Tagged 802.2-Untagged	Formatted according to IEEE Standards 802.2 and 802.3 with a DSAP/SSAP value not equal to hexadecimal AAAA. Encapsulation: SAP
	Ethii ETHII-Tagged ETHII-Untagged	Formatted according to RFC 894, *Standard for the transmission of IP datagrams over Ethernet networks.* Encapsulation: Ethernet II
	Netwareraw NETWARERAW-Tagged NETWARERAW-Untagged	Formatted as an IPX packet according to IEEE Standard 802.3. Encapsulation: NetWare Raw or Novell
	Snap SNAP-Tagged SNAP-Untagged	Formatted according to IEEE Standards 802.2 and 802.3 and RFC 1042, *Standard for the transmission of IP datagrams over IEEE 802 networks.* Encapsulation: SNAP
	ANY	The classifier ignores the Ethernet encapsulation.
MACDaddr	The destination MAC address of the frame. An Ethernet six-octet MAC address, expressed as six pairs of hexadecimal digits delimited by hyphens. For classifiers attached to switch ports, it may not be possible to specify Layer 2 MAC address-based packet matching rules in conjunction with Layer 3 packet matching rules. If so, **macdaddr** and **macsaddr** will not be available. Default: **any** (ignores destination MAC address).	
MACSaddr	The source MAC address of the frame. An Ethernet six-octet MAC address, expressed as six pairs of hexadecimal digits delimited by hyphens. For classifiers attached to switch ports, it may not be possible to specify Layer 2 MAC address-based packet matching rules in conjunction with Layer 3 packet matching rules. If so, **macdaddr** and **macsaddr** will not be available. Default: **any** (ignores source MAC address).	

Layer 2 Ethernet parameters (Continued)	
MACType	The type of destination MAC address on the frame. Only valid in classifiers for software QoS on egress interfaces. Default: **any**
L2Ucast	Layer 2 unicast addresses.
L2BMcast	Layer 2 broadcast or multicast addresses.
ANY	The classifier ignores the MAC address type.
PROTocol	The protocol, determined from the value of the following Ethernet field: ● for 802.2 (SAP encapsulation): the DSAP field, 1 byte hexadecimal ● for ETHII encapsulation: the ETYPE field, 2 bytes hexadecimal ● for NETWARERAW encapsulation: the IPX checksum field, 2 bytes hexadecimal with value FFFF ● for SNAP encapsulation: the ETYPE field, 5 bytes hexadecimal. The classifier matches on the last 2 bytes. The encapsulation type (**ethformat** parameter) and **protocol** must match.Table 2-1 on page 2-13 and Table 2-2 on page 2-14 show possible combinations and whether they are valid. Default: **any**, unless you also specify a TCP or UDP parameter (for example, **tcpsport**). Then the default is **IP**.
protocol-type	The protocol number or the predefined protocol name. Table 2-3 on page 2-15 shows predefined protocols, their numbers, and their encapsulations.
IP	Internet Protocol version 4. Valid with **ethformat** of **ethii** or **snap**.
IPV6	Internet Protocol version 6. Valid with **ethformat** of **ethii**. Only valid in classifiers for software QoS.
ARP	Address Resolution Protocol. Valid with **ethformat** of **ethii** or **snap**. Only valid in classifiers for software QoS.
IPX	IPX. Valid with **ethformat** of **802.2**, **ethii**, **netwareraw** or **snap**.
NONIPIPX	All protocols except for IP and IPX. Valid with **ethformat** of **802.2**, **ethii** or **snap**. Only valid in classifiers for hardware filters and switch port QoS.
ANY	The classifier ignores the protocol.

Layer 2 parameters (ATM, frame relay and PPP)	
ATMVCi	The Virtual Channel Identifier for an ATM connection. An integer in the range 0 to 255, or a range of integers separated by a hyphen (for example 0-3). Only valid in classifiers for software QoS on egress interfaces. Default: **any** (ignores ATM VCI).
ATMVPi	The Virtual Path Identifier for an ATM connection. An integer in the range 0 to 4095, or a range of integers separated by a hyphen (for example 0-3). Only valid in classifiers for software QoS on egress interfaces. Default: **any** (ignores ATM VPI).
DLCi	The identification number of a Frame Relay Data Link Connection (DLC). An integer in the range 0 to 1023, or a range of integers separated by a hyphen (for example 0-3). Only valid in classifiers for software QoS on egress interfaces. Default: **any** (ignores DLCI).
PPPIndex	The PPP interface number. For example, for ppp2, **pppindex**=**2**.

Layer 2 parameters (ATM, frame relay and PPP) (Continued)		
PPPProtocolid	The network layer protocol of the PPP encapsulated packet. Note that network and link control packets are processed by the software QoS policy's system traffic class. Examples of control packets include NCP, LCP, IPCP and PAP. Only valid in classifiers for software QoS on egress interfaces. Default: **any**, unless you also specify a TCP or UDP parameter. Then the default is **IP**.	
	ppp-protocol-id|	A 4 byte hexadecimal protocol number. Table 2-4 on page 2-16 shows valid protocols and numbers.
	IP	Internet Protocol.
	IPV6	Internet Protocol version 6.
	ANY	The classifier ignores PPP protocol ID.
Layer 3 parameters		
IPDAddr	The destination IPv4 or IPv6 address of the packet. Default: **any**	
	ipadd[/0..32]	The destination IPv4 address, in dotted decimal notation. You can optionally specify a subnet by specifying a mask.
	ipv6add[/0..128]	The destination IPv6 address, specified as eight pairs of hexadecimal octets separated by colons. You can optionally specify a prefix length. Default prefix length is 128—a single address. IPv6 addresses are only valid in classifiers for software QoS on egress or tunnel interfaces.
	ANY	The classifier ignores destination IP or IPv6 address.
IPSAddr	The source IPv4 or IPv6 address of the packet. Default: **any**	
	ipadd[/0..32]	The source IPv4 address, in dotted decimal notation. You can optionally specify a subnet by specifying a mask.
	ipv6add[/0..128]	The source IPv6 address, specified as eight pairs of hexadecimal octets separated by colons. You can optionally specify a prefix length. Default prefix length is 128—a single address. IPv6 addresses are only valid in classifiers for software QoS on egress or tunnel interfaces.
	ANY	The classifier ignores source IPv4 or IPv6 address.
IPDScp	The DSCP value—the Code Point bits of the DiffServ field of an IPv4 or IPv6 packet. An integer in the range 0 to 63; a range of integers separated by hyphens; or a comma separated list of integers and/or ranges (for example 0,2,4-6). You can also specify EF, AF1, AF2, AF3 or AF4. **Ipdscp** and **Iptos** are mutually exclusive. Default: **any** (ignores DSCP).	
IPTOs	The TOS value—the value of the precedence field within the TOS byte of an IPv4 packet. An integer in the range 0 to 7. **Ipdscp** and **Iptos** are mutually exclusive. **Iptos** is only valid for IPv4 packets. Default: **any** (ignores TOS).	

Layer 3 parameters (Continued)	
IPFRAg	Whether the IPv4 packet is fragmented. Only valid in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores whether the packet is fragmented).
IPOPtions	Whether the packet includes the IPv4 header options field. Only valid in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores whether the header options field is present or not).
IPFLowlabel	The IPv6 flow label in an IPv6 packet, an integer in the range 0 to 1048575. Only valid for IPv6 packets in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores IPv6 flow label).
IPXDAddr	The destination network address of an IPX packet, expressed as a 4 byte hexadecimal number. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores destination IPX address).
IPXPacket	The value of the Packet Type field of an IPX packet. One of the options NLSp, RIP, SAP, SPX, NCP, or NETbios; or a 2 byte hexadecimal IPX packet number; or a recognised IPX packet type. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores IPX packet type).
IPPRotocol	The Layer 4 IPv4 or IPv6 protocol of the packet. For IPv6 packets, **ipprotocol** matches against the Next Header field of the IPv6 packet header. You can use a total of 29 unique **ipprotocol** values, plus TCP and UDP, in total across all classifiers. Default: ● **tcp** if you also specify a TCP parameter (for example, **tcpsport**). ● **udp** if you also specify a UDP parameter (for example, **udpsport**). ● Otherwise, **any** (ignores IP protocol).

ip-protocol	A 1 byte decimal IPv4 or IPv6 protocol number or a well-known protocol name.
TCP	Transmission Control Protocol.
UDP	User Datagram Protocol.
NONTcpudp	Any IPv4 or IPv6 protocol except TCP or UDP.
ICMP	Internet Control Message Protocol.
IGMP	Internet Group Multicast Protocol.
OSPF	Open Shortest Path First. Only valid in classifiers for software QoS on egress or tunnel interfaces.
ANY	The classifier ignores the IP protocol value.

Layer 4 parameters	
ICmptype	The ICMP message type to match against the ICMP type field in an ICMP packet header. One of the list of options, or a decimal value in the range 0 to 65535. Only valid if **ipprotocol=icmp** in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores ICMP type).
ICMPCode	The ICMP message reason code to match against the ICMP code field in an ICMP packet header. One of the list of options, or a decimal value in the range 0 to 65535. Only valid if **ipprotocol=icmp** in classifiers for software QoS on egress or tunnel interfaces. Default: **any** (ignores ICMP code).

Layer 4 parameters (Continued)	
TCPFlags	The TCP flags of the TCP/IP packet. One or a comma-separated list of the options URG, ACK, RST, SYN and FIN. Default: **any** (ignores TCP flag).
TCPDport	The destination TCP port—the value in the TCP destination port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores destination TCP port).
TCPSport	The source TCP port—the value in the TCP source port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores source TCP port).
UDPDport	The destination UDP port—the value in the UDP destination port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores destination UDP port).
UDPSport	The source UDP port—the value in the UDP source port field of the packet. For classifiers for hardware filters or switch port QoS, a single port number. For classifiers for software QoS on egress or tunnel interfaces, a single port number or a range of port numbers separated by a hyphen. Default: **any** (ignores source UDP port).
IPXDSocket	The destination IPX socket number of an IPX packet. One of the options NCP, SAP, RIP, NNB, DIAg, NLSp or IPXwan; or a 2 byte hexadecimal IPX socket number; or a recognised IPX socket type. You can use a total of 7 unique **ipxdsocket** values across all classifiers, including any of the options. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores destination IPX socket).
IPXSSocket	The source IPX socket number of an IPX packet. One of the options NCP, SAP, RIP, NNB, DIAg, NLSp or IPXwan; or a 2 byte hexadecimal IPX socket number; or a recognised IPX socket type. You can use a total of 7 unique **ipxssocket** values across all classifiers, including any of the options. Only valid in classifiers for hardware filters and QoS on switch ports. Default: **any** (ignores source IPX socket).
Bit matching parameters	
MATCH1 MATCH2 MATCH3	A general 16-bit word to match inside a packet, specified as a 2 byte hexadecimal number. **Match** specifies the actual data to match. You must specify all three of **match*x***, **mask*x*** and **offset*x*** together, where *x* is 1, 2 or 3. Only valid in classifiers for hardware filters and QoS on switch ports on Rapier i Series switches. Default: no default
MASK1 MASK2 MASK3	Whether the corresponding bit in **match** is "on" for a match or "don't care" for a match, specified as a 2 byte hexadecimal number. If the **mask** bit is set (on), the bit in **match** must be the same as the corresponding bit in the actual packet (so place binary "ones" in bit positions you want to match). If the **mask** bit is clear (don't care), the same bit in **match** will not be checked with the corresponding bit in the actual packet. Only valid in classifiers for hardware filters and QoS on switch ports on Rapier i Series switches. Default: no default

Bit matching parameters	
OFFSET1 OFFSET2 OFFSET3	The offset from the start of the packet, specified as an integer in the range 0 to 62. You must specify offsets in order (e.g. **offset1** before **offset2**). Only valid in classifiers for hardware filters and QoS on switch ports on Rapier i Series switches. Default: no default

* The shortest string you can enter is shown in capital letters.

Examples

To set packet matching rule 1 so that it matches all IP packets from the IP subnet 192.168.100.2 (mask=255.255.255.0), with a destination TCP port of 23, use one of the commands:

```
set class=1 ipsa=192.168.100.2/24 tcpd=23
set class=1 prot=ip ipsa=192.168.100.2/24 tcpd=23
```

To change packet matching rule 2 so that it matches traffic over ppp3, use the command:

```
set class=2 pppi=3
```

Related Commands

create classifier
destroy classifier
show classifier

show classifier

Syntax `SHow CLASSifier[=id-list|ALL|DYNAMIC]`

Description This command displays information about the classifiers configured.

If you specify **classifier** with no value, then a summary of all classifiers is displayed (Figure 2-1 and Table 2-6).

If you specify **classifier=all**, then details of all classifiers are displayed.

If you specify **classifier**=*id-list*, then details of the specified classifiers are displayed.

If you specify **classifier=dynamic**, then details of classifiers created by the DAR objects are displayed.

Figure 2-1: Example summary output from the **show classifier** command

```
Classifier General Info
-------------------------------------------------------------
 Total number of rules .... 9

 Rule ..................... 1
   Related module(s) ...... Software QoS

 Rule ..................... 2
   Related module(s) ...... Software QoS

 Rule ..................... 3
   Related module(s) ...... Software QoS

 Rule ..................... 4
   Related module(s) ...... Software QoS

 Rule ..................... 5
   Related module(s) ...... Software QoS

 Rule ..................... 6
   Related module(s) ...... None

 Rule ..................... 34
   Related module(s) ...... Software QoS

 Rule ..................... 345
   Related module(s) ...... Software QoS

 Rule ..................... 8888
   Related module(s) ...... Software QoS
-----------------------------------------
```

Table 2-6: Parameters in the summary output of the **show classifier** command

Parameter	Meaning
Rule	The identifier number for the classifier.
Related module(s)	The name of the module(s) that are currently using the classifier.

Figure 2-2: Example detailed output from the **show classifier=2,3** command for classifiers that can be applied to software QoS

```
 Classifier Rules
-------------------------------------------------------------
 Rule ................. 2
  Ingress Port ......... 1
  Egress Port .......... ANY
 Rule ................. 3
  D-MAC Address ........ ANY
  S-MAC Address ........ ANY
  M-Type ............... ANY
  S-VLAN ............... ANY
  D-VLAN ............... ANY
  E-Format ............. ANY
  Protocol ............. IP
  VLAN Priority ........ ANY
  S-IP Address ......... ANY
  D-IP Address ......... ANY
  IP flow label ........ ANY
  IP Protocol .......... ANY
  DSCP ................. 10,12,14 (AF1)
  IPOPTIONS ............ ANY
  IPFRAG ............... ANY
-------------------------------------------------------------
```

Figure 2-3: Example output from the **show classifier=dynamic** command

```
Dynamic Classifier Rules
-------------------------------------------------------------
 Rule ................. 10001
  D-MAC Address ........ ANY
  S-MAC Address ........ ANY
  M-Type ............... ANY
  VLAN ................. ANY
  S-VLAN ............... ANY
  D-VLAN ............... ANY
  E-Format ............. ANY
  Protocol ............. IPv4/IPv6
  VLAN Priority ........ ANY
  S-IP Address ......... ANY
  D-IP Address ......... ANY
  IP flow label ........ ANY
  IP Protocol .......... UDP
  TOS/DSCP ............. ANY
  IPOPTIONS ............ ANY
  IPFRAG ............... ANY
  S-UDP Port ........... ANY
  D-UDP Port ........... 42678
-------------------------------------------------------------
```

Table 2-7: Parameters in the detailed output of the **show classifier** command for classifiers that can be applied to software QoS

Parameter	Meaning
Rule	The ID number of the classifier. For classifiers for hardware filters and switch port QoS, the ID number only uniquely identifies the rule, it does not imply an ordering between rules. For software QoS, the traffic class ID number and classifier ID number together determine the rule matching order. Classifiers within each traffic class are checked in ascending order of ID number (lowest first).
Egress Port	The Ethernet switch port through which the frame is destined to leave the switch.
Ingress Port	The Ethernet switch port through which the frame arrives at the switch.
Ingress Interface	The interface through which the frame arrives at the switch.
D-MAC Address	The destination MAC address of the frame.
S-MAC Address	The source MAC address of the frame.
M-Type	The type of destination MAC address on the frame; one of L2Ucast (Layer 2 unicast addresses), L2BMcast (Layer 2 broadcast or multicast addresses) or ANY.
S-VLAN	The source VLAN—the VLAN associated with the frame when it arrives at the switch.
D-VLAN	The destination VLAN—the VLAN that the frame will be transmitted to.
E-Format	The Ethernet encapsulation type of the frame.
Protocol	The protocol, determined from the value of the following Ethernet field: ● for 802.2 (SAP encapsulation): the DSAP field, 1 byte hexadecimal ● for ETHII encapsulation: the ETYPE field, 2 bytes hexadecimal ● for NETWARERAW encapsulation: the IPX checksum field, 2 bytes hexadecimal with value FFFF ● for SNAP encapsulation: the ETYPE field, 5 bytes hexadecimal. The classifier matches on the last 2 bytes.
VLAN Priority	The 802.1p VLAN priority value in the frame.
ATM VCI	The Virtual Channel Identifier for an ATM connection.
ATM VPI	The Virtual Path Identifier for an ATM connection.
DLCI	The identification number of a Frame Relay Data Link Connection (DLC).
PPP Index	The PPP interface number. For example, for ppp2, PPP Index is 2.
PPP Protocol ID	The network layer protocol of the PPP encapsulated packet. Table 2-4 on page 2-16 shows valid protocols and numbers. Note that network and link control packets are processed by the software QoS policy's system traffic class. Examples of control packets include NCP, LCP, IPCP and PAP.
S-IP Address	The source IPv4 or IPv6 address of the packet.
D-IP Address	The destination IPv4 or IPv6 address of the packet.
IP flow label	The IPv6 flow label in an IPv6 packet.

Table 2-7: Parameters in the detailed output of the **show classifier** command for classifiers that can be applied to software QoS (Continued)

Parameter	Meaning
IP Protocol	The Layer 4 IPv4 or IPv6 protocol of the packet. For IPv6 packets, IP protocol matches against the Next Header field of the IPv6 packet header.
DSCP	The DSCP value—the Code Point bits of the DiffServ field of an IPv4 or IPv6 packet.
TOS	The TOS value—the value of the precedence field within the TOS byte of an IPv4 packet.
IPOPTIONS	Whether the packet includes the IPv4 header options field.
IPFRAG	Whether the IPv4 packet is fragmented.
ICMP Code	The ICMP message reason code to match against the ICMP code field in an ICMP packet header.
ICMP Type	The ICMP message type to match against the ICMP type field in an ICMP packet header.
S-TCP Port	The source TCP port—the value in the TCP source port field of the packet.
D-TCP Port	The destination TCP port—the value in the TCP destination port field of the packet.
TCP Flags	The TCP flags of the TCP/IP packet. One or a comma-separated list of the options URG, ACK, RST, SYN and FIN.
S-UDP Port	The source UDP port—the value in the UDP source port field of the packet.
D-UDP Port	The destination UDP port—the value in the UDP destination port field of the packet.

Figure 2-4: Example detailed output from the **show classifier** command for a classifier that can be applied to a hardware filter or switch port, for a TCP/IP data flow

```
Classifier Rules
-------------------------------------------------
 Rule ................. 1
  Ingress Port ......... 1
  Egress Port .......... 24
  S-MAC Address ........ 00-00-cd-00-03-48
  D-MAC Address ........ 00-00-cd-00-01-e4
  VLAN ................. vlan1234(1234)
  E-Format ............. ETHII
  Protocol ............. 0800(IP)
  S-IP Address ......... 192.168.123.123/32
  D-IP Address ......... 192.168.123.123/32
  IP Protocol .......... TCP
  S-TCP Port ........... 23
  D-TCP Port ........... 23
-------------------------------------------------
```

Figure 2-5: Example detailed output from the **show classifier** command for a classifier that can be applied to a hardware filter or switch port, for a UDP/IP data flow

```
Classifier Rules
-------------------------------------------------
 Rule ................. 21
  Ingress Port ......... 1
  Egress Port .......... 24
  S-MAC Address ........ 00-00-cd-00-03-48
  D-MAC Address ........ 00-00-cd-00-01-e4
  VLAN ................. vlan1234(1234)
  E-Format ............. ETHII
  Protocol ............. 0800(IP)
  S-IP Address ......... 192.168.123.123/32
  D-IP Address ......... 192.168.123.123/32
  IP Protocol .......... UDP
  S-UDP Port ........... 23
  D-UDP Port ........... 23
-------------------------------------------------
```

Table 2-8: Parameters in detailed output from the **show classifier** command for a classifier that can be applied to a hardware filter or switch port, for a TCP or UDP/IP data flows

Parameter	Meaning
Rule	The rule identifier for the packet matching rule/classifier.
Ingress Port	The number of the ingress switch port associated with the rule.
Egress Port	The number of the egress switch port associated with the rule.
VLAN	The name of a VLAN with only the first 10 characters shown. The VLAN Identifier appears in brackets. If the packet is Layer 3 switched, this VLAN is the destination VLAN. If the packet is Layer 3 routed by the CPU, this VLAN is the source VLAN. If the packet is Layer 2 switched, its source and destination VLAN are the same.
D-MAC Address	The destination MAC address field of a packet.
S-MAC Address	The source MAC address field of a packet.
E-Format	The Ethernet encapsulation format for the packet, suffixed with "-TAGGED" or "-UNTAGGED" if this has been specified in the rule.
Protocol	The hexadecimal value of the protocol. If the protocol is not for the general family of IP and IPX protocols, and the commonly known name for the protocol is known to the Classifier, then this commonly known name will be printed in brackets after the hexadecimal number.
S-IP Address	The source IP address field of a packet.
D-IP Address	The destination IP address field of a packet.
IP Protocol	The Layer 4 IP protocol field of a packet.
TOS/DSCP	The IP TOS or DiffServ Code Point field of a packet.
S-TCP Port	The source TCP/IP port field of a packet.
D-TCP Port	The destination TCP/IP port field of a packet.
S-UDP Port	The source UDP/IP port field of a packet.
D-UDP Port	The destination UDP/IP port field of a packet.
TCP Flags	A series of letters representing the TCP/IP flag field, one of URG, ACK, RST, SYN or FIN.

Figure 2-6: Example detailed output from the **show classifier** command for a MAC-address-based classifier that can be applied to a hardware filter or switch port

```
Classifier Rules
-------------------------------------------------------
 Rule ............................. 2222
  VLAN ....... vlan1234(1234)
  E-Format ......................... SNAP
  D-MAC Address ..................... aa-bb-cc-dd-ee-ff
  S-MAC Address ..................... aa-bb-cc-dd-ee-ff
  Protocol ......................... 1234567890
-------------------------------------------------------
```

Table 2-9: Parameters in the detailed output from the **show classifier** command for a MAC-address-based classifier that can be applied to a hardware filter or switch port

Parameter	Meaning
Rule	The rule identifier for the packet matching rule/classifier.
VLAN	The name of a VLAN with only the first 10 characters shown. The VLAN Identifier appears in brackets. If the packet is Layer 3 switched, this VLAN is the destination VLAN. If the packet is Layer 3 routed by the CPU, this VLAN is the source VLAN. If the packet is Layer 2 switched, its source and destination VLAN are the same.
D-MAC Address	The destination MAC address field of a packet.
S-MAC Address	The source MAC address field of a packet.
E-Format	The Ethernet encapsulation format for the packet, suffixed with "-TAGGED" or "-UNTAGGED" if this has been specified in the rule.
Protocol	The hexadecimal value of the protocol. If the protocol is not for the general family of IP and IPX protocols, and the commonly known name for the protocol is known to the Classifier, then this commonly known name will be printed in brackets after the hexadecimal number.

Figure 2-7: Example detailed output from the **show classifier** command for a classifier that can be applied to a hardware filter or switch port, for an IPX data flow

```
Classifier Rules
---------------------------------
 Rule .................. 31
  VLAN .................. ANY
  E-Format .............. ANY
  Protocol .............. IPX
  D-IPX Address ........ ANY
  D-IPX Socket ......... RIP
  S-IPX Socket ......... ANY
---------------------------------
```

Table 2-10: Parameters in the detailed output from the **show classifier** command for a classifier that can be applied to a hardware filter or switch port, for an IPX data flow

Parameter	Meaning
Rule	The rule identifier for the packet matching rule.
VLAN	The name of a VLAN with only the first 10 characters shown. The VLAN Identifier appears in brackets. If the packet is Layer 3 switched, this VLAN is the destination VLAN. If the packet is Layer 3 routed by the CPU, this VLAN is the source VLAN. If the packet is Layer 2 switched, its source and destination VLAN are the same.
E-Format	TThe Ethernet encapsulation format for the packet, suffixed with "-TAGGED" or "-UNTAGGED" if this has been specified in the rule.
Protocol	The hexadecimal value of the protocol, its common name, or both.
D-IPX Address	The destination IPX network address field of a packet.
D-IPX Socket	The destination IPX socket field of a packet.
S-IPX Socket	The source IPX socket field of a packet.

Examples

To display the number of each of the classifiers and which module is using each classifier, use the command:

```
sh class
```

To display what each classifier matches against, use the command:

```
sh class=all
```

Related Commands

create classifier
destroy classifier
set classifier

Chapter 3

Software Quality of Service (QoS)

Introducing QoS 3-4
How a QoS Solution is Described in this Software Reference 3-5
Introducing Software QoS 3-5
When to use Software QoS 3-6
Separate Traffic—Separate Needs 3-6
Applying QoS in a Network 3-7
Local Level 3-7
Domain Level: DiffServ, TOS and 802.1p Priority 3-7
Hierarchy of a Software QoS Solution 3-10
Traffic Class Trees 3-11
Traffic Classes 3-13
Policies 3-14
Order of Classifier Matching 3-14
Dynamic Application Recognition for Voice and Video 3-15
Software QoS Processing Points 3-16
Ingress QoS 3-18
Egress QoS 3-18
Tunnel QoS 3-18
Stages of a Software QoS Solution 3-19
Packet flow 3-19
Classification: Identifying and sorting traffic 3-21
Bandwidth class 3-21
Premarking: Labelling packets before metering 3-21
Metering: Bandwidth conformance 3-22
Packet queuing 3-25
RED Curves 3-25
Dequeuing 3-27
Queue Scheduling 3-30
Remarking 3-31
Virtual Bandwidth 3-32
How to Configure a Software QoS Hierarchy 3-33
The Total Software QoS Solution 3-33
Default traffic class 3-36
How to Configure the Stages of a QoS Solution 3-37
Premarking 3-37
Metering 3-38
RED 3-41
Remarking 3-42
Queue scheduling 3-44
How to Configure DAR for Voice and Video Traffic 3-48
How to Configure Software QoS on Particular Interfaces 3-50

PPP and PPPoE 3-50
Frame Relay 3-53
The Switch Instance 3-54
How to Configure Software QoS on Tunnels 3-56
VPN 3-56
6 to 4 3-57
Generic Router Encapsulation (GRE) 3-58
Interaction with Other Modules 3-59
Network Address Translation (NAT) 3-59
Resource Reservation Protocol (RSVP) 3-60
Priority Filters 3-60
Policy Filters 3-60
Bandwidth Limiting on Ethernet Interfaces 3-60
Counters 3-60
Debugging 3-61
Network Configuration Examples 3-64
1: Guaranteeing VoIP Traffic 3-64
2: Guaranteeing VoIP Traffic using DAR 3-67
3: Guaranteeing VoIP Traffic While Maintaining File Server Traffic 3-70
4: Guaranteeing VoIP Traffic over a VPN Tunnel 3-73
5: VoIP, Critical Database, and File Server Traffic 3-78
6: Multiple Applications over Frame Relay 3-81
Command Reference 3-86
add sqos interface dar 3-86
add sqos policy trafficclass 3-88
add sqos trafficclass classifier 3-89
add sqos trafficclass dar 3-90
add sqos trafficclass subclass 3-91
create sqos dar 3-93
create sqos dscpmap 3-95
create sqos meter 3-96
create sqos policy 3-99
create sqos red 3-102
create sqos trafficclass 3-104
delete sqos interface dar 3-109
delete sqos policy trafficclass 3-110
delete sqos trafficclass classifier 3-111
delete sqos trafficclass dar 3-112
delete sqos trafficclass subclass 3-113
destroy sqos dar 3-114
destroy sqos dscpmap 3-114
destroy sqos meter 3-115
destroy sqos policy 3-115
destroy sqos red 3-116
destroy sqos trafficclass 3-116
disable sqos 3-117
disable sqos debug 3-118
enable sqos 3-119
enable sqos debug 3-120
purge sqos 3-121
reset sqos counters 3-122
set sqos dar 3-124
set sqos dscpmap 3-126
set sqos interface 3-128
set sqos meter 3-130
set sqos policy 3-133
set sqos red 3-136
set sqos trafficclass 3-138
show sqos 3-143

show sqos counters 3-145
show sqos dar 3-151
show sqos dscpmap 3-153
show sqos interface 3-155
show sqos meter 3-157
show sqos policy 3-159
show sqos red 3-163
show sqos trafficclass 3-166

Introducing QoS

Quality of Service refers to the ability to intelligently manage network traffic to allow stable and predictable end-to-end network performance. It helps you achieve either or both of the following fundamental aims:

- providing sensitive traffic with the network resources it needs even when the network is congested—including traffic that is sensitive to delay, jitter and packet loss. To achieve this, the switch will delay or drop non-sensitive traffic, or traffic which exceeds the bandwidth it is entitled to.
- guaranteeing and/or limiting the resources available to a particular customer or traffic type.

The concept of QoS is a departure from the "best effort" approach to data networking, which treats all traffic on the Internet or within a LAN the same. Without QoS, the switch is equally likely to drop every different traffic type when a link becomes oversubscribed. With QoS, the switch can give preferential treatment to a subset of traffic. It does this by sorting packets according to criteria you set, measuring the bandwidth the packets are using, assigning them to an appropriate queue or dropping them, and then scheduling the transfer of packets from queues onto the wire. The switch can also mark packets so that downstream routers or switches know how to process them, and act on the marking from an upstream router or switch.

Quality of service mechanisms allow:

- traditional voice and data carriers to effectively compete against aggressive competition from wireless, satellite, and cable providers through the ability to integrate and deliver voice, video, and data services over a single network
- network service providers to sell different levels of service to customers, based on what customers require, and be confident in their ability to guarantee the reliable delivery of these services
- enterprise and educational organisations to actively manage and provide many services across one network, for example live video streaming and standard data services, with preferential treatment given for mission-critical traffic
- network administrators to manage network congestion as network traffic levels increase and time-critical applications, such as streaming media, become more widely in demand by customers and organisations

How a QoS Solution is Described in this Software Reference

Configuring Quality of Service involves separate stages that are described in different chapters of the Software Reference.

The two stages are:

1. Classifying traffic into flows, according to a wide range of criteria.

   Classification is performed by the switch's packet classifier and is not described in this chapter, but in the Generic Packet Classifier chapter.

2. Acting on these traffic flows.

   - For QoS on traffic ingressing a switch port on a Rapier i Series switch, the approaches, methods, and commands are described in the Quality of Service chapter.
   - For filters on traffic ingressing a switch port, the approaches, methods and commands are described in the Switching chapter.
   - For acting on traffic ingressing or egressing other interfaces, including WAN interfaces and Layer 3 tunnels, the approaches, methods, and commands for this are described in this chapter.

   The implementations of QoS on switch ports and software QoS are similar, but not identical.

Introducing Software QoS

Software QoS refers to QoS performed by software in the switch's CPU, rather than by a switching ASIC. It can apply to traffic over most WAN interfaces, plus IPv6, IPsec and GRE tunnels. Table 3-1 lists the available interfaces.

For the Layer 2 interfaces (eth, PPP, FR and ATM) most software QoS processing occurs as part of sending the traffic out. You can also use QoS to drop or prioritise traffic as soon as it arrives at the switch, to reduce the probability of packet loss at a congested ingress interface.

Table 3-1: Interface types for software QoS

Interface, tunnel or policy type	Example
ETH ports (but not individual switch ports)	eth0
PPP interfaces	ppp0
Frame Relay interfaces	fr0
ATM interfaces	atm0.0
the switch instance on AR400 series and AR750S routers (all switch ports as a unit)	swi0
6 to 4 tunnels	virt0
IPsec tunnels	ipsec-CentralOffice
GRE tunnels	gre1

When to use Software QoS

Software QoS may benefit your network if:

- Traffic rates over an interface are too high, and therefore:
  - high-priority traffic is being dropped
  - delay-sensitive traffic, such as VoIP traffic, is being delayed
  - jitter-sensitive traffic, such as streaming video, is experiencing variable gaps between packets.
- Network congestion is occurring at other devices in your network which have no or minimal QoS capability. You can slow traffic down or mark it with priority information as it leaves the switch so that downstream devices are not overwhelmed.
- You want to control the bandwidth available to particular users, depending on their required level of service.

The quality improvements are greatest for slower interface types.

Software QoS will be of no benefit, and may reduce overall performance, if:

- The network is not congested. Prioritising traffic is only useful if the target traffic is otherwise dropped or delayed unacceptably.
- All traffic has equal priority or is equally sensitive to delay, jitter and loss.
- Your network is so congested that not all target traffic will get processed adequately even with QoS. In this case, the only solution is to upgrade the network infrastructure.

Separate Traffic—Separate Needs

Separate traffic has separate needs. Deciding which type of service is appropriate to deploy in the network depends on the service needs. For example, interactive voice and video requires high priority, low latency, low jitter, and controlled bandwidth. Table 3-2 describes different types of service and their requirements.

Table 3-2: The QoS requirements of different types of traffic.

This service	Requires
Interactive voice and video conferencing	High priority, low latency, low jitter, controlled bandwidth
Client-server applications	High priority, low latency, low loss
Streaming audio and video	Medium priority, low jitter
Network control traffic	High priority, controlled bandwidth
Circuit emulation	Guaranteed, but controlled bandwidth
Everything else—Best effort	Low priority, long queues

Applying QoS in a Network

Broadly speaking, there are two major scenarios for applying QoS in a network: local-level and domain-level.

Local Level

You can configure QoS on the switch as a local "action" which only affects only the flow of data from the switch. This approach is suitable for many networks, for example those with a single switch, or a single bottleneck, and which do not have to conform to a Service Level Agreement (SLA).

The QoS solution takes immediate action on the traffic passing through the switch, directly affecting the flow of data. The profile of the traffic exiting the switch reflects the QoS policy but the transmitted packets do not carry any QoS information to be used by the next-hop device.

Domain Level: DiffServ, TOS and 802.1p Priority

Alternatively, you can deploy a QoS solution across an entire domain. The domain's QoS schema is designed for the whole domain so that the per-hop behaviour within the domain is consistent with the requirements that the schema serves. Packets that enter at the domain's edge may not carry any QoS information, but the edge device places such information into the packets before transmitting them to the next node in the domain. Thus, QoS information is preserved between nodes within the domain and the nodes treat the packets accordingly.

There are three main options for preserving QoS information:

- the 802.1p priority field within the VLAN tag of tagged Ethernet packets (see Figure 3-1)
- the IP Type of Service (TOS) field
- the Differentiated Services (DiffServ) Code Point (DSCP).

TOS and DSCP are mutually exclusive, and TOS is not available on IPv6 packets.

DiffServ is a method of dividing IP traffic into classes of service without requiring that every in a network remember detailed information about traffic flows. Switches within a DiffServ domain process traffic on the basis of the DSCP (DiffServ Code Point) value in the IP header's Differentiated Services (DS) field[1] (see Figure 3-2).

1. The Differentiated Services field supersedes the IPv4 Type of Service (TOS) field and the IPv6 Traffic Class field.

Figure 3-1: The VLAN tag field in Ethernet packets

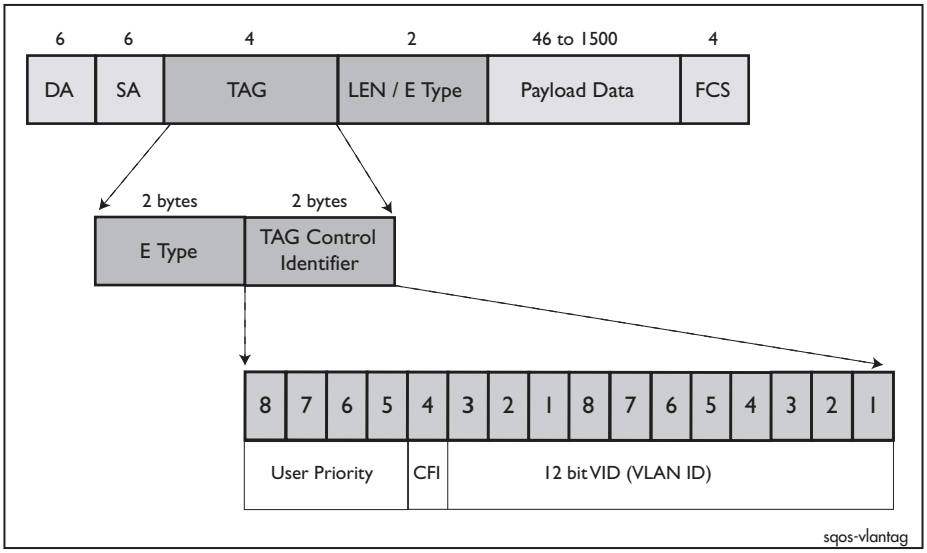


Figure 3-2: The DSCP bits of the DS field in the IPv4 header.

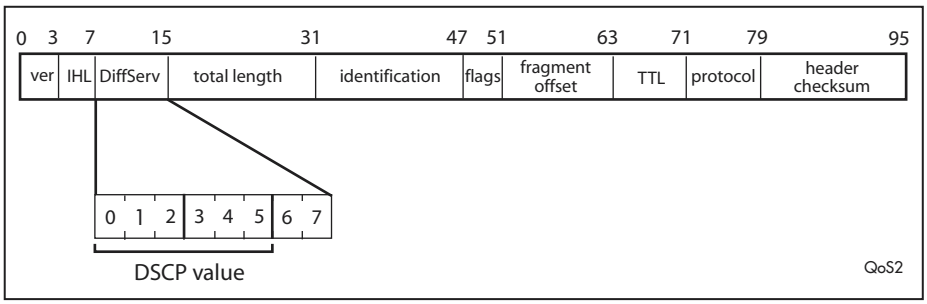
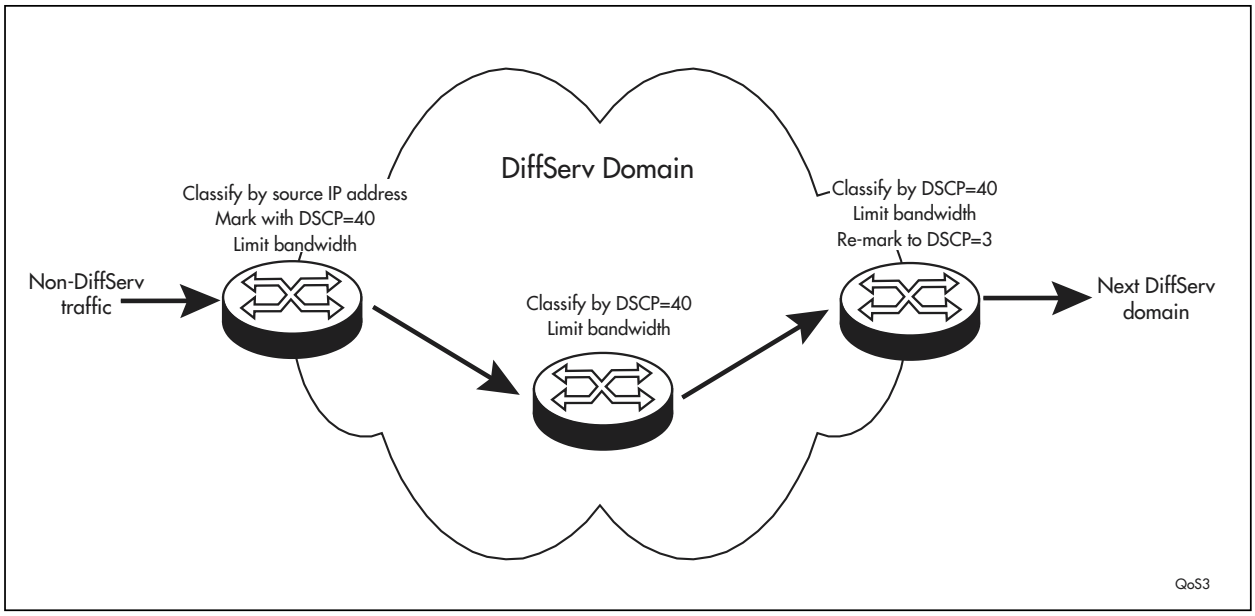


Figure 3-3 shows a simple example of a DiffServ domain. Packets that originate from a particular IP address have their IP headers marked with a Differentiated Services Code Point (DSCP) value of 40 when they arrive at the edge of the domain. This DSCP value is preserved in the packets as they are sent to the next node. The next node classifies packets into flows according to their incoming DSCP and applies appropriate QOS functionality to them.
As the packets exit the domain they are re-marked with a different DSCP value, which will be read and acted upon at the edge of the next QoS domain.

In this example the DSCP is determined by the nature of the packet (its source IP address). The DSCP can instead be determined by bandwidth conformance, so that packets which exceed their flow's allowed bandwidth at one node are treated differently by later nodes.

For more information about DiffServ see RFC 2474, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*.

Figure 3-3: An example of a DiffServ domain.



Assured Forwarding and Expedited Forwarding

Assured Forwarding (AF) provides predefined DSCP values for four classes of service. Each class has DSCP values that correspond to a high drop probability, a medium drop probability and a low drop probability. You can use AF in a DiffServ domain to offer different classes of service, in which traffic which conforms with its bandwidth allocation has a lower drop probability than partially- or non-conformant traffic. AF is described in RFC 2597, *Assured Forwarding PHB Group*, June 1999.

Expedited Forwarding (EF) also provides a predefined DSCP value, for a single "premium" service. You can use EF in a DiffServ domain to offer a low loss, low latency, low jitter, assured bandwidth, end-to-end service. EF is described in RFC 2598, *An Expedited Forwarding PHB*, June 1999.

The DSCP values corresponding to each of these services are shown in Table 3-3, in binary and decimal.

Table 3-3: DSCP and TOS values for Assured Forwarding and Expedited Forwarding

Per Hop Behaviour (PHB)	Class	DSCP value (AF/EF name, then binary, then decimal)			IP TOS precedence value
Default		000000 0			0
Assured Forwarding		**Low drop prob**	**Medium drop prob**	**High drop prob**	
	Class 1	AF11 001010 10	AF12 001100 12	AF13 001110 14	1
	Class 2	AF21 010010 18	AF22 010100 20	AF23 010110 22	2
	Class 3	AF31 011010 26	AF32 011100 28	AF33 011110 30	3
	Class 4	AF41 100010 34	AF42 100100 36	AF43 100110 38	4
Expedited Forwarding		EF 101110 46			5

Hierarchy of a Software QoS Solution

For each interface or tunnel, you need to build up your software QoS solution out of:

- A **policy**, which you attach to the interface. The policy defines a complete QoS solution for all traffic on the interface or group of interfaces.
- **Traffic classes** within the policy. Each traffic class defines the QoS processing for a group of traffic flows. The traffic flows for each traffic class are identified by classifiers or sub traffic classes attached to the traffic class. In software QoS, queues are contained within traffic classes.
- **Classifiers**, to sort traffic into the appropriate traffic classes. Classification is simply a method of dividing the incoming traffic into traffic flows so that packets of one type can be treated differently to packets of another type. The switch supports two types of classifiers:
  - static classifiers, which can identify packets by a large number of characteristics. Available classification options depend on the interface and traffic type, and range from layer 1 and 2 features (for example, destination VLAN), to layer 3 features (for example, destination IP address) and layer 4 features (for example, destination TCP port).
  - Dynamic Application Recognition objects, which identify and sort VoIP and video traffic.

Traffic Class Trees

Definition

Each software QoS policy contains a **traffic class tree**, which provides hierarchical queue scheduling. Depending on the policy settings and the types of traffic class in the tree, the policy empties queues using priority queuing (PQ), weighted round robin (WRR), deficit weighted round robin (DWRR) or mixed scheduling (PQ plus WRR or DWRR). For information about these queue scheduling methods, see "Queue Scheduling" on page 3-30.

The first level

The first (top) level of the traffic class tree is made up of three weighted traffic classes:

- A **system** traffic class for important system traffic. On egress and tunnel policies, this includes ARP, RIP, RIPv2, BGP, OSPF, IPv6 control packets such as ND and NS, PPP control packets, ISAKMP, keepalive messages, and SNMP messages generated by the switch. On ingress policies, it includes PPP control packets. The system class is a weighted traffic class with a configurable weighting, and a default weight of 20.
- A **root** traffic class. When you assign traffic classes to a policy, the switch attaches them to the root traffic class. The root class is not configurable. Its weight is calculated using the formula:

  ```
  100 - SystemClassWeight - DefaultClassWeight
  ```

  By default, this gives a root traffic class weight of:

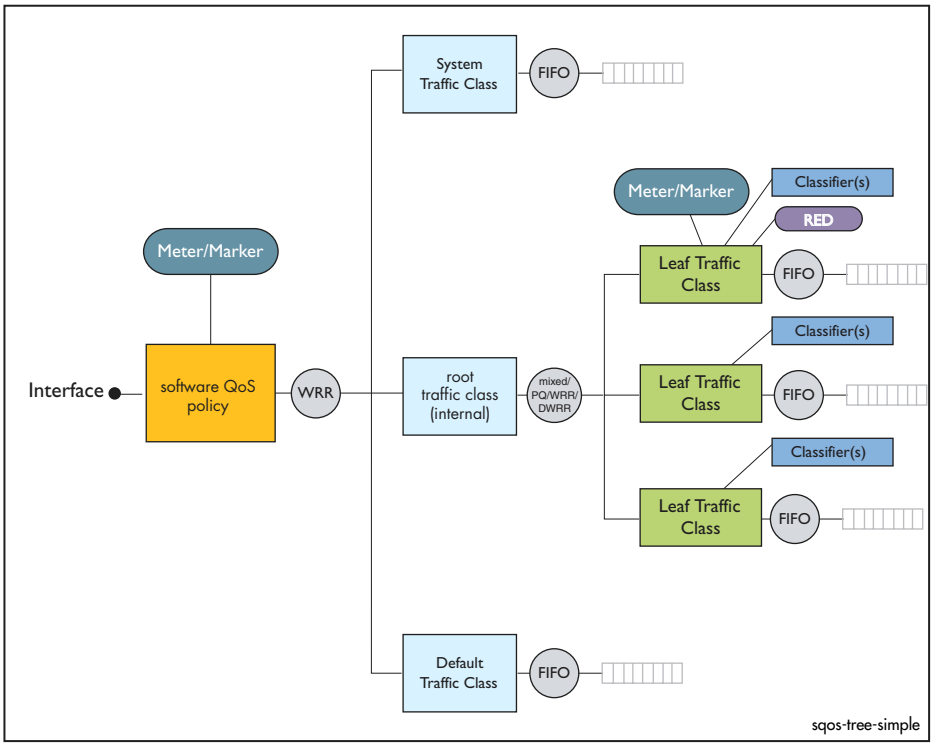
  ```
  100 - 20 - 0 = 80
  ```

- A **default** traffic class (DTC) for unclassified traffic. The DTC provides a catch-all for any traffic that does not match one of the traffic classes you assign to a policy. When you create a policy, the switch creates a default traffic class with a weight of 0. A weight of 0 means that the DTC is only emptied if all other queues are empty. If you require other behaviour, you can specify another traffic class as the default instead.

The second level

Other levels of the traffic class tree are made up of traffic classes attached to the root traffic class. Figure 3-4 shows a simple two-level traffic class tree.

Figure 3-4: A simple traffic class tree



Multi-level trees

Figure 3-5 on page 3-13 shows a more complex multi-level traffic class tree, which is used in Configuration Example "6: Multiple Applications over Frame Relay" on page 3-81 to prioritise real-time traffic while controlling file server downloads. Multi-level trees offer hierarchical queuing and bandwidth management. You can attach up to three levels of traffic class to a policy.

This chapter uses the following terms for traffic classes within complex traffic class trees:

- **Intermediate**

  A traffic class that contains one or more other traffic classes is called an *intermediate* traffic class.

- **Sub**

  A traffic class that is part of an intermediate traffic class is called a *sub* traffic class.

- **Leaf**

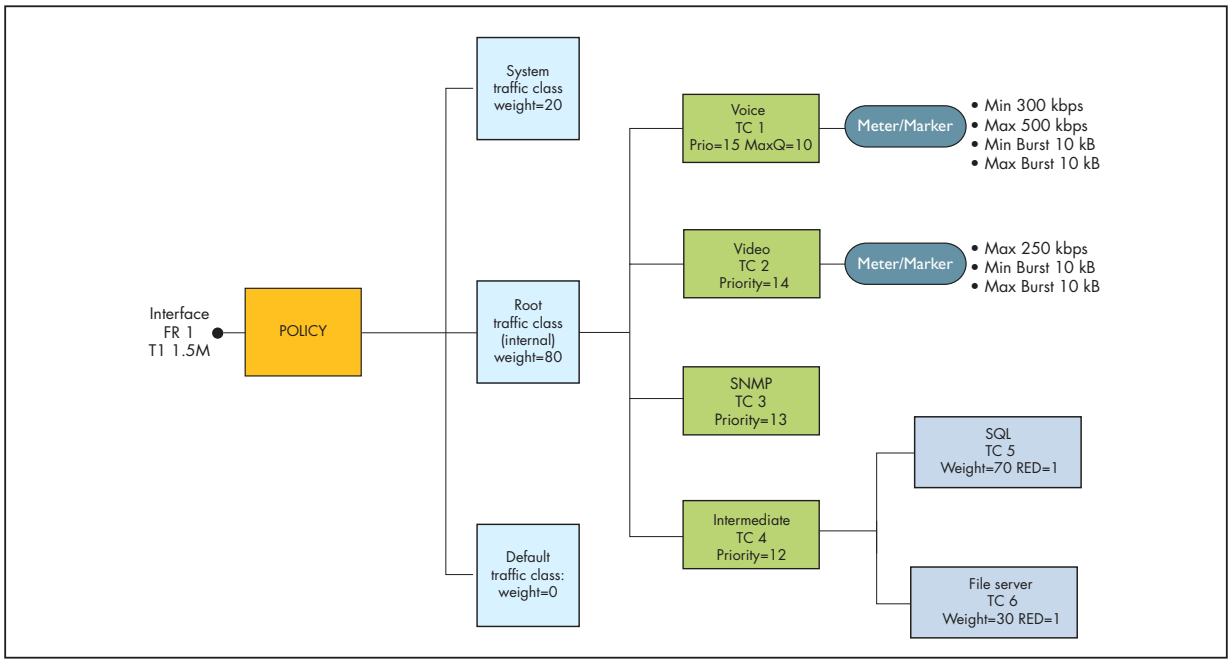
  A traffic class that contains no sub traffic classes, and that is therefore at the edge of the tree, is called a *leaf* traffic class.

- **Sibling**

  Traffic classes at the same position in the traffic class tree are called *sibling* traffic classes.

You should plan your traffic class tree so that traffic classes containing latency and jitter-sensitive traffic, such as VoIP, are not children of any weighted traffic classes (except for the root class).

Figure 3-5: A complex traffic class tree



Traffic Classes

For each traffic class, you can specify any combination of:

- the bandwidth class and DSCP that packets are given on admission to the traffic class
- the meter the traffic class uses and how the switch responds to non-conformant packets
- if and how the switch modifies the packet DSCP and/or VLAN priority when dequeuing packets. The switch can assign the DSCP on the basis of metering results.
- for leaf traffic classes, the RED curve set.
- for leaf traffic classes, the maximum queue length, notification when queue length is exceeded, total internal bandwidth available to the traffic class, and whether packets are dropped from the head or tail of the queue.
- the priority or weight of this traffic class compared with others in the policy's traffic class tree
- the scheduling method used for weighted traffic classes that are attached to this traffic class (WRR or DWRR)

These settings specify the action the switch will take at each of the software QoS processing stages described in "Stages of a Software QoS Solution" on page 3-19. Settings in a traffic class override settings in the policy.

Policies

For each policy, you can specify any combination of:

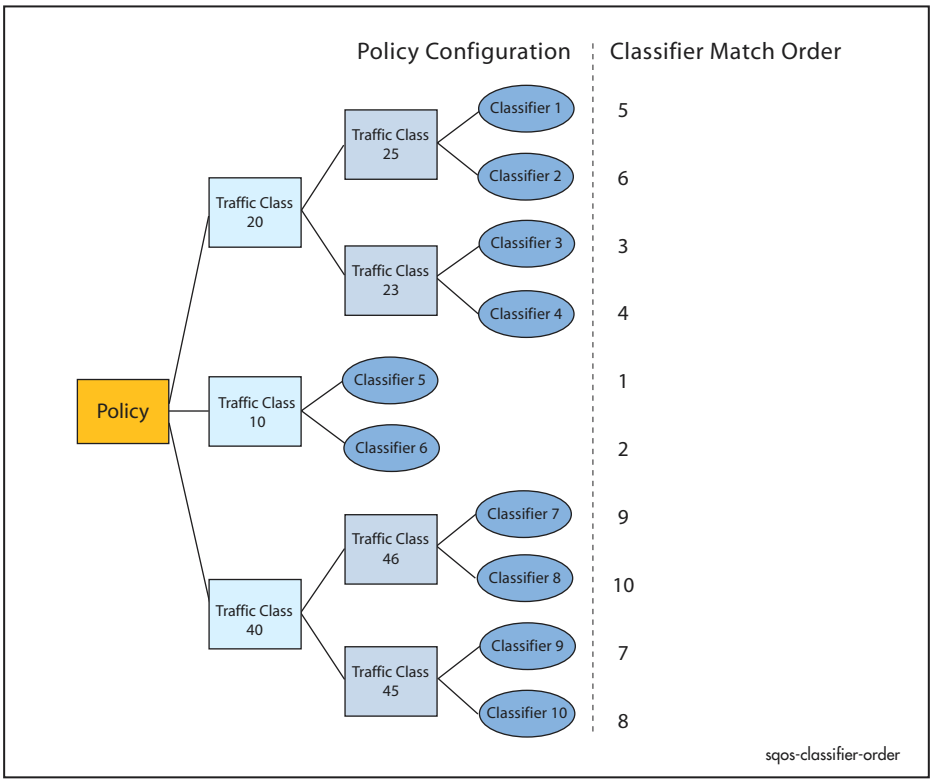
- the meter the policy uses and how the switch responds to non-conformant packets
- if and how the switch modifies the packet DSCP and/or VLAN priority when dequeuing packets
- total internal bandwidth available to the policy
- the scheduling method used for weighted traffic classes in the policy's traffic class tree
- the amount of resource given to system traffic

These settings specify the action the switch will take at each of the software QoS processing stages described in "Stages of a Software QoS Solution" on page 3-19. Settings in a traffic class override settings in the policy.

Order of Classifier Matching

Each packet is sorted according to the first classifier in the classifier list that matches it. Within each policy, classifiers are listed first by the ID number of the traffic class they are attached to, then by classifier ID number. Figure 3-6 shows the order in which classifiers are matched in an example of a traffic class tree. In this example, if a packet matched the criteria of classifier 2 and of classifier 6, then the packet would be associated to traffic class **10** because classifier 6 is above classifier 2 in the match order.

Figure 3-6: Order of classifier matching in a traffic class tree



Dynamic Application Recognition for Voice and Video

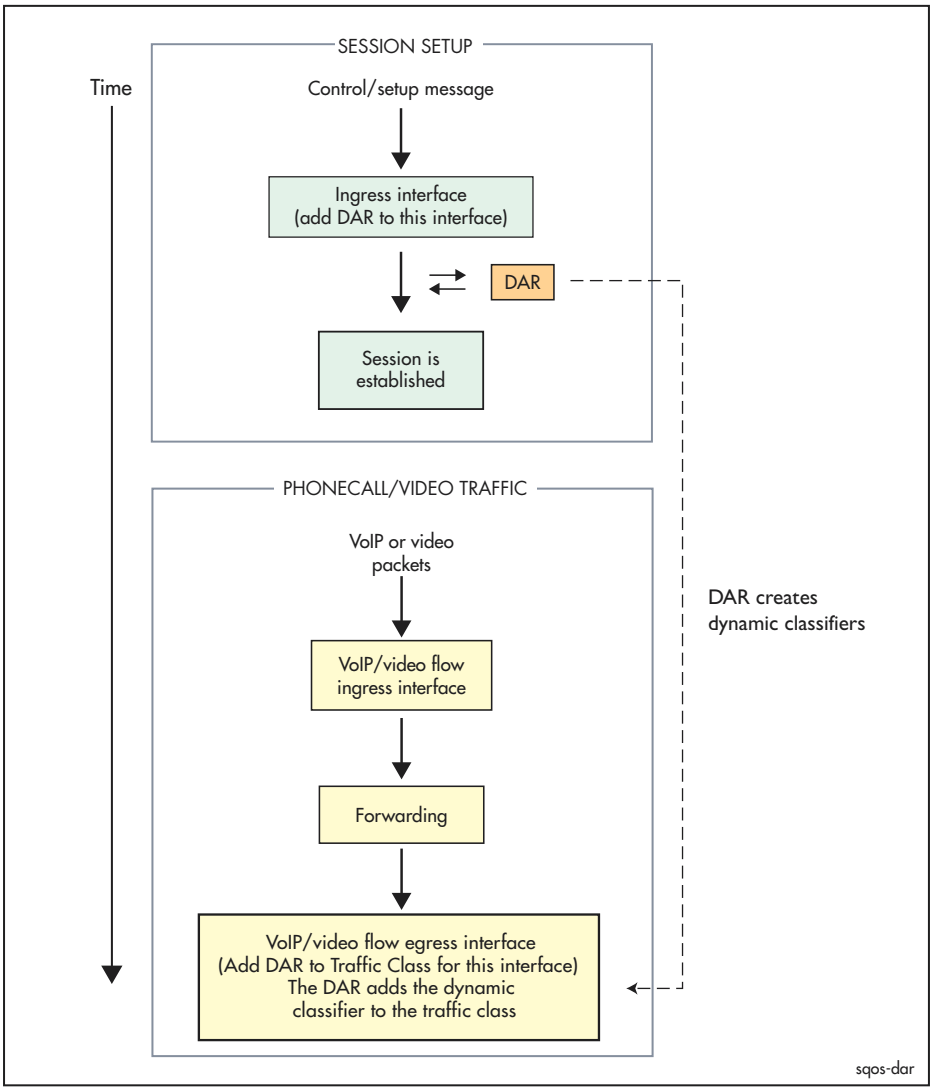
The Dynamic Application Recognition (DAR) system applies full software QoS functionality to voice and video packets, by creating dynamic classifiers.

There are two stages to the DAR process, which are shown in Figure 3-7 on page 3-15.

1. First, the switch examines incoming voice or video session initiation messages that arrive at an interface, and compares them against a DAR object. The DAR object tells the switch what kind of session to match on that interface. The switch creates a dynamic classifier to match that session, and applies it to the interface that uses a traffic class that the DAR object belongs to.
2. Second, the switch uses that dynamic classifier to sort voice or video packets into traffic classes and apply the configured QoS processing to them.

In most networks the control messages and VoIP/video traffic flow ingress the same interface, but the system does not require this.

Figure 3-7: Process flow for Dynamic Application Recognition (DAR)



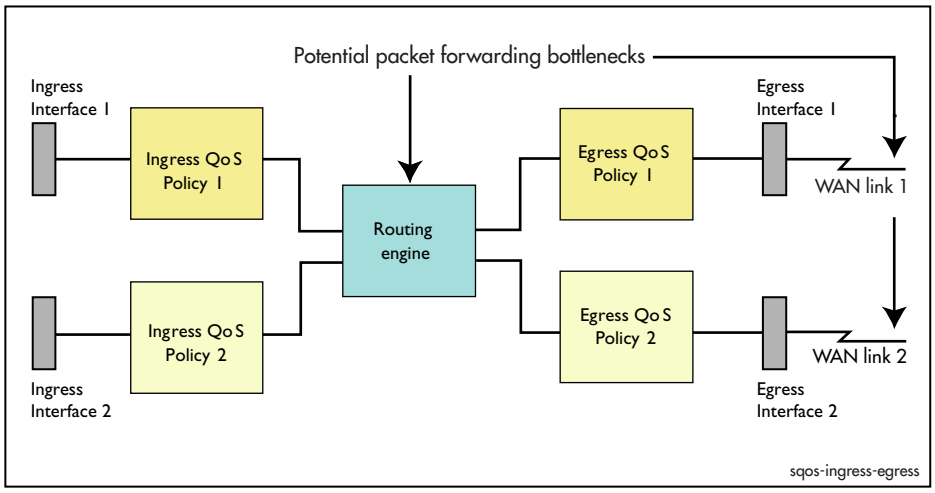
Software QoS Processing Points

The switch can perform software QoS at

- the ingress interface, immediately after the packet arrives at the switch, and/or
- the egress interface, before the packet leaves the switch, or
- the entry to a tunnel, before the packet is encapsulated.

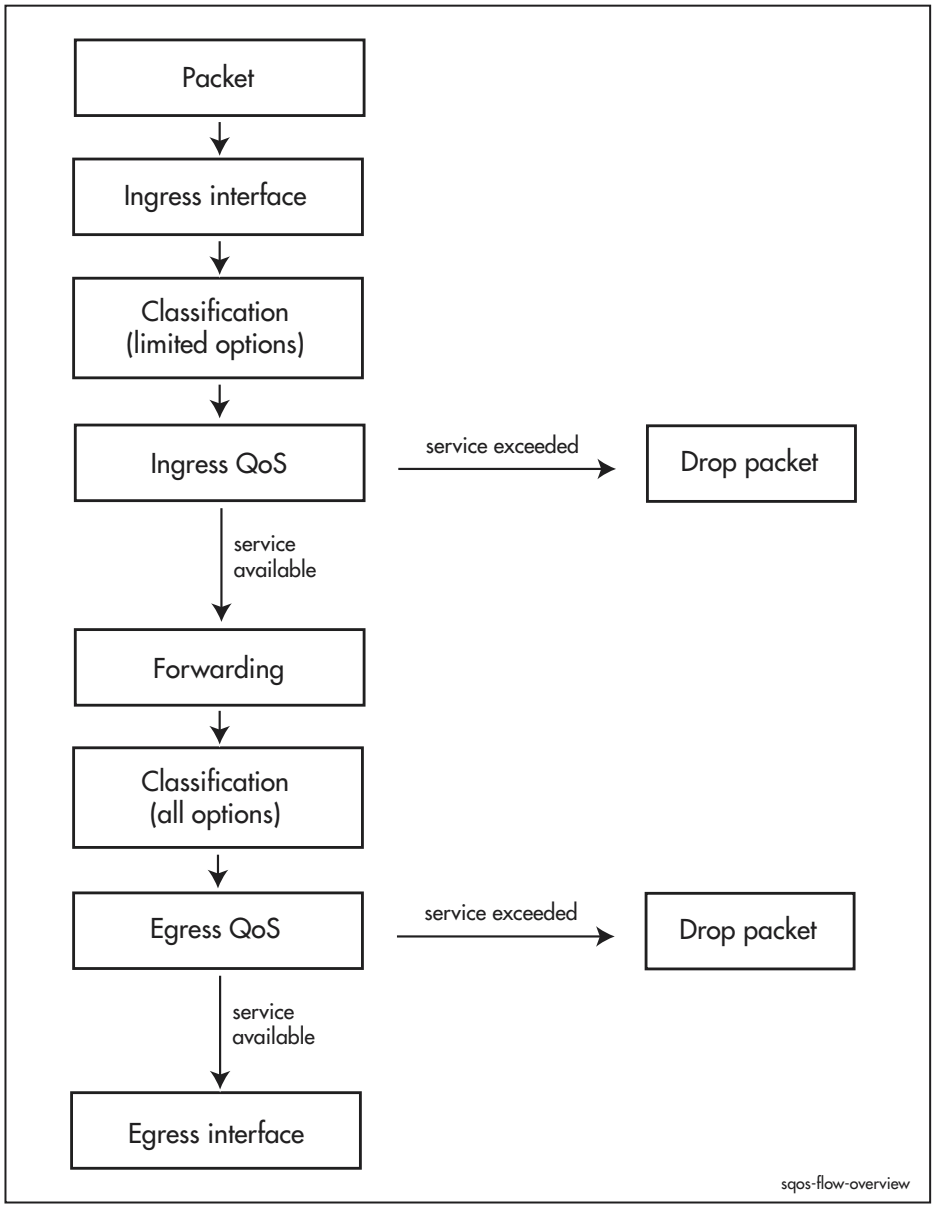
Typically you would apply ingress QoS to high-bandwidth interfaces, because they can send enough traffic to oversubscribe the switch's routing engine; and apply egress QoS to low-bandwidth interfaces, because the switch can oversubscribe those links.

Figure 3-8: The points in the switch that can be oversubscribed



The flow from ingress interface to egress interface is summarised in Figure 3-9.

Figure 3-9: A summary of the packet flow through the switch



Ingress QoS

In heavy traffic conditions, the processing within the switch's routing engine can itself be a bottleneck. Ingress QoS enables you to give some packets preferential access to the routing engine. It is designed to identify very high priority traffic and ensure the switch processes it as quickly as possible, or to identify very low priority traffic and drop it if necessary.

Ingress QoS processing adds to the load on the CPU. To minimise the impact, ingress QoS can only classify packets by their 802.1p VLAN tag user priority, or their DSCP or TOS value. However, an ingress QoS policy can contain all the queuing, metering, and marking facilities of any QoS policy, so the switch can perform all QoS functions on the classified ingress traffic. To minimise the processing overhead, we recommend that you minimise the number of QoS entities you create, and in particular that you configure as few ingress classifiers as possible.

Egress QoS

The most significant QoS processing takes place at the egress interface, on WAN links with limited bandwidth. QoS at the egress interface can classify packets according to a wide range of layer 1 to 4 characteristics. It is designed to affect the fate of the packet:

- While it is still within the switch.

  This fate can be only one of two possibilities: packets can either be dropped or placed in an egress queue. Packets that belong to an "important" traffic flow will be placed in a high priority egress queue to ensure their timely delivery. Less important packets will be placed in a lower priority queue or may even be dropped if the switch is congested.

- As it crosses the QoS domain.

  This involves permanently marking the packet in a way that downstream devices will be able to understand and use to determine the actions that they will take on the packet.

These two types of action are not mutually exclusive. A single action that is taken on a packet may well determine which egress queue it is assigned to and also determine how a downstream device will treat the packet.

Tunnel QoS

QoS at a tunnel is performed on packets before they are encapsulated and enter the tunnel. Tunnels are at layer 3 in the OSI model, so the switch can classify tunnelled traffic according to layer 3 and 4 characteristics, plus ingress interface and port. The switch can perform all QoS functions on the classified traffic.

Stages of a Software QoS Solution

This section first summarises the processes that make up a QoS solution, then discusses them in detail.

Software QoS consists of the following overarching processes:

- Classification: putting packets into traffic classes

  As described in "Hierarchy of a Software QoS Solution" on page 3-10, the switch uses static or dynamic (DAR) classifiers to identify packets.

- Enqueuing: putting packets into queues, or dropping them if the queues are oversubscribed

  Premarking, metering, RED curves, and tail or head drop interact to determine which packets get put into queues.

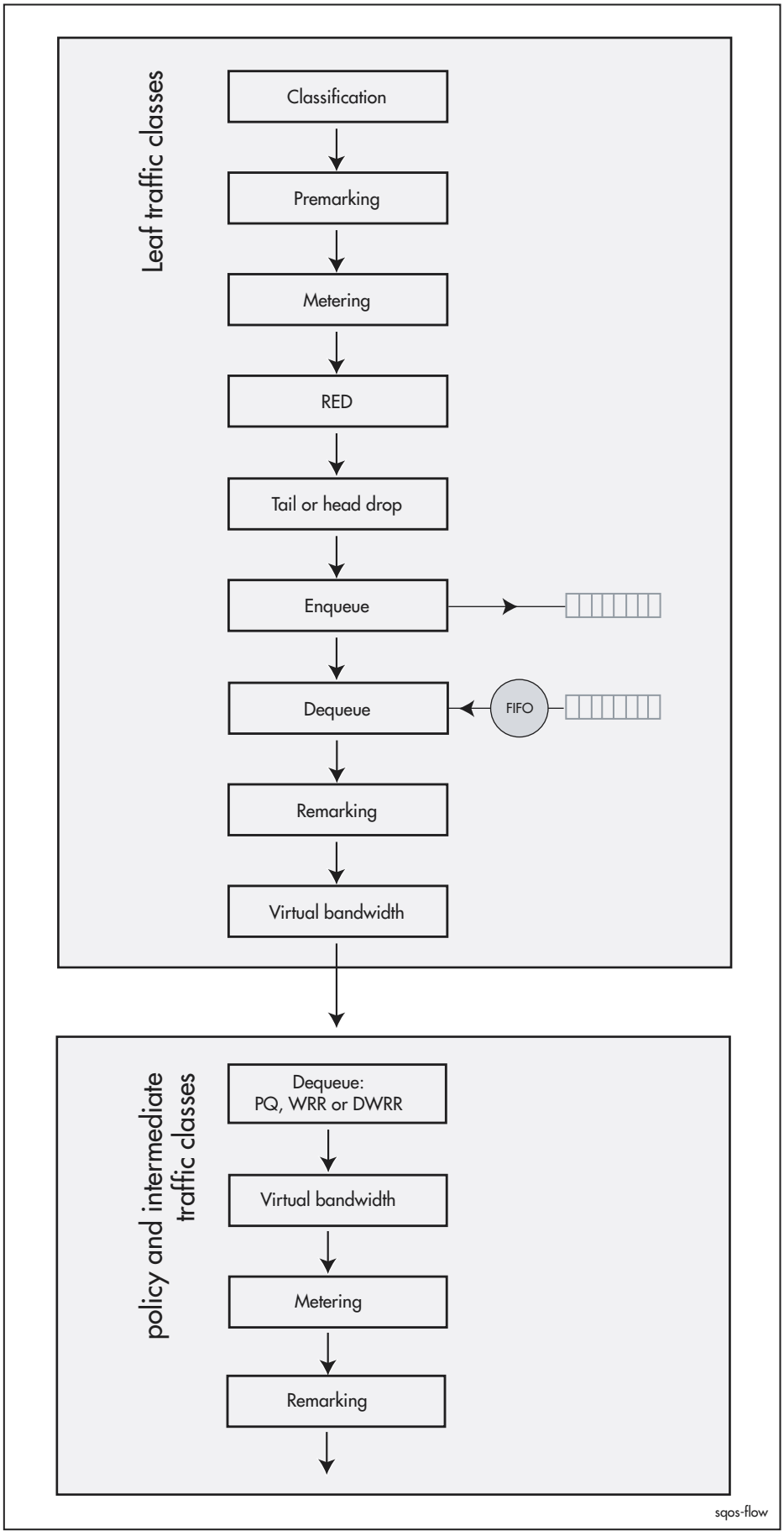
- Dequeuing: removing packets from the queues in appropriate order for transmission.

  Virtual bandwidth, metering, and remarking can apply to packets as they are dequeued.

Packet flow

Figure 3-10 on page 3-20 shows the path of a packet through the QoS engine. The diagram shows all possible QoS stages for completeness, but few QoS solutions will require every tool. As described in "Software QoS Processing Points" on page 3-16, each stage applies equally to QoS on ingress interfaces, egress interfaces and tunnels; the only difference is the characteristics on which you can classify traffic.

Figure 3-10: Detailed packet flow through the software QoS system



Classification: Identifying and sorting traffic

Once you have enabled Software QoS, all traffic is examined by the classifier, which assigns packets to a traffic class on the basis of any combination of a large number of characteristics. Therefore, the first step in QoS is to create classifiers for each type of sensitive traffic. You can also configure the switch to create dynamic classifiers for VoIP traffic, which are referred to as DAR objects.

For information about the classifier see the Generic Packet Classifier chapter. For more information about dynamic classifiers see "Dynamic Application Recognition for Voice and Video" on page 3-15.

Bandwidth class

Before and during QoS processing the switch assigns packets to bandwidth conformance class 1, 2, or 3. At the start of the QoS process, the bandwidth class may indicate how well the packet's flow conformed with its bandwidth allocation at the previous hop. This is achieved by *premarking*; see below. During the QoS process, the bandwidth class indicates how well the packet's flow conforms with its bandwidth allocation at this switch. Bandwidth class can also be described using a 3-colour model. The classes, corresponding colours and meanings are shown in Table 3-4.

Table 3-4: The meaning of each bandwidth class, for a single rate three colour marker

Bandwidth class	Colour	Meaning
1	Green	Conformant: Processing this packet leaves the flow within acceptable bandwidth limits. If this packet causes the flow to burst, the burst is acceptably low.
2	Yellow	Partially conformant: Processing this packet causes the flow to burst, but the burst is not unacceptably large.
3	Red	Nonconformant: Processing this packet causes an unacceptably high burst.

Premarking: Labelling packets before metering

Premarking assigns the packet to a bandwidth class and/or replaces the packet's initial DSCP value. It occurs immediately after classification, before the switch applies any bandwidth metering to a traffic class.

The "pre" part of premarking means this process happens before any bandwidth metering takes place, so involves no measurement of actual bandwidth use. The "marking" part refers to attaching QoS information to packets.

Premarking to assign the packet to a bandwidth class enables you to use the bandwidth conformance information from the previous hop. By default, all packets are assigned to bandwidth class 1 (green, conformant).

Premarking to set the DSCP enables you to identify or mark traffic appropriately at the edge of a DiffServ domain. For example, you can create a traffic class for each of the four AF classes, and premark packets with the appropriate DSCP for that AF class (see Table 3-3).

There are two options for specifying the new bandwidth class and DSCP:

- Directly, by specifying a new value for all packets that belong to the traffic class.
- By using the premarking table of a DSCP map. The switch reads the packet's current DSCP and looks up the table to determine the new values for that DSCP. DSCP maps enable you to premark packets with different incoming DSCPs differently.

Table 3-5: A conceptual diagram of part of a premarking table in a DSCP map

Original DSCP	New bandwidth class	New DSCP
0	newbwclass	newdscp
1	newbwclass	newdscp
. . .		
63	newbwclass	newdscp

Metering: Bandwidth conformance

Metering involves measuring how much bandwidth the packets in a traffic flow use, and how well the bandwidth use conforms with the bandwidth specifications for the traffic class that the flow belongs to. It assigns the packet to a bandwidth class depending on its conformance (Table 3-4 on page 3-21).

Note that the role of metering is to assign the packet to a bandwidth class, not to directly discard or queue it. Later QoS processes can use the metering results to decide how to process the packet.

The switch supports Single Rate Three Colour Marker meters, as described in RFC 2697, and Two Rate Three Colour Marker meters, as described in RFC 2698.

Both meters are based on the concept of a **committed** rate plus a level of committed burst, below which packets are conformant. Above this, there is a certain level of allowed excess. If the flow exceeds the committed rate and exhausts the committed burst size, but not the excess, its packets are marked partially conformant. For the single rate meter, the excess can only be a temporary burst (the **excess** burst). If packets are sent at a steady rate that exceeds the committed rate, the single rate meter will eventually mark them non-conformant. For the two rate meter, the excess can be steady. The two rate marker will only mark packets non-conformant if they exceed its second rate (the **peak** rate). Therefore the two rate marker can give you two different rate options.

Single Rate Three Colour Marker Meter

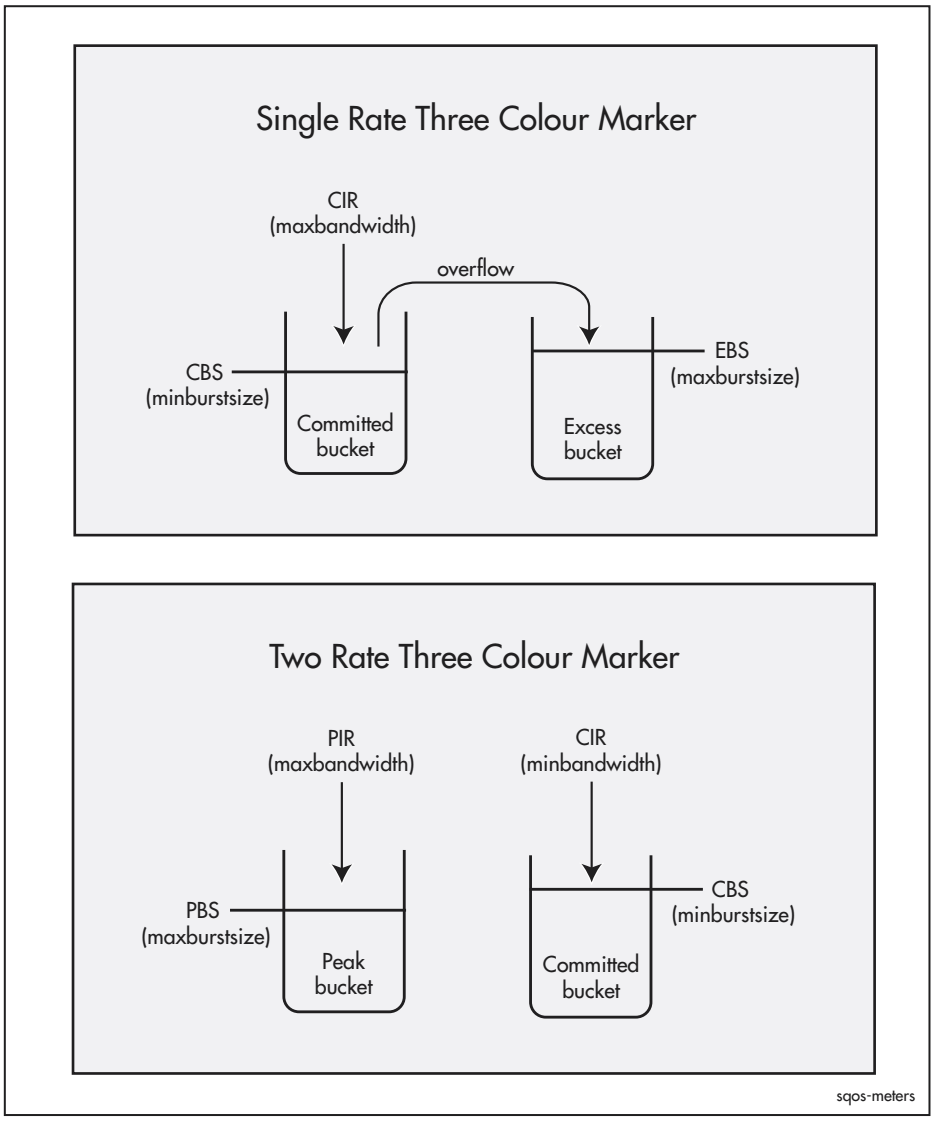
The meter is based on a token bucket model with two token buckets. Each token represents one byte. The size of one bucket is the Committed Burst Size (CBS) and the other is the Excess Burst Size (EBS). The CBS bucket is refilled at a rate called the Committed Information Rate (CIR) and the EBS bucket is refilled from the overflow of the CBS bucket. Figure 3-11 on page 3-23 shows the model and the names of the associated parameters.

Colour-blind

The meter can be either colour-blind or colour-aware. When a packet arrives at a colour-blind meter, the switch determines if the packet is smaller than the number of tokens (bytes) in the CBS bucket. If the CBS bucket contains enough tokens, the switch removes those tokens from the CBS bucket and assigns the packet to bandwidth class 1 (conformant, green). If the CBS bucket does not contain enough tokens but the EBS bucket does, the switch removes the tokens from the EBS bucket and assigns the packet to bandwidth class 2 (partially conformant, yellow). If the packet is larger than the number of tokens in either bucket, the switch assigns the packet to bandwidth class 3 (non-conformant, red).

Note that the metering process compares the number of tokens in the buckets to the size of the packet, for each packet. Therefore for metering to work logically, the buckets must be at least as big as the packets passing through the system. The burst sizes define the bucket size, so one or both of the CBS (**minburstsize**) or EBS (**maxburstsize**) must be at least as large as the packets being metered. It is usually best to configure a burst size that is several times the MTU.

Figure 3-11: Single and two rate three colour marker meters



Colour-aware

The process is similar for a colour-aware meter, except that the metering process depends on the initial bandwidth class (colour) of the packet. If the packet is in bandwidth class 3 (red) before metering, the switch leaves it in bandwidth class 3. If the packet is in bandwidth class 2 (yellow) before metering, the switch meters it from the EBS token bucket only. If the EBS bucket has enough tokens, the packet stays in bandwidth class 2; if not, the switch assigns it to bandwidth class 3. If the packet is in bandwidth class 1 (green) before metering, the switch uses both token buckets, as described above for a colour-blind meter. Both meters have the same effect on packets that were conformant at the previous switch, but you can use a colour-aware meter to stop packets that were non-conformant or partially-conformant at the previous switch from being marked conformant at this switch.

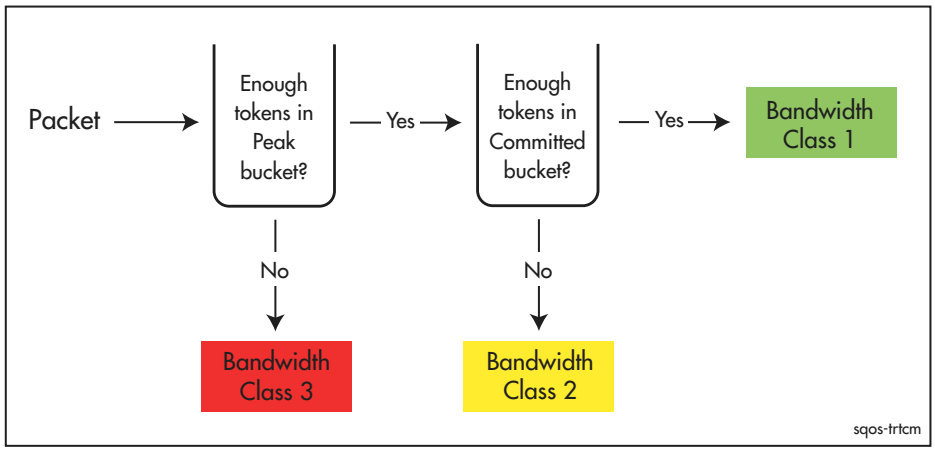
Two Rate Three Colour Marker Meter

The two rate meter also has two token buckets, but they are filled separately, and are inspected in the opposite order. The first token bucket is called the Peak bucket. It is refilled at the Peak Information Rate (PIR) to a maximum of the Peak Burst Size (PBS). The second token bucket is called the Committed bucket. It is refilled at the Committed Information Rate (CIR) to a maximum of the Committed Burst Size (CBS). Figure 3-11 on page 3-23 shows the model and the names of the associated parameters.

Figure 3-12 on page 3-24 shows the meter's decision flow. The meter first compares the packet size with the number of tokens in the Peak bucket. If the Peak bucket does not have enough tokens, the meter assigns the packet to bandwidth class 3 (non-conformant, red). If the Peak bucket has enough tokens, the meter subtracts them from the Peak bucket and then compares the packet size with the number of tokens in the Committed bucket. If the Committed bucket does not have enough tokens, the meter assigns the packet to bandwidth class 2 (partially conformant, yellow). If the Committed bucket has enough tokens, the meter subtracts the tokens and assigns the packet to bandwidth class 1 (conformant, green).

The meter only functions correctly if the Committed bucket is mostly less full than the Peak bucket. Therefore PIR must be greater than or equal to CIR.

Figure 3-12: How the two rate three colour marker assigns bandwidth class



Acting on Flow Conformance

If the meter assigns a packet to bandwidth class 3 (non-conformant), you can configure it to drop the packet. You can also configure it to pause the traffic flow by not dequeuing packets belonging to that flow for some seconds. If it pauses a traffic flow, it can produce a log message and SNMP trap.

The packet can be marked according to the metering results when it leaves the switch, by changing the DSCP. These options are described in "Remarking" on page 3-31.

Packet queuing

Each leaf traffic class contains a queue, which stores packets. The switch first enqueues packets that belong to the traffic class—puts packets into the queue. When appropriate resource becomes available, it dequeues them—removes them from the queue—and sends them out in order of priority at controlled speed. Therefore queues can reduce jitter by smoothing the traffic flow.

You can also use enqueuing to control which packets the switch drops when congested. Parameters which interact include:

- Queue length. When the queue gets too full, packets are dropped.
- RED. RED is an active queue management technique that randomly discards packets as the queue builds up. RED is described below.
- Head or tail drop. With tail drop, the switch drops packets from the tail of the queue. With head drop, the switch drops packets from the queue head. Tail dropping drops the newest packets; head dropping drops the oldest.

  Head drop may be more appropriate than tail drop for real-time applications, for example voice traffic, because the latest data is of greater interest than older data.

RED Curves

Random Early Detection/Discard (RED) is a congestion avoidance mechanism that allows the switch to drop packets randomly before the egress queue exceeds the allocated maximum queue length. It is bandwidth class aware, therefore it can drop less conformant packets when some congestion occurs, and can drop more conformant packets as congestion becomes more severe.

Each RED curve set consists of three curves, one for each bandwidth class. Generally RED curves are designed so that bandwidth class 3 packets start to be dropped when the average queue length reaches a reasonably low threshold. Bandwidth class 2 packets start to be dropped at a higher average queue-length threshold and bandwidth class 1 packets start to be dropped at a higher still threshold.

RED stops the switch from dropping bursts of packets and therefore breaks the global synchronisation of TCP flows. This maximises link utilisation. Using RED on UDP traffic flows is not recommended because UDP has no inherent congestion detection mechanism and does not react to packet drops.

For each bandwidth class *x*, the parameters used in defining a RED curve are:

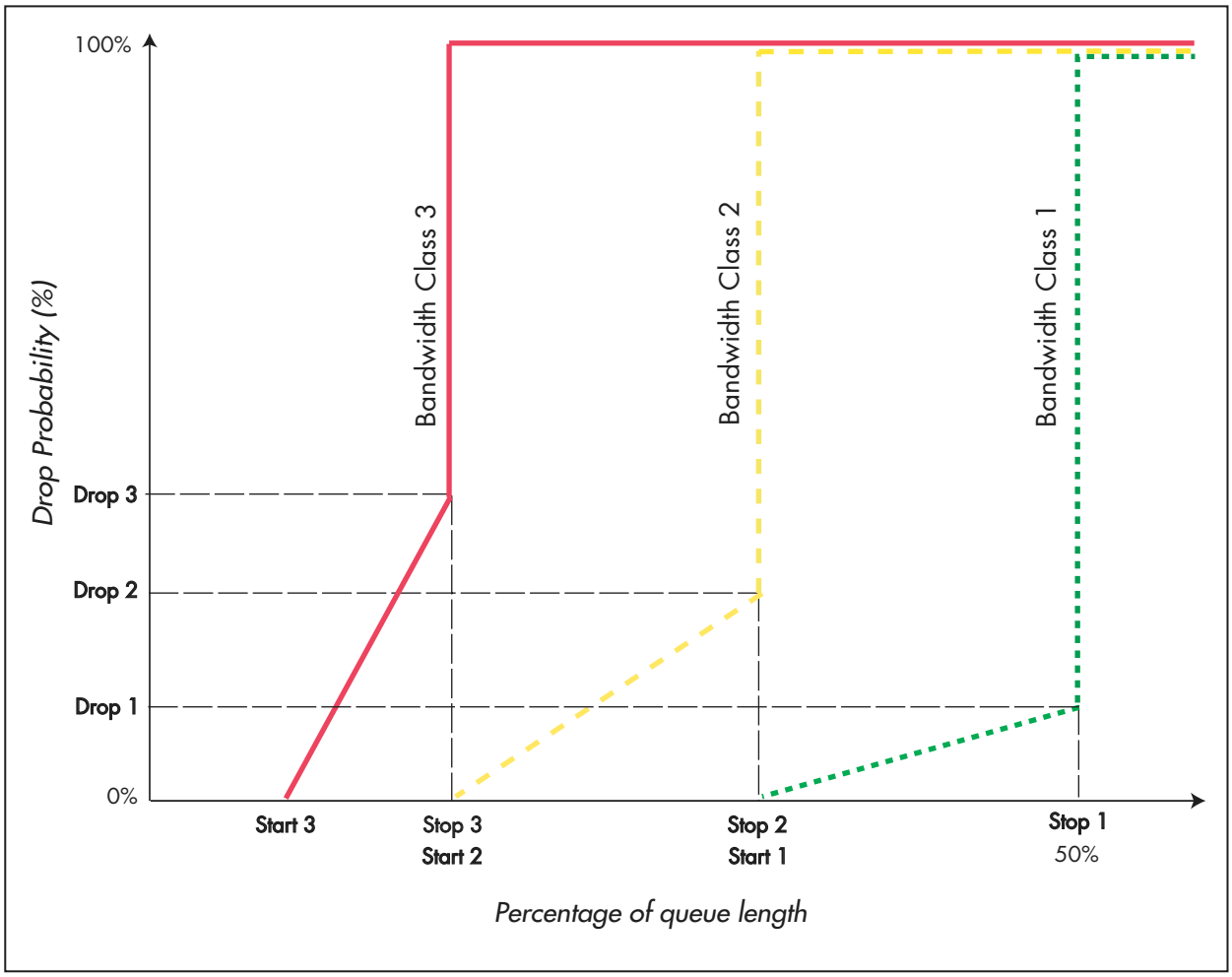
- **start***x*: the average percentage of queue length below which packets belonging to bandwidth class *x* are always accepted.
- **stop***x*: The average percentage of queue length above which all packets belonging to bandwidth class *x* are discarded.

  As the queue length increases, the random probability that a packet in that class will be dropped increases linearly until the queue length reaches the **stop***x* value.
- **drop***x*: The probability that a packet belonging to bandwidth class *x* will be dropped at the queue length determined by the **stop***x* value.

The RED algorithm calculates a running average queue length which the Start and Stop values use. Because they are averages, Start and Stop must be less than 100% of the maximum queue length.

Figure 3-13 on page 3-26 shows an example of a RED curve set. Note that packets from bandwidth class 3 begin to be dropped while the queue is still quite short (Start 3), whereas the queue is half full before packets from bandwidth class 1 begin to be dropped (Start 1), even in this aggressive RED curve set.

Figure 3-13: The default Aggressive RED curve set



Three RED curve sets exist by default. Table 3-6 shows their properties.

Table 3-6: The properties of the default RED curve sets

ID	Description	BW Class 1 (green)			BW Class 2 (yellow)			BW Class 3 (red)		
		Start	Stop	Drop prob.	Start	Stop	Drop prob.	Start	Stop	Drop prob.
0	Aggressive	35%	50%	20%	20%	35%	30%	10%	20%	40%
1	Medium	50%	70%	20%	30%	50%	30%	15%	30%	40%
2	Passive	80%	95%	20%	60%	80%	30%	40%	60%	40%

Dequeuing

The processes discussed up to this point are part of enqueuing; they determine which packets the switch puts into the egress queues for each traffic class. The next processes are applied during dequeuing, when the switch removes packets from the queues to send out.

The traffic class tree is the central structure that supports the dequeuing process. Each leaf traffic class has its own queue. This section gives an overview of the process of dequeuing, and the following sections discuss aspects of the dequeuing process in detail.

Summary of Dequeuing Flow of Events

In the dequeuing process, all the weights and/or priorities assigned to the traffic classes actually come into play.

The general flow of events in the dequeuing process is:

1. The physical interface (or egress tunnel) calls on the policy to give it a packet to send onto the wire.
2. The policy, in turn, calls on one of the traffic classes below it, to send up a packet (so that it can be passed onto the interface). It uses the WRR scheduling algorithm to determine whether the root, system, or default classes gets the current opportunity to send a packet.
3. The chosen class will, in turn, call upon one of its subclasses to send up a packet. Again, the choice of which subclass it calls on will depend on a priority or round-robin scheduling algorithm.
4. And so on ... right down to the leaf traffic classes.

Pull not push

This method by which packets make their way from the traffic classes to the interface is a “pull” mechanism. In this “pull” mechanism, the traffic class tree as a whole implements the scheduling algorithm, providing multiple choices as to which traffic class is called upon to pass a packet up to the next level of the tree.

Applying QoS Controls on Intermediate Classes

In fact, the dequeuing process involves more than just the pulling of packets up through the traffic class tree. Intermediate traffic classes, and the policy itself, can apply the following processes to packets on the way through:

- metering
- remarking
- virtual bandwidth shaping

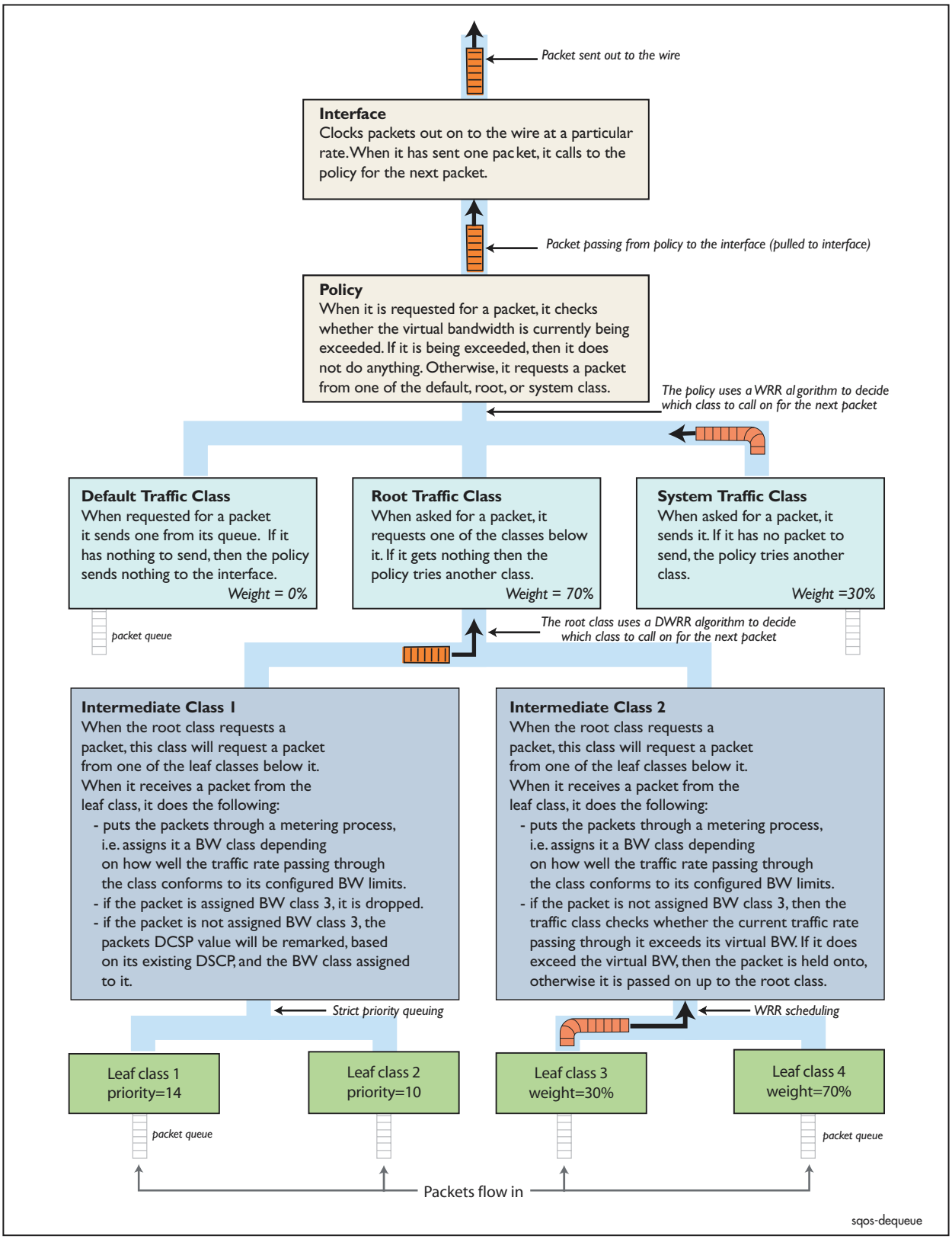
Any combination of those three processes can be defined on any of the intermediate traffic classes and on the policy.

To illustrate this, consider an example in which a policy has:

- A virtual bandwidth of Vp
- A default traffic class with weight = 0%
- A system traffic class with weight = 30%
- A root traffic class with weight = 70%
  The root class is using DWRR to schedule the drawing of packets from the classes beneath it.
- Two intermediate traffic classes below the root traffic class, one configured with metering and DSCP remarking, the other configured with metering and virtual bandwidth. Both intermediate traffic classes are configured to drop bandwidth class 3 packets.
- Two leaf classes on each intermediate class.

The dequeuing process for this example is shown in Figure 3-14 on page 3-29.

Figure 3-14: Example to illustrate the dequeuing process



Queue Scheduling

Queue scheduling refers to how the switch determines the order in which to empty queues, and what proportion of the bandwidth goes to packets from each queue.

In a multi-level traffic class tree, the policy pulls packets out of the leaf traffic class queues on a first-in-first-out (FIFO) basis into their intermediate traffic class queues, as described in "Dequeuing" on page 3-27. It then uses the scheduling method defined for the intermediate traffic class to dequeue the packets upwards in the tree. This continues until they reach the policy and are sent out.

You should put latency and jitter-sensitive traffic such as VoIP into a high priority traffic class, rather than a weighted class. You also need to plan your traffic class tree so that the VoIP traffic class is not a child of a weighted traffic class (except for the root class).

The available queue scheduling methods are:

- Priority Queuing (PQ)

  To select priority queuing, you assign a priority to each traffic class. Queues from higher priority traffic classes are emptied before any packets are transmitted from queues from lower priority traffic classes. This means, for example, that the queues for a traffic class with priority 7 must be totally empty before any packets from a traffic class with priority 6 or lower are sent. Note that when all your traffic classes use priorities, the policy will still use WRR or DWRR to assign a proportion of the bandwidth to system and default traffic.

  The risk with priority queuing is that high priority traffic may use all the available bandwidth, forcing the switch to drop all medium and low priority traffic.

- Weighted Round Robin (WRR)

  To select WRR, you assign a weighting to each traffic class and specify WRR in the policy or intermediate traffic class that the traffic classes belong to. The traffic classes share bandwidth on the basis of these weightings. Using this method, the switch can transmit packets from all traffic classes even under conditions of high congestion. You can configure the weightings to ensure that more packets per second are sent from traffic classes that are more sensitive to packet drop, than from less sensitive or lower priority traffic classes.

  The disadvantage of WRR is that flows with large packets get more than their fair share of the bandwidth.

- Deficit Weighted Round Robin (DWRR)

  To select DWRR, you assign a weighting to each traffic class and specify DWRR in the policy or intermediate traffic class that the traffic classes belong to.

  DWRR is similar to WRR, but much fairer across packets of different sizes. DWRR provides similar functionality to Class Based Weighted Fair Queuing (CBWFQ), with less performance impact.

- Mixed scheduling

  To select mixed scheduling on a policy or intermediate traffic class, you assign priorities to one or more of its sub traffic classes and weightings to the rest. The switch first services all the priority traffic classes, then the remaining bandwidth is divided among the weighted traffic classes. This option provides the functionality of low latency queuing.

Example of Mixed Scheduling

To demonstrate the effect of mixed scheduling, consider an interface with a speed limit of 2 Mbps, and 4 different flows, each arriving at up to 2 Mbps. The interface has a policy attached to it that contains 4 traffic classes, one for each flow:

- Traffic class 1, priority=15
- Traffic class 2, weight=50
- Traffic class 3, weight=30
- Traffic class 4, weight=20

The policy uses DWRR to schedule the 3 weighted traffic classes. Table 3-7 shows the egress rate ("out rate") for each traffic class, when traffic arrives at the traffic class at different rates ("in rate"). This table shows that the policy schedules all packets from the priority traffic class first, up to the 2 Mbps limit. If any bandwidth is left over, the policy schedules packets from the weighted traffic classes. The bandwidth is divided among the traffic classes that have packets to send, in proportion to their weights.

Table 3-7: Egress rates for different traffic classes using mixed scheduling

traffic class 1		**traffic class 2**		**traffic class 3**		**traffic class 4**	
in rate (Mbps)	**out rate (Mbps)**	**in rate (Mbps)**	**out rate (Mbps)**	**in rate (Mbps)**	**out rate (Mbps)**	**in rate (Mbps)**	**out rate (Mbps)**
2	2	2	0	2	0	2	0
1	1	2	0.5	2	0.3	2	0.2
1	1	0	0	2	0.6	2	0.4
0	0	0	0	0	0	2	2

Remarking

The switch remarks packets after it removes them from the traffic class queue, as part of the process of sending the packet out the egress interface. Unlike premarking, remarking occurs after metering, so the packet markers depend on its conformance to its bandwidth allocation. Remarking can change the packet:

- DSCP value. This enables you to mark the packet with information about its conformance for devices further downstream to use. For example, you may choose to mark bandwidth class 3 packets with a particular DSCP so a downstream device can give them a lower priority. There are two options for specifying the new DSCP:
  - Directly, by specifying a new DSCP for the traffic class

- Using the remarking table of a DSCP map (see Table 3-8). The switch reads the packet's DSCP and metered bandwidth class, and looks up the table to determine the new DSCP value for that combination of DSCP and bandwidth class.

■ bandwidth class. This enables hierarchical processing within a traffic class tree. To remark with a new bandwidth class, the switch uses the remarking table of a DSCP map. It reads the packet's current DSCP and metered bandwidth class and looks up the table to determine the new bandwidth class for that combination.

■ VLAN priority. This enables downstream switches to give a higher priority to packets, assuming those downstream switches are configured to do so.

Table 3-8: A conceptual diagram of part of a remarking table in a DSCP map

Current DSCP	Bandwidth Class 1		Bandwidth Class 2		Bandwidth Class 3	
	New BW class	New DSCP	New BW class	New DSCP	New BW class	New DSCP
0	newbwclass	newdscp	newbwclass	newdscp	newbwclass	newdscp
1	newbwclass	newdscp	newbwclass	newdscp	newbwclass	newdscp
. . .						
63	newbwclass	newdscp	newbwclass	newdscp	newbwclass	newdscp

Virtual Bandwidth

The virtual bandwidth for a policy or traffic class determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media. Setting a virtual bandwidth limit allows you to shape traffic by limiting the total bandwidth available to a policy or an intermediate traffic class.

Note that the rate you set needs to be lower than the required actual transmission rate, because the virtual transmission rate does not include the transmission of bits for the inter-frame gap, and some packet framing. For most packet types the difference is a few bytes. Virtual bandwidth may not be a useful tool for tunnelled packets, because packet padding may mean the difference is many bytes.

If you specify a virtual bandwidth for a policy or traffic class (intermediate or leaf), give the appropriate leaf traffic classes large maximum queue lengths. This enables them to buffer bursts of packets and avoids packet loss.

How to Configure a Software QoS Hierarchy

This section first describes how to build up your software QoS solution for an interface, out of classifiers, DAR objects, traffic classes and a policy. Then it describes how to change the default traffic class for QoS on unclassified traffic.

Later sections describe how to configure individual stages of the QoS solution.

The Total Software QoS Solution

This section gives a complete sequence of steps for configuring a QoS hierarchy, to show all commands available and their connections.

Before you configure

We strongly recommend that you plan the QoS scheme for your whole network on paper before applying it on the switch. In particular, we recommend you minimise the number of classifiers you configure, to maximise performance.

You will need the following information about your network:

- The different types of traffic flow that are currently performing below an acceptable standard, or that need bandwidth control
- Whether the switch is dropping an unacceptable number of packets at ingress
- The parameters you can use to uniquely identify each flow
- The interfaces through which the flows ingress and/or egress
- The tunnels through which the flows egress
- The level of service each flow requires (for example, bandwidth requirements, relative priorities)

Configuration rules

The following rules apply when building up a software QoS hierarchy

- An interface may only have one ingress policy and/or one egress policy.
- A tunnel may only have one software QoS policy.
- A policy may be assigned to many interfaces and tunnels.
- A policy may have many traffic classes.
- A traffic class may only be assigned to one policy or intermediate traffic class.
- An intermediate traffic class may have many sub traffic classes.
- A classifier may only be assigned to leaf traffic classes.
- A DAR object may be assigned to many leaf traffic classes. However, a DAR object should only be used once within a policy. A DAR object applies dynamic classifiers to traffic classes it is assigned to.
- A leaf traffic class may have many classifiers and/or DAR objects.
- A classifier may be assigned to many leaf traffic classes. However, a classifier may only be used once within a policy.

Configuration order

The steps for configuring a software QoS hierarchy are presented in Table 3-9, in a functional order. You will probably not require all the steps, and this order is not the only possible order. The order restrictions are:

- For elements like DSCP maps, meters and RED curves, you must create the element before you can use it in a traffic class or policy. If desired you can **create** the traffic class or policy first and then **set** it to use the appropriate element.
- For elements like classifiers, DAR objects, traffic classes and policies, you must create the element before you can add it to the element above it in the hierarchy.
- You must combine the elements into a hierarchy from the top down. Add traffic classes to the policy, then subclasses to the traffic classes if required, then classifiers and/or DAR objects to the traffic classes or subclasses.

Table 3-9: Overall procedure for configuring a software QoS hierarchy

Step	Command	Action
1	create classifier=0..9999 *options* **create sqos dar**=*id-list* [codec={audio|video|any}] [description=*description*] [dstip={*ipadd*[/0..32]|*ipv6add*[/0..128]}] [h323port=1..65535] [inactivetimeout={1..60|none}] [protocol={sip|rtsp|h323|all}] [rtspport=1..65535] [sipport=1..65535] [srcip={*ipadd*[/0..32]|*ipv6add*[/0..128]}]	Create static and/or dynamic classifiers to sort traffic into flows. For more information about static classifiers, see the Generic Packet Classifier chapter. For more information about dynamic classifiers, see "How to Configure DAR for Voice and Video Traffic" on page 3-48.
2	**create sqos meter**=*id-list* [description=*description*] [ignorebwclass3={yes|no}] [minbandwidth=*rate*[kbps|mbps|gbps]] [maxbandwidth=*rate*[kbps|mbps|gbps]] [minburstsize=*burstsize*[bytes|kbytes| mbytes|gbytes]] [maxburstsize=*burstsize*[bytes|kbytes| mbytes|gbytes]] [type={srtcm|trtcm}]	If required, create meters. Meters determine the bandwidth used by the packet and how conformant it is. For more information, see "Metering" on page 3-38.
3	**create sqos dscpmap**=*id-list* [description=*description*] **set sqos dscpmap**=*id-list* table=premark [description=*description*] [dscp=*dscp-list*] [newbwclass=1..3] [newdscp=0..63] **set sqos dscpmap**=*id-list* table=remark [description=*description*] [bwclass=*bwclass-list*] [dscp=*dscp-list*] [newbwclass=1..3] [newdscp=0..63]	If required, create DSCP maps and configure the tables in them. For each policy, the switch can use the tables in the policy DSCP map to set the DSCP bits and/or bandwidth class. The premarking table applies before metering, and the remarking table applies after. For more information, see "Premarking" on page 3-37 and "Remarking" on page 3-42.

Table 3-9: Overall procedure for configuring a software QoS hierarchy (Continued)

Step	Command	Action
4	**create sqos red**=*id-list* [description=*description*] [start1=0..100] [stop1=0..100] [drop1=0..100] [start2=0..100] [stop2=0..100] [drop2=0..100] [start3=0..100] [stop3=0..100] [drop3=0..100]	If required, create extra RED curves. RED curves allow early dropping of TCP packets to slow and smooth a congested TCP flow. For more information, see "RED" on page 3-41.
5	**create sqos policy**=*id-list* [bwclass3action={drop|pause|none}] [description=*description*] [dscpmap={0..9999|none}] [meter={0..9999|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [remarking={0..63|usedscpmap|none}] [remarkvlanpri={0..7|none}] [systemtraffic={5..50}] [virtbw={*bandwidth*[kbps|mbps|gbps]|none}] [weightscheduler={wrr|dwrr}]	Create a policy for each interface or tunnel, and specify QoS processing parameters if required. The policy determines the QoS processing for each flow on the interface. You can set QoS processing parameters on the policy or on the traffic classes attached to it. You can use the same policy on multiple interfaces or tunnels if they have sufficiently similar traffic flows.
6	**create sqos trafficclass**=*id-list* [bwclass3action={drop|pause|none}] [description=*description*] [maxqlen=1..1023] [meter={0..9999|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [premarkbwcl={1..3|usedscpmap}] [premarkdscp={0..63|usedscpmap|none}] [{priority=0..15|weight=0..100}] [qlimitexceedaction={none|log|trap|both}] [queuedrop={head|tail}] [red={0..9999|none}] [remarking=0..63|usedscpmap|none}] [remarkvlanpri={0..7|none}] [virtbw={*bandwidth*[kbps|mbps|gbps]|none}] [weightscheduler={wrr|dwrr}]	Create traffic classes. Traffic classes group similar traffic flows together and specify the QoS actions taken on them. Each traffic class contains an egress queue.
7	**add sqos interface dar** add sqos interface=*interface* dar=*id-list*	If configuring DAR, add DAR objects to interfaces.
8	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add traffic classes to policies.
9	**add sqos trafficclass subclass** add sqos trafficclass=0..9999 subclass=*id-list*	Add sub traffic classes to traffic classes if you need a multi-level traffic class tree. Continue until you have built up the traffic class tree for each policy.

Table 3-9: Overall procedure for configuring a software QoS hierarchy (Continued)

Step	Command	Action
10	**add sqos trafficclass classifier** add sqos trafficclass=0..9999 classifier=*id-list* or **add sqos trafficclass dar** add sqos trafficclass=0..9999 dar=*id-list*	Add static and/or dynamic classifiers to leaf traffic classes. You can use up to 16 classifiers per policy on AR725 and AR745 routers and 64 per policy on other routers and switches. Both static classifiers and the dynamic classifiers created by DAR objects count towards this quantity.
11	**set sqos interface**=*interface* [inpolicy={0..9999|none}] [outpolicy={0..9999|none}] [tunnelpolicy={0..9999|none}]	Apply policies to interfaces.
12	**enable sqos**	Enable software QoS.

Default traffic class

By default, the switch attaches a non-configurable Default Traffic Class (DTC) to each new policy, for unclassified traffic. The settings of this DTC are:

- queuedrop= tail
- weight=0
- maxqlen=64

If you require a DTC with different settings, follow the instructions in Table 3-10.

Table 3-10: Procedure for configuring a default traffic class (DTC)

Step	Command	Action
1	**create sqos trafficclass**=0..9999 weight=0..20 [*options*]	Create a new traffic class with the desired parameters.
2	**create sqos policy**=0..9999 defaulttrafficclass=0..9999 **set sqos policy**=0..9999 defaulttrafficclass=0..9999	Make the new traffic class the default traffic class for the policy.

How to Configure the Stages of a QoS Solution

This section describes how to configure each of the stages of a software QoS solution. Figure 3-10 on page 3-20 shows the stages.

Premarking

Premarking is performed at the traffic class level. Options are:

- Directly specifying a new bandwidth class and/or DSCP for all packets that belong to the traffic class. This is the approach you need for an AF domain. Use one of the commands:

```
create sqos trafficclass=id-list premarkbwcl=1..3
    premarkdscp=0..63 [other-options]

set sqos trafficclass=id-list premarkbwcl=1..3
    premarkdscp=0..63 [other-options]
```

- Using the policy's DSCP map, with the packet's current DSCP as an index into the table. Follow the instructions in Table 3-11.

The command order in the table is one of several possible orders. See "Configuration order" on page 3-34 for more information.

Premarking with a DSCP map

Table 3-11: Procedure for configuring premarking using a DSCP map

Step	Command	Action
1	**create sqos dscpmap**=*id-list* [description=*description*]	Create the DSCP map.
2	**set sqos dscpmap**=*id-list* table=premark [dscp=*dscp-list*] newbwclass=1..3 newdscp=0..63	Configure the map's premarking table. For each incoming **dscp** that you want to change, specify a **newdscp**. For each incoming **dscp** that you want to assign to a particular bandwidth class, specify the **newbwclass**.
3	**create sqos policy**=*id-list* dscpmap=0..9999 [*other-options*]	Create the policy and specify the DSCP map.
4	**create sqos trafficclass**=*id-list* [premarkbwcl=usedscpmap] [premarkdscp=usedscpmap] [*other-options*]	Create the traffic class and specify premarking.
5	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add the traffic class to the policy.
6	create classifier **create sqos dar** **create sqos meter** **create sqos red**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
7	**add sqos interface dar** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.

Table 3-11: Procedure for configuring premarking using a DSCP map (Continued)

Step	Command	Action
8	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
9	**show sqos dscpmap** **show sqos policy** **show sqos trafficclass**	Check the configuration.

Metering

Meters measure the bandwidth conformance of packets. You have a choice of meters:

- Single Rate Three Colour Marker: Follow the instructions in Table 3-12.
- Two Rate Three Colour Marker: Follow the instructions in Table 3-13.

The most common use of metering is to determine which packets are non-conformant and drop them. When you meter on traffic classes in which TCP flows are prevalent, and drop non-conformant packets, you need to choose burst sizes carefully. If burst sizes are too low, packets are dropped from flows that exceed their guaranteed bandwidth by small amounts. TCP flows react drastically to dropped packets—for example, they may halve the sending rate. This reduces the total TCP rate significantly and may stop the flows from getting their guaranteed bandwidths.

Burst sizes should be in proportion to the minimum and maximum rates, so if you increase the rates, also increase the burst sizes.

The command order in the tables is one of several possible orders. See "Configuration order" on page 3-34 for more information.

Single rate meter

Table 3-12: Procedure for creating and using a Single Rate Three Colour Marker meter

Step	Command	Action
1	**create sqos meter**=*id-list* [description=*description*] [ignorebwclass3={yes|no}] [maxbandwidth=*rate*[kbps|mbps|gbps]] [minburstsize=*burstsize*[bytes|kbytes| mbytes|gbytes]] [maxburstsize=*burstsize*[bytes|kbytes| mbytes|gbytes]]	Create the meter. The default meter type is single rate.
2	**create sqos policy**=*id-list* meter=0..9999 [bwclass3action={drop|pause|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [*other-options*] or **create sqos trafficclass**=*id-list* meter=0..9999 [bwclass3action={drop|pause|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [*other-options*]	Create the policy or traffic class, and specify the meter. You can also specify that the switch drop non-conformant packets or pause that flow. **Drop** discards the packet. **Pause** discards the packet and stops dequeuing packets from the flow for **pausetime** seconds. The switch can generate log messages and SNMP traps when it pauses a flow.
3	create classifier **create sqos dar** **create sqos red** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
4	**add sqos interface dar** **add sqos policy trafficclass** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
5	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
6	**show sqos meter** **show sqos policy** **show sqos trafficclass**	Check the configuration.

Two rate meter

Table 3-13: Procedure for creating and using a Two Rate Three Colour Marker meter

Step	Command	Action
1	**create sqos meter**=*id-list* type=trtcm [description=*description*] [ignorebwclass3={yes|no}] [maxbandwidth=*rate*[kbps|mbps|gbps]] [maxburstsize=*burstsize*[bytes|kbytes| mbytes|gbytes]] [minbandwidth=*rate*[kbps|mbps|gbps]] [minburstsize=*burstsize*[bytes|kbytes| mbytes|gbytes]]	Create the meter, specifying that it is a two rate meter.
2	**create sqos policy**=*id-list* meter=0..9999 [bwclass3action={drop|pause|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [*other-options*] **create sqos trafficclass**=*id-list* meter=0..9999 [bwclass3action={drop|pause|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [*other-options*]	Create the policy or traffic class, and specify the meter. You can also specify that the switch drop non-conformant packets or pause that flow. **Drop** discards the packet. **Pause** discards the packet and stops dequeuing packets from the flow for **pausetime** seconds. The switch can generate log messages and SNMP traps when it pauses a flow.
3	create classifier **create sqos dar** **create sqos red** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
4	**add sqos interface dar** **add sqos policy trafficclass** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
5	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
6	**show sqos meter** **show sqos policy** **show sqos trafficclass**	Check the configuration.

RED

RED curve sets allow gradual dropping of TCP packets as the traffic class queue becomes congested. Packets are dropped instead of being enqueued. You can:

- Use one of the default RED curve sets. Follow the instructions in Table 3-14.
- Create another RED curve set for your particular requirements. Follow the instructions in Table 3-15.

The command order in the tables is one of several possible orders. See "Configuration order" on page 3-34 for more information.

Using default RED curve sets

Table 3-14: Procedure for using one of the default RED curve sets

Step	Command	Action
1	**create sqos trafficclass**=*id-list* red=0..2 [maxqlen=1..1023] [queuedrop={head|tail}] [*other-options*]	Create the traffic class, and specify the RED curve set. Only use RED on leaf traffic classes. RED curve **0** is aggressive and starts dropping packets early. RED curve **1** is medium and starts dropping packets later. RED curve **2** is passive and only drops packets when the queue is almost full. You can also specify the queue length and whether to tail or head drop. The default is tail drop.
2	create classifier **create sqos dar** **create sqos policy** **create sqos meter** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
3	**add sqos interface dar** **add sqos policy trafficclass** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
4	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
5	**show sqos red** **show sqos trafficclass**	Check the configuration.

Creating new RED curve sets

Table 3-15: Procedure for creating and using a new RED curve set

Step	Command	Action
1	**create sqos red**=*id-list* [description=*description*] [start1=0..100] [stop1=0..100] [drop1=0..100] [start2=0..100] [stop2=0..100] [drop2=0..100] [start3=0..100] [stop3=0..100] [drop3=0..100]	Create the RED curve.
2	**create sqos trafficclass**=*id-list* red=3..9999 [maxqlen=1..1023] [queuedrop={head|tail}] [*other-options*]	Create the traffic class, and specify the RED curve set Only use RED on leaf traffic classes. You can also specify the queue length and whether to tail or head drop. The default is tail drop.
3	create classifier **create sqos dar** **create sqos policy** **create sqos meter** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
4	**add sqos interface dar** **add sqos policy trafficclass** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
5	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
6	**show sqos red** **show sqos trafficclass**	Check the configuration.

Remarking

Remarking is performed at the traffic class or policy level. You can replace any combination of DSCP, VLAN priority and bandwidth class. Options are:

- Directly specifying a new DSCP for packets belonging to a traffic class or policy, using one of the commands:

  ```
  create sqos trafficclass=id-list remarking=0..63
  create sqos policy=id-list remarking=0..63
  ```

- Using the policy's DSCP map, with the packet's metered bandwidth class and current DSCP as an index into the table. Follow the instructions in Table 3-16.
- Directly specifying a new VLAN priority for packets belonging to a traffic class or policy, using one of the commands:

  ```
  create sqos trafficclass=id-list remarkvlanpri=0..7
  create sqos policy=id-list remarkvlanpri=0..7
  ```

The command order in the table is one of several possible orders. See "Configuration order" on page 3-34 for more information.

Remarking with a DSCP map

Table 3-16: Procedure for configuring remarking using a DSCP map

Step	Command	Action
1	**create sqos dscpmap**=*id-list* [description=*description*]	Create the DSCP map.
2	**set sqos dscpmap**=*id-list* table=remark [bwclass=*bwclass-list*] [dscp=*dscp-list*] newbwclass=1..3 newdscp=0..63	Configure the map's remarking table. For each combination of **dscp** and bandwidth class (**bwclass**) that you want to change, specify a **newdscp** and/or **newbwclass**.
3	**create sqos meter**=*id-list* [description=*description*] [ignorebwclass3={yes|no}] [maxbandwidth=*rate*[kbps|mbps|gbps]] [maxburstsize=*burstsize*[bytes|kbytes|mbytes|gbytes]] [minbandwidth=*rate*[kbps|mbps|gbps]] [minburstsize=*burstsize*[bytes|kbytes|mbytes|gbytes]] [type={srtcm|trtcm}]	Create a meter to determine bandwidth conformance. For configuration details, see "Metering" on page 3-38.
4	**create sqos policy**=*id-list* dscpmap=0..9999 [meter=0..9999] [remarking=usedscpmap] [*other-options*]	Create the policy and specify the DSCP map. If you want traffic from all traffic classes in the policy to be remarked in the same way, specify remarking as part of the policy. If you want to use one meter for the whole policy, specify the meter as part of the policy.
5	**create sqos trafficclass**=*id-list* [remarking=usedscpmap] [meter=0..9999] [*other-options*]	Create the traffic class. If you want traffic from different traffic classes in the policy to be remarked differently, specify remarking as part of the traffic class. If you want to use different meters for different traffic classes in the policy, specify the meter as part of traffic class.
6	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add the traffic class to the policy.
7	create classifier **create sqos dar** **create sqos red**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
8	**add sqos interface dar** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.

Table 3-16: Procedure for configuring remarking using a DSCP map (Continued)

Step	Command	Action
9	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
10	**show sqos dscpmap** **show sqos policy** **show sqos trafficclass**	Check the configuration.

Queue scheduling

The queue scheduling mechanism determines the order in which the switch empties traffic class queues, and therefore which packets it sends out. Options are:

- Priority Queuing (PQ). Follow the instructions in Table 3-17.
- Weighted Round Robin (WRR) and Deficit Weighted Round Robin (DWRR). DWRR is provided instead of Class Based Weighted Fair Queuing (CBWFQ) because it provides similar functionality with less effect on performance. Follow the instructions in Table 3-18.
- Mixed scheduling, which is equivalent to Low Latency Queuing (LLQ). Mixed scheduling occurs when some traffic classes are priority classes and some are weighted. Follow the instructions in Table 3-19.

You should put latency and jitter-sensitive traffic such as VoIP into a high priority traffic class, rather than a weighted class. You also need to plan your traffic class tree so that the VoIP traffic class is not a child of a weighted traffic class (except for the root class).

The command order in the tables is one of several possible orders. See "Configuration order" on page 3-34 for more information.

Priority Queuing

Table 3-17: Procedure for configuring priority queuing

Step	Command	Action
1	**create sqos trafficclass**=*id-list* priority=0..15 [*other-options*]	Create the required traffic classes, giving each a priority. The higher the number, the higher the priority.
2	**create sqos policy**=0..9999 [systemtraffic={5..50}] [weightscheduler={wrr|dwrr}] [*other-options*]	Create the policy. If necessary, specify the proportion of bandwidth that the policy allows for system traffic. By default, the policy uses WRR to schedule the system, root and default traffic classes, which are all weighted classes. If required, change to DWRR.
3	**create sqos trafficclass**=0..9999 weight=0..100 [*other-options*] **set sqos policy**=0..9999 defaulttrafficclass=0..9999	If necessary, change the proportion of bandwidth that the policy allows for default traffic. First create a new traffic class with the desired weighting, then make the new traffic class the default traffic class for the policy.
4	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add the traffic class to the policy.
5	create classifier **create sqos dar** **create sqos meter** **create sqos red** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
6	**add sqos interface dar** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
7	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
8	**show sqos dscpmap** **show sqos policy** **show sqos trafficclass**	Check the configuration.

WRR and DWRR Queuing

Table 3-18: Procedure for configuring WRR or DWRR queue scheduling

Step	Command	Action
1	**create sqos trafficclass**=*id-list* weight=0..100 [*other-options*]	Create the required traffic classes, giving each a weight. The higher the number, the higher the proportion of bandwidth allocated to the traffic class. Weights do not have to total 100%. If they do not, normalised weights are used.
2	**create sqos trafficclass**=*id-list* [weightscheduler={wrr|dwrr}] [*other-options*]	If you are building a multi-level traffic class tree, create intermediate traffic classes to attach the weighted traffic classes to. By default, the intermediate traffic class uses WRR to schedule the weighted traffic classes. If required, change to DWRR.
3	**create sqos policy**=0..9999 [systemtraffic={5..50}] [weightscheduler={wrr|dwrr}] [*other-options*]	Create the policy. If necessary, specify the proportion of bandwidth that the policy allows for system traffic. By default, the intermediate traffic class uses WRR to schedule the weighted traffic classes, including the system and default traffic classes. If required, change to DWRR.
4	**create sqos trafficclass**=0..9999 weight=0..100 [*other-options*] **set sqos policy**=0..9999 defaulttrafficclass=0..9999	If necessary, change the proportion of bandwidth that the policy allows for default traffic. First create a new traffic class with the desired weighting, then make the new traffic class the default traffic class for the policy.
5	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add the traffic class to the policy.
6	create classifier **create sqos dar** **create sqos meter** **create sqos red** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.
7	**add sqos interface dar** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
8	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.

Table 3-18: Procedure for configuring WRR or DWRR queue scheduling (Continued)

Step	Command	Action
9	**show sqos dscpmap** **show sqos policy** **show sqos trafficclass**	Check the configuration.

Mixed Scheduling

Table 3-19: Procedure for configuring mixed scheduling

Step	Command	Action
1	**create sqos trafficclass**=*id-list* priority=0..15 [*other-options*]	Create the required priority traffic classes. The higher the priority number, the higher the priority.
1	**create sqos trafficclass**=*id-list* weight=0..100 [*other-options*]	Create the required weighted traffic classes. The higher the weight number, the higher the proportion of bandwidth allocated to the traffic class. Weights do not have to total 100%. If they do not, normalised weights are used.
2	**create sqos trafficclass**=*id-list* [weightscheduler={wrr|dwrr}] [*other-options*]	If you are building a multi-level traffic class tree, create intermediate traffic classes to attach the traffic classes to. By default, the intermediate traffic class uses WRR to schedule the weighted traffic classes. If required, change to DWRR.
3	**create sqos policy**=0..9999 [systemtraffic={5..50}] [weightscheduler={wrr|dwrr}] [*other-options*]	Create the policy. If necessary, specify the proportion of bandwidth that the policy allows for system traffic. By default, the policy uses WRR to schedule the weighted traffic classes that are attached to the root traffic class. If required, change to DWRR.
4	**create sqos trafficclass**=0..9999 weight=0..100 [*other-options*] **set sqos policy**=0..9999 defaulttrafficclass=0..9999	If necessary, change the proportion of bandwidth that the policy allows for default traffic. First create a new traffic class with the desired weighting, then make the new traffic class the default traffic class for the policy.
5	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add the traffic class to the policy.
6	create classifier **create sqos dar** **create sqos meter** **create sqos red** **create sqos dscpmap** **set sqos dscpmap**	Create the remaining QoS elements as required. For configuration details, see Table 3-9 on page 3-34.

Table 3-19: Procedure for configuring mixed scheduling (Continued)

Step	Command	Action
7	**add sqos interface dar** **add sqos trafficclass subclass** **add sqos trafficclass classifier** **add sqos trafficclass dar**	Build your QoS elements into a QoS hierarchy. For configuration details, see Table 3-9 on page 3-34.
8	**set sqos interface** **enable sqos**	Apply the policy to the required interface and enable software QoS. For configuration details, see Table 3-9 on page 3-34.
9	**show sqos dscpmap** **show sqos policy** **show sqos trafficclass**	Check the configuration.

How to Configure DAR for Voice and Video Traffic

Configuring DAR involves creating the required DAR objects, creating the rest of the software QoS policy, and applying the policy and DAR objects to the appropriate interfaces.

On a slow interface that carries voice (VoIP) traffic, you also need to force large (non-voice) packets to be fragmented, by setting a low interface MTU (maximum transmission unit), such as 256 bytes. This stops large data packets from delaying the voice packets, which are small. For example, a 1500 byte packet takes at least 190 milliseconds to send over a 64 kbps link. Acceptable total end-to-end latency for VoIP packets is only 150 ms.

On an interface that carries both VoIP and video traffic, it may or may not be desirable to configure a low MTU. This is because a low MTU will force the fragmentation of the video packets, and so cause significant overhead in the processing of the video stream. You may need to tune the MTU value to get a good balance between the latency in the VoIP and the CPU load induced by fragmenting the video packets.

You should put latency and jitter-sensitive traffic such as VoIP into a high priority traffic class, rather than a weighted class. You also need to plan your traffic class tree so that the VoIP traffic class is not a child of a weighted traffic class (except for the root class).

For an example, see Configuration Example "2: Guaranteeing VoIP Traffic using DAR" on page 3-67.

Table 3-20: Procedure for configuring Dynamic Application Recognition for VoIP and video traffic

Step	Commands	Action
1	**create sqos dar**=*id-list* [codec={audio|video|any}] [description=*description*] [dstip={*ipadd*[/0..32]|*ipv6add*[/0..128]}] [srcip={*ipadd*[/0..32]|*ipv6add*[/0..128]}] [inactivetimeout={1..3600|none}] [protocol={sip|rtsp|h323|all}] [h323port=1..65535] [rtspport=1..65535] [sipport=1..65535]	Create the DAR object. If necessary, limit it to matching packets with particular codec, protocol or IP settings.
2	**add sqos interface dar** add sqos interface=*interface* dar=*id-list*	Add the DAR object to the interface that voice or video session initiation messages are received on.
3	**create sqos policy**=*id-list* [bwclass3action={drop|pause|none}] [description=*description*] [dscpmap={0..9999|none}] [meter={0..9999|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [remarking={0..63|usedscpmap|none}] [remarkvlanpri={0..7|none}] [systemtraffic={5..50}] [virtbw={*bandwidth*[kbps|mbps|gbps]|none}] [weightscheduler={wrr|dwrr}]	Create a policy for the interface or tunnel through which voice or video traffic egresses, and specify QoS processing parameters if required.
4	**create sqos trafficclass**=*id-list* [bwclass3action={drop|pause|none}] [description=*description*][maxqlen=1..1023] [meter={0..9999|none}] [pauseaction={none|log|trap|both}] [pausetime={1..30}] [premarkbwcl={1..3|usedscpmap}] [premarkdscp={0..63|usedscpmap|none}] [{priority=0..15|weight=0..100}] [qlimitexceedaction={none|log|trap|both}] [queuedrop={head|tail}] [red={0..9999|none}] [remarking=0..63|usedscpmap|none}] [remarkvlanpri={0..7|none}] [virtbw={*bandwidth*[kbps|mbps|gbps]|none}] [weightscheduler={wrr|dwrr}]	Create at least one traffic class, and specify QoS processing parameters as required. Traffic classes group similar traffic flows together. Each traffic class contains an egress queue. VoIP traffic should go into a high-priority traffic class, not a weighted class.
5	**add sqos policy trafficclass** add sqos policy=0..9999 trafficclass=*id-list*	Add the traffic classes to the policy.
6	**add sqos trafficclass subclass** add sqos trafficclass=0..9999 subclass=*id-list*	If required, add sub traffic classes to the traffic classes. Continue until you have built up the traffic class tree for the policy.
7	**add sqos trafficclass dar** add sqos trafficclass=0..9999 dar=*id-list*	Add the DAR object to the appropriate leaf traffic class.

Table 3-20: Procedure for configuring Dynamic Application Recognition for VoIP and video traffic (Continued)

Step	Commands	Action
8	**set sqos interface**=*interface* outpolicy=0..9999 **set sqos interface**=*interface* tunnelpolicy=0..9999	Apply the policy to the interface or tunnel through which voice or video traffic egresses.
9	**enable sqos**	Enable software QoS.
10	**set interface mtu** set interface=*interface* mtu=256	If the interface is slow and carries voice traffic, which is sensitive to latency, force large packets to be fragmented.

How to Configure Software QoS on Particular Interfaces

This section describes how to configure a software QoS solution on PPP, PPPoE, and Frame Relay interfaces, and the switch instance. The biggest difference between the interfaces is the valid classifier options.

PPP and PPPoE

For PPP interfaces over:

- a synchronous port (SYN*n*)
- a DS3 port (DS3*n*)
- an ISDN call (ISDN-*callname*)
- an ACC call (ACC-*callname*)
- a MIOX circuit (MIOX*n-circuitname*)
- a TDM group (TDM-*groupname*)
- an L2TP call (TNL-*callname*)
- an ATM channel (for example, atm0.1)

configure Software QoS on the PPP interface. Follow the instructions in Table 3-21.

For PPPoE interfaces over Ethernet ports (ETH*n-servicename)*, configure software QoS on the Ethernet port. Follow the instructions in Table 3-22.

For PPPoE interfaces over VLANs on AR400 series and AR750S routers (VLAN*n-servicename*), configure software QoS on swi0. Follow the instructions in Table 3-24 on page 3-55. Use the **dvlan** parameter in the classifier to identify packets for each VLAN if required.

Table 3-21: Procedure for configuring software QoS on PPP interfaces

Step	Commands	Action
1		Configure the underlying physical interface as required. See the Interfaces chapter or the chapter for the physical interface that you are using.
2	create ppp=*ppp-interface* over=*physical-interface* [*other-options*]	Create and configure the PPP interface. See the Point-to-Point (PPP) chapter.
3	create classifier=1..9999 [iinterface=*interface*] [iport=*port*] [pppprotocolid={*ppp-protocol-id*|ip|ipv6|any}] [ipdaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipsaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipdscp={*dscp-list*|any}] [iptos={0..7|any}] [ipfrag={yes|no|any}] [ipoptions={yes|no|any}] [ipflowlabel={0..1048575|any}] [ipprotocol={tcp|udp|icmp|igmp|ospf|nontcpudp|any|*ip-protocol*}] [icmptype={any|echorply|unreachable|quench|redirect|echo|advertisement|solicitation|timeexceed|parameter|tstamp|tstamprply|inforeq|inforep|addrreq|addrrep|namereq|namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm|hostcomm|hostisolated|hostprec|hostredirect|hostrtos|hosttos|hostunknown|hostunreach|netcomm|netredirect|netrtos|nettos|netunknown|netunreach|noptr|portunreach|precedent|protunreach|ptrproblem|sourceroute|ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● ingress interface or port, for egress QoS only ● PPP protocol ID ● layer 3 ● layer 4
4		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
5	**set sqos interface**=*ppp-interface* inpolicy=0..9999 and/or **set sqos interface**=*ppp-interface* outpolicy=0..9999	Attach the policy to the PPP interface.
6	**enable sqos**	Enable software QoS.
7	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

Table 3-22: Procedure for configuring software QoS on PPP over Ethernet interfaces

Step	Commands	Action
1	create ppp=*ppp-interface* over=eth*n-servicename* [*other-options*]	Create and configure the PPP interface. See the Point-to-Point (PPP) chapter..
2	create classifier=1..9999 [pppindex=0..1023] [iinterface=*interface*] [iport=*port*] [svlan={*vlan-name*|1..4094|any}] [dvlan={*vlan-name*|1..4094|any}] [vlanpriority={*priority-list*|any}] [ethformat={802.2|ethii|netwareraw|snap|any}] [macdaddr={*macadd*|any}] [macsaddr={*macadd*|any}] [mactype={l2ucast|l2bmcast|any}] [protocol={*protocol-type*|arp|ip|ipv6|ipx|any}] [ipdaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipsaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipdscp={*dscp-list*|any}] [iptos={0..7|any}] [ipfrag={yes|no|any}] [ipoptions={yes|no|any}] [ipflowlabel={0..1048575|any}] [ipprotocol={tcp|udp|icmp|igmp|ospf| nontcpudp|any|*ip-protocol*} [icmptype={any|echorply|unreachable|quench| redirect|echo|advertisement|solicitation| timeexceed|parameter|tstamp|tstamprply| inforeq|inforep|addrreq|addrrep|namereq| namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm| hostcomm|hostisolated|hostprec|hostredirect| hostrtos|hosttos|hostunknown|hostunreach| netcomm|netredirect|netrtos|nettos| netunknown|netunreach|noptr|portunreach| precedent|protunreach|ptrproblem|sourceroute| ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● PPP index. This lets you separate different PPP interfaces over one Ethernet interface. ● ingress interface or port, for egress QoS only ● VLAN settings ● Ethernet settings ● layer 3 ● layer 4
3		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
4	**set sqos interface**=*eth-interface* inpolicy=0..9999 and/or **set sqos interface**=*eth-interface* outpolicy=0..9999	Attach the policy to the underlying Ethernet interface.
5	**enable sqos**	Enable software QoS.
6	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

Frame Relay

Software QoS treats each frame relay interface as a single ingress or egress interface with a single policy. If you need to control the quality of service given to traffic on individual DLCs, classify packets into different traffic classes according to DLCI and specify the appropriate QoS controls in each traffic class.

Table 3-23: Procedure for configuring software QoS on frame relay interfaces

Step	Commands	Action
1		If necessary, configure the underlying physical interface. See the Interfaces chapter or the chapter for the physical interface that you are using.
2	create framerelay=*fr-interface* over=*physical-interface* [*other-options*]	Create and configure the frame relay interface. See the Frame Relay (FR) chapter.
3	create classifier=1..9999 [iinterface=*interface*] [eport=*port*] [iport=*port*] [dlci={*dlci-range*|any}] [protocol={*protocol-type*|arp|ip|ipv6|ipx|any}] [ipdaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipsaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipdscp={*dscp-list*|any}] [iptos={0..7|any}] [ipfrag={yes|no|any}] [ipoptions={yes|no|any}] [ipflowlabel={0..1048575|any}] [ipprotocol={tcp|udp|icmp|igmp|ospf| nontcpudp|any|*ip-protocol*} [icmptype={any|echorply|unreachable|quench| redirect|echo|advertisement|solicitation| timeexceed|parameter|tstamp|tstamprply| inforeq|inforep|addrreq|addrrep|namereq| namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm| hostcomm|hostisolated|hostprec|hostredirect| hostrtos|hosttos|hostunknown|hostunreach| netcomm|netredirect|netrtos|nettos| netunknown|netunreach|noptr|portunreach| precedent|protunreach|ptrproblem|sourceroute| ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● interface and port ● DLCI ● layer 3 ● layer 4
4		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
5	**set sqos interface**=*ppp-interface* inpolicy=0..9999 and/or **set sqos interface**=*ppp-interface* outpolicy=0..9999	Attach the policy to the PPP interface.

Table 3-23: Procedure for configuring software QoS on frame relay interfaces (Continued)

Step	Commands	Action
6	**enable sqos**	Enable software QoS.
7	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

The Switch Instance

You can configure software QoS on the *switch instance*, swi0, which is the internal interface to the switch ports. This is helpful if:

- You use both Ethernet ports on an AR450S or AR750S router for high-speed WAN connections, because the switch instance may form a bottleneck (Figure 3-15).
- You need to control the quality of service given to traffic destined for particular VLANs or ports. In this case, you can classify packets into different traffic classes according to VLAN or port, specify the appropriate QoS controls in each traffic class, add the traffic classes to a policy, and apply the policy to swi0.
- You need to control the quality of service for a PPPoVLAN interface. You can classify packets according to their PPP index, which lets you control different PPP interfaces over the same VLAN.

Figure 3-15: Example of when you may require software QoS on the switch instance

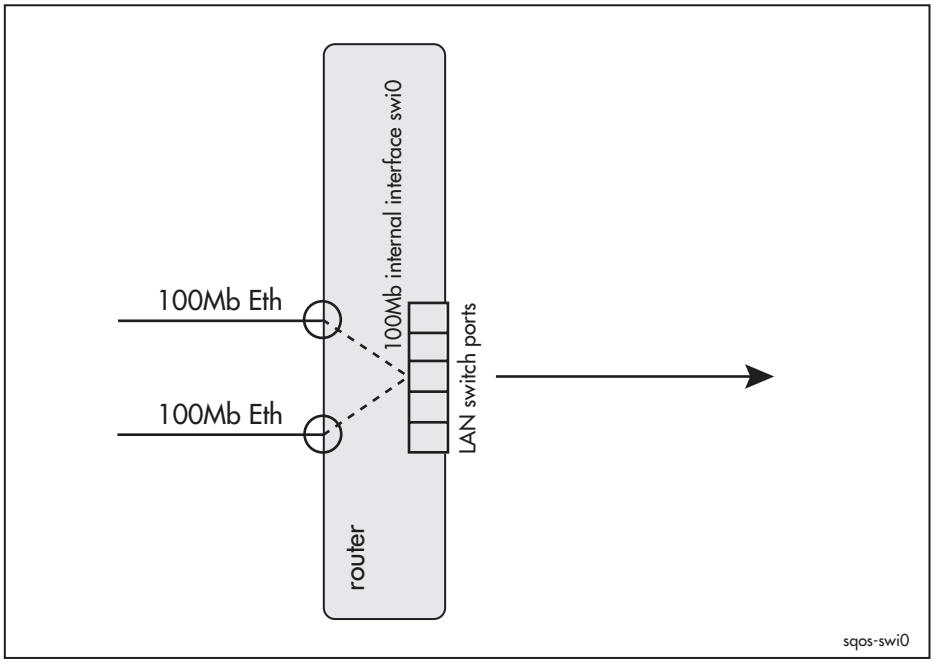


Table 3-24: Procedure for configuring software QoS on the switch instance

Step	Commands	Action
1	create classifier=1..9999 [iinterface=*interface*] [eport=*port*] [iport=*port*] [svlan={*vlan-name*|1..4094|any}] [dvlan={*vlan-name*|1..4094|any}] [vlanpriority={*priority-list*|any}] [ethformat={802.2|ethii|netwareraw|snap|any}] [macdaddr={*macadd*|any}] [macsaddr={*macadd*|any}] [mactype={l2ucast|l2bmcast|any}] [protocol={*protocol-type*|arp|ip|ipv6|ipx|any}] [pppindex=0..1023] [ipdaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipsaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipdscp={*dscp-list*|any}] [iptos={0..7|any}] [ipfrag={yes|no|any}] [ipoptions={yes|no|any}] [ipflowlabel={0..1048575|any}] [ipprotocol={tcp|udp|icmp|igmp|ospf| nontcpudp|any|*ip-protocol*} [icmptype={any|echorply|unreachable|quench| redirect|echo|advertisement|solicitation| timeexceed|parameter|tstamp|tstamprply| inforeq|inforep|addrreq|addrrep|namereq| namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm| hostcomm|hostisolated|hostprec|hostredirect| hostrtos|hosttos|hostunknown|hostunreach| netcomm|netredirect|netrtos|nettos| netunknown|netunreach|noptr|portunreach| precedent|protunreach|ptrproblem|sourceroute| ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● interface, port and VLAN. This lets you separate traffic destined for different VLANs or ports. ● Ethernet ● PPP index for PPPoVLAN traffic. This lets you separate different PPP interfaces on the same VLAN. ● layer 3 ● layer 4
2		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
3	**set sqos interface**=swi0 outpolicy=0..9999	Attach the policy to swi0.
4	**enable sqos**	Enable software QoS.
5	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

How to Configure Software QoS on Tunnels

QoS functionality is performed on packets before they are encapsulated and enter the tunnel.

VPN

VPN tunnels provide a secure connection across a WAN link.

Table 3-25: Procedure for configuring software QoS on VPN tunnels

Step	Commands	Action
1		Create the VPN tunnel. See the IP Security (IPsec) chapter for commands and examples.
2	create classifier=1..9999 [iinterface=*interface*] [iport=*port*] [ipdaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipsaddr={*ipadd*[/0..32]|*ipv6add*[/0..128]|any}] [ipdscp={*dscp-list*|any}] [iptos={0..7|any}] [ipfrag={yes|no|any}] [ipoptions={yes|no|any}] [ipflowlabel={0..1048575|any}] [ipprotocol={tcp|udp|icmp|igmp|ospf| nontcpudp|any|*ip-protocol*} [icmptype={any|echorply|unreachable|quench| redirect|echo|advertisement|solicitation| timeexceed|parameter|tstamp|tstamprply| inforeq|inforep|addrreq|addrrep|namereq| namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm| hostcomm|hostisolated|hostprec|hostredirect| hostrtos|hosttos|hostunknown|hostunreach| netcomm|netredirect|netrtos|nettos| netunknown|netunreach|noptr|portunreach| precedent|protunreach|ptrproblem|sourceroute| ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● ingress interface and port ● IP and IPv6 layer 3 ● layer 4
3		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
4	**set sqos interface**=ipsec-*policyname* tunnel|policy=0..9999 where *policyname* is the name of the IPsec policy	Attach the policy to the VPN tunnel.
5	**enable sqos**	Enable software QoS.
6	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

6 to 4

IPv6 to IPv4 tunnels take IPv6 traffic from your LAN and send it out over an IPv4 WAN link.

Table 3-26: Procedure for configuring software QoS on 6-to-4 tunnels

Step	Commands	Action
1		Create the VPN tunnel. See the Internet Protocol Version 6 (IPv6) chapter for commands and an example.
2	create classifier=1..9999 [iinterface=*interface*] [iport=*port*] [ipdaddr={*ipv6add*[/0..128]|any}] [ipsaddr={*ipv6add*[/0..128]|any}] [ipdscp={*dscp-list*|any}] [ipflowlabel={0..1048575|any}] [ipprotocol={tcp|udp|icmp|any|*ip-protocol*} [icmptype={any|echorply|unreachable|quench| redirect|echo|advertisement|solicitation| timeexceed|parameter|tstamp|tstamprply| inforeq|inforep|addrreq|addrrep|namereq| namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm| hostcomm|hostisolated|hostprec|hostredirect| hostrtos|hosttos|hostunknown|hostunreach| netcomm|netredirect|netrtos|nettos| netunknown|netunreach|noptr|portunreach| precedent|protunreach|ptrproblem|sourceroute| ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● ingress interface and port ● IPv6 layer 3 ● IPv6 layer 4
3		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
4	**set sqos interface**=*virtn* tunnel|policy=0..9999 where *virtx* is the name of the tunnel (e.g. virt0)	Attach the policy to the 6-to-4 tunnel.
5	**enable sqos**	Enable software QoS.
6	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

Generic Router Encapsulation (GRE)

Table 3-27: Procedure for configuring software QoS on GRE tunnels

Step	Commands	Action
1		Create the GRE tunnel. See the Generic Routing Encapsulation (GRE) chapter for commands and examples.
2	create classifier=1..9999 [iinterface=*interface*] [iport=*port*] [ipdaddr={*ipadd*[/0..32]|any}] [ipsaddr={*ipadd*[/0..32]|any}] [ipdscp={*dscp-list*|any}] [iptos={0..7|any}] [ipfrag={yes|no|any}] [ipoptions={yes|no|any}] [ipprotocol={tcp|udp|icmp|igmp|ospf| nontcpudp|any|*ip-protocol*} [icmptype={any|echorply|unreachable|quench| redirect|echo|advertisement|solicitation| timeexceed|parameter|tstamp|tstamprply| inforeq|inforep|addrreq|addrrep|namereq| namerply|*icmp-type*}] [icmpcode={any|filter|fragment|fragreassm| hostcomm|hostisolated|hostprec|hostredirect| hostrtos|hosttos|hostunknown|hostunreach| netcomm|netredirect|netrtos|nettos| netunknown|netunreach|noptr|portunreach| precedent|protunreach|ptrproblem|sourceroute| ttl|*icmp-code*}] [tcpflags={{urg|ack|rst|syn|fin}[,...]|any}] [tcpdport={*port-range*|any}] [tcpsport={*port-range*|any}] [udpdport={*port-range*|any}] [udpsport={*port-range*|any}]	Create classifiers. Valid parameters include: ● ingress interface and port ● IP layer 3 ● layer 4
3		Create the QoS policy and its underlying hierarchy. See Table 3-9 on page 3-34 for details.
4	**set sqos interface**=*gren* tunnel|policy=0..9999 where *gren* is the name of the tunnel (e.g. gre1)	Attach the policy to the GRE tunnel.
5	**enable sqos**	Enable software QoS.
6	**set interface mtu** set interface=*interface* mtu=256	If the interface carries traffic that is sensitive to latency, such as voice traffic, force large packets to be fragmented.

Interaction with Other Modules

This section describes the effect of software QoS on some other software features. Some of these are alternatives to software QoS, and some interact with it.

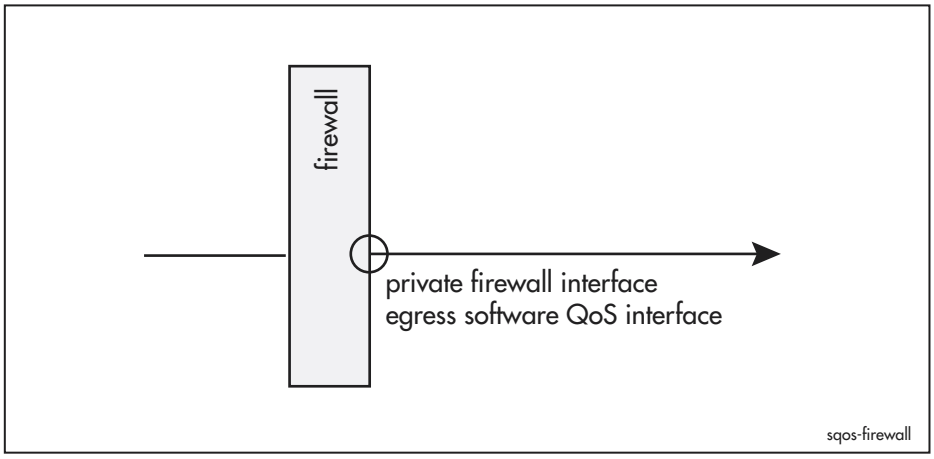
Network Address Translation (NAT)

Network Address Translation (NAT) is available through the firewall or the IP NAT feature. It translates between private IP settings on the LAN and public IP settings on the WAN. Therefore it can change a packet's source or destination IP address, TCP port or UDP port, depending on the NAT settings and direction of traffic flow. In general, NAT gives the same IP settings to all translated packets that leave your LAN through a given interface.

Software QoS may use these IP settings to classify packets. By default, it uses the pre-NAT settings for classification because these contain the distinguishing information. You need to use the post-NAT settings instead if **all** the following points apply (Figure 3-16):

- you are configuring software QoS on traffic that comes from a WAN link to your LAN
- you are applying QoS at the egress (LAN) interface
- you are classifying on destination IP address, destination TCP port or destination UDP port
- the interface is a bottleneck.

Figure 3-16: Firewall interface which requires post-NAT IP settings



To use post-NAT IP settings, use one of the commands:

```
create sqos policy=id-list ignoreprenatinfo=yes
   [other-options]

set sqos policy=id-list ignoreprenatinfo=yes [other-options]
```

Resource Reservation Protocol (RSVP)

RSVP and software QoS cannot effectively be used together. RSVP assigns packets to a reserved queue, which software QoS treats as system traffic. Software QoS policies use WRR to schedule system traffic, and limit it to 20% of the available bandwidth by default, and a maximum of 50%.

Priority Filters

Priority filters and software QoS cannot be used together. Priority filters provide a limited subset of QoS functionality. Software QoS offers much greater control over service quality.

Policy Filters

Policy filters cannot be used in a DiffServ domain. Policy filters use the TOS precedence field of the IP packet to determine routing. DiffServ also uses this field.

Bandwidth Limiting on Ethernet Interfaces

When software QoS is enabled on an Ethernet interface such as eth0, the command **set eth maxbandwidth** has no effect. To use software QoS to limit the bandwidth, configure a virtual bandwidth limit on the appropriate policy or traffic class.

Counters

You can view an extensive set of counters relating to the operation of software QoS:

- for **classifier** counters, use the command:

```
show sqos counters classifier[=id-list]
    [direction={in|out|tunnel|all}] [interface=interface]
```

- for **DAR** counters, use the command:

```
show sqos counters dar[=id-list]
    [direction={in|out|tunnel}] [interface=interface]
```

- for **traffic class** counters, use the command:

```
show sqos counters trafficclass[={id-list|default|system}]
    [direction={in|out|tunnel|all}] [interface=interface]
```

- for **policy** counters, use the command:

```
show sqos counters policy[=id-list]
    [direction={in|out|tunnel}] [interface=interface]
```

For each category of counter, you can view counters relating to:

- a particular entity, for example a particular classifier or a particular traffic class, by specifying its ID
- actions in ingress QoS, egress QoS, or QoS on tunnel interfaces, by specifying the **direction** parameter

- QoS on a particular interface, by specifying the **interface** parameter

The counters provide the necessary information to enable you to determine if your QoS configuration is working as intended. For example, you can work out which traffic classes are the busiest, and where most of the packets are being dropped. If the traffic distribution across classes is not what you originally intended, you can re-arrange the properties of the classes and re-monitor until your configuration operates as desired.

If the Software QoS policy does not appear to be processing packets in the way you expect, the counters can help you track down what is wrong. For example, if a classifier has been misconfigured so it is actually getting no hits, the classifier counters will show this.

Some of the sorts of statistics that you can collect from these counters are:

- How many hits there have been on a given classifier
- How many dynamic classifiers have been created by a DAR
- A list of the currently active dynamic classifiers for a given DAR
- Total packets processed by a traffic class or policy
- Current and average queue lengths for a traffic class
- Average latency for packets passing through a given traffic class
- Number of packets classified green, yellow, and red by a given traffic class
- Number of packets dropped by the RED curves on a given traffic class

The counters also provide clear evidence of whether Service Level Agreements are adhered to, by showing whether the traffic levels for any part of a contracted service are meeting the contracted requirements.

Debugging

The are eight software QoS debug options:

Option	Meaning
all	All debugging modes.
dar	Notifications when DAR objects and instances are created or destroyed.
dardata	More detailed information about SIP and RTSP data.
engine	Debugging information related to the packet conditioning engine.
error	Critical error debugging information, including a stack trace.
info	General command debugging information.
mark	Packet marking debugging information.
pkt	Packet debugging.

Some of these debug modes may help you check that software QoS is functioning as expected. For example, the **info** option gives additional information when the switch carries out commands. The **mark** option shows that packets are being marked with different priority values. The **dar** option shows that dynamic classifiers have been created for appropriate voice traffic.

To enable debugging, use the command:

```
enable sqos
   debug={all|dar|dardata|engine|error|info|mark|pkt}
```

The following figures show examples of the output from some of these debugging modes.

Figure 3-17: Example output from the command **enable sqos debug=dar**

```
Manager >
SQOS DAR: Classifer=10000 tc=1 ip=192.168.2.1/32 port=45678-45679 created
Manager >
SQOS DAR: Classifer=10001 tc=1 ip=192.168.1.1/32 port=38168-38169 created
Manager >
SQOS DAR: Classifer=10001 tc=1 ip=192.168.1.1/32 port=38168-38169 destroyed
SQOS DAR: Classifer=10000 tc=1 ip=192.168.2.1/32 port=45678-45679 destroyed
```

Figure 3-18: Example output from the command **enable sqos debug=dardata**

```
Manager >
SQOS DARDATA: INVITE sip:192.168.1.1:5060 SIP/2.0
SQOS DARDATA: Via: SIP/2.0/UDP 192.168.2.3:5060
SQOS DARDATA: Max-Forwards: 70
SQOS DARDATA: From: "user1"
<sip:user123456@mycompa.com>;tag=3b2ec08b66ee47b496f3a49a58aa7fa2
SQOS DARDATA: To: <sip:192.168.1.1>
SQOS DARDATA: Call-ID: 470f605893de4c4e9bc9e95adedee8b1@192.168.0.1=CALL-ID
SQOS DARDATA: CSeq: 1 INVITE
SQOS DARDATA: Contact: <sip:192.168.2.3:5060>
SQOS DARDATA: User-Agent: RTC/1.2
SQOS DARDATA: Content-Type: application/sdp
SQOS DARDATA: CONTENT-LENGTH: 284=284
SQOS DARDATA: <END>
SQOS DARDATA: v=0
SQOS DARDATA: o=- 0 0 IN IP4 192.168.2.1
SQOS DARDATA: s=session
SQOS DARDATA: c=IN IP4 192.168.2.1
SQOS DARDATA: b=CT:110
SQOS DARDATA: t=0 0
SQOS DARDATA: m=audio 45678 RTP/AVP 97 0 8 4 101
SQOS DARDATA: a=rtpmap:97 red/8000
SQOS DARDATA: a=rtpmap:0 PCMU/8000
SQOS DARDATA: a=rtpmap:8 PCMA/8000
SQOS DARDATA: a=rtpmap:4 G723/8000
SQOS DARDATA: a=rtpmap:101 telephone-event/8000
SQOS DARDATA: a=fmtp:101 0-16
SQOS DARDATA: a=encryption:rejected
```

Figure 3-19: Example output from the command **enable sqos debug=pkt**

```
Manager >
SQOS PKT: POLI=1 (OUT) TC=sys qPkts=1 qBytes=42 Enqueued
SQOS PKT: POLI=1 (OUT) TC=sys qPkts=0 qBytes=0 Dequeued
SQOS PKT: POLI=1 (OUT) TC=1 qPkts=1 qBytes=1512 Enqueued
SQOS PKT: POLI=1 (OUT) TC=1 qPkts=0 qBytes=0 Dequeued
SQOS PKT: POLI=1 (OUT) TC=1 qPkts=1 qBytes=1512 Enqueued
SQOS PKT: POLI=1 (OUT) TC=1 qPkts=0 qBytes=0 Dequeued
Manager >
SQOS PKT: POLI=1 (OUT) TC=1 qPkts=1 qBytes=1512 Enqueued
SQOS PKT: POLI=1 (OUT) TC=1 qPkts=0 qBytes=0 Dequeued
```

Figure 3-20: Example output from the command **enable sqos debug=mark**

```
Manager >
SQOS MARK: Pkt 04b7880c, Old DSCP 0
SQOS MARK: IPv4 Pkt 04b7880c, New DSCP 10
SQOS MARK: Packet dump after marking
    0050fc31 d7ad0000 cd08106f 08004528 05dab22d 40003f11 27d9ac70 0101ac72
    0201041b 138905c6 07e80000 000041a6 57740009 939c0000 00000000 00010000
    13890000 00000000 fa00ffff ff9c3637 38393031 32333435 36373839 30313233
    34353637
```

Figure 3-21: Example output from the command **enable sqos debug=engine**

```
SQOS ENGINE: callback eth0 2
SQOS ENGINE: callback eth0 2
SQOS ENGINE: callback eth0 2
SQOS ENGINE: callback eth0 2
SQOS ENGINE: callback eth0 2
SQOS ENGINE: callback eth0 2
SQOS ENGINE: callback eth0 2
```

Figure 3-22: Example output from the command **enable sqos debug=info**

```
Manager > set sqos int=eth0 outpolicy=1

SQOS INFO: SQOS Active on eth0 (Egress)
Info (1123003): Operation successful.
```

Network Configuration Examples

This series of examples covers common network situations, with increasing complexity. The first example begins with a simple desire to make VoIP calls at the same time as non-critical file server downloads over a 128 kbps PPP link. The next 4 examples build on this, showing how to use DAR, how to improve the quality of the file server downloads, how to secure the link with VPN, and how to deal with critical file server traffic.

The last example shows multiple applications running over a frame relay link, including voice, video conferencing, network monitoring, and server traffic.

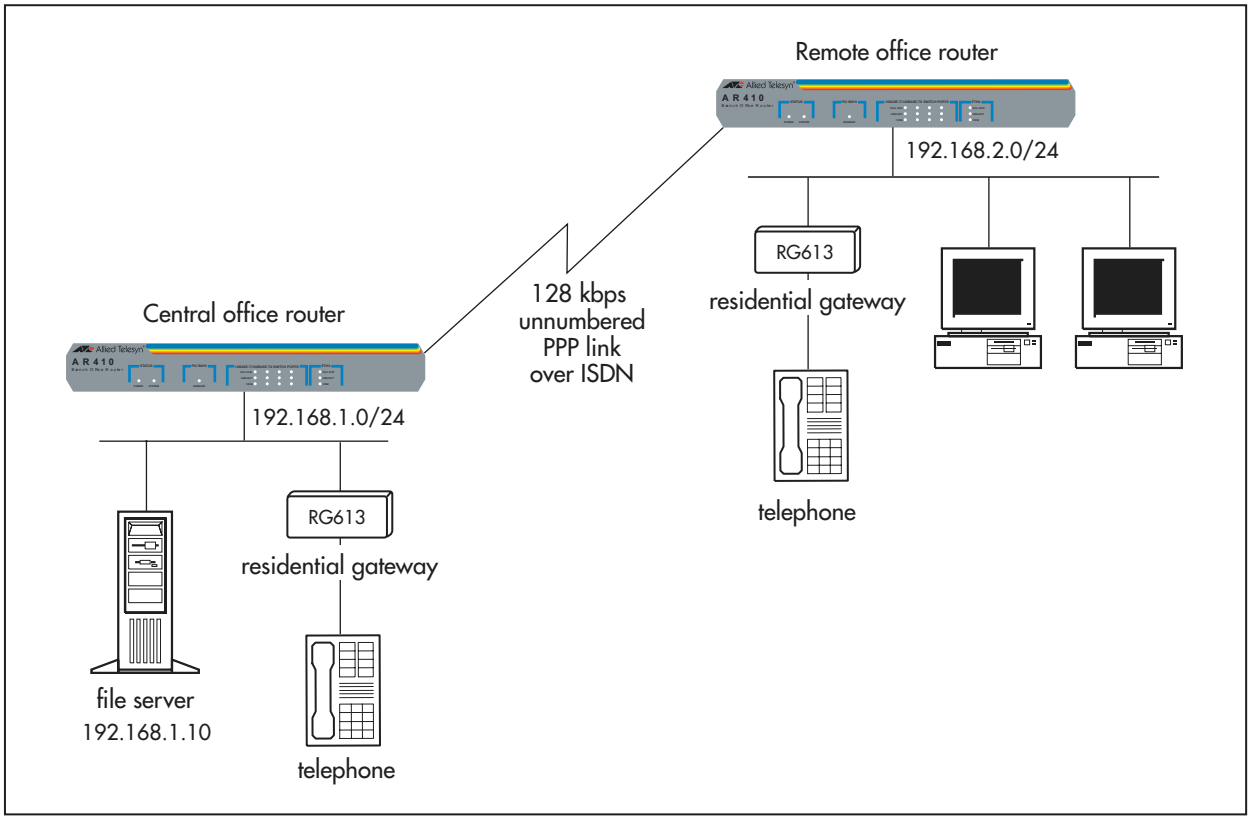
Some interface types, port types, or command options in these examples may not be supported on your switch. Interfaces and port types vary depending on the switch model, and whether an expansion unit (PIC, NSM) is installed. For more information, see the Hardware Reference.

1: Guaranteeing VoIP Traffic

In this scenario (Figure 3-23):

- VoIP traffic has the highest priority possible and a short queue, so the switchess always transmit VoIP traffic with low drop rate, delay, and jitter.
- the switches identify VoIP traffic by the UDP ports it uses.
- The switches drop any other traffic they need to, so they can send the VoIP traffic at high quality.

Figure 3-23: Configuration for guaranteeing VoIP traffic



```
# Guaranteeing VoIP traffic on 128kbps PPP link over ISDN
# Central office configuration

set system name=central
set system territory=europe

# Configure ISDN and PPP
add user=remote password=rempass login=no telnet=no
add isdn call=office number=0 prec=in dir=in searchsub=local
create ppp=0 over=isdn-office authentication=chap echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.1.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.2.0 int=ppp0 next=0.0.0.0

# Create a classifier for VoIP (This example assumes voice traffic
# uses UDP ports between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a QoS policy.
create sqos policy=1

# Create a traffic class for VoIP traffic and give it the highest
# priority and a short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give
# it the second highest priority
create sqos trafficclass=2 priority=14

# Add the traffic classes to the policy.
add sqos policy=1 trafficclass=1,2

# Add the classifiers to the traffic classes
add sqos trafficclass=1 class=1
add sqos trafficclass=2 class=2

# Use the policy on ppp0
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

```
# Guaranteeing VoIP traffic on 128kbps PPP link over ISDN
# Remote office configuration

set system name=remote
set system territory=europe

# Configure ISDN and PPP
add isdn call=office number=your-central-office-number prec=out outsub=local
create ppp=0 over=isdn-office username=remote passw=rempass echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.2.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.1.0 int=ppp0 next=0.0.0.0

# Create a classifier for VoIP (This example assumes Voice traffic
# is using UDP ports between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create  class=2 udpdport=5060

# Create a QoS policy.
create sqos policy=1

# Create a traffic class for VoIP traffic and give it the highest
# priority and a short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give
# it the second highest priority
create sqos trafficclass=2 priority=14

# Add the traffic classes to the policy.
add sqos policy=1 trafficclass=1,2

# Add the classifiers to the traffic classes
add sqos trafficclass=1 class=1
add sqos trafficclass=2 class=2

# Use the policy on ppp0
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

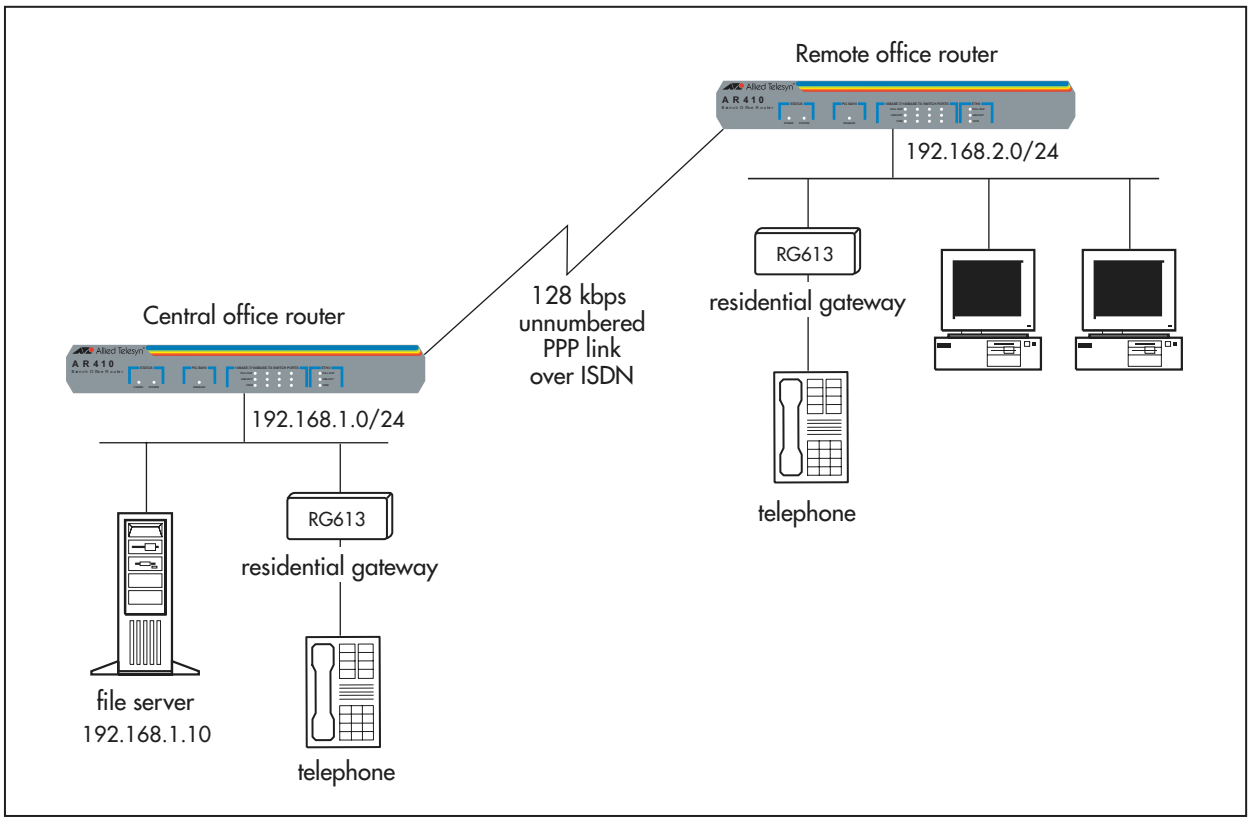
2: Guaranteeing VoIP Traffic using DAR

This scenario is an alternative to scenario 1, and uses DAR to identify the voice traffic. The network set-up is identical to scenario 1.

In this scenario (Figure 3-24):

- VoIP traffic has the highest priority possible and a short queue, so the switches always transmit VoIP traffic with low drop rate, delay, and jitter.
- The switches drop any other traffic they need to, so they can send the VoIP traffic at high quality.

Figure 3-24: Configuration for guaranteeing VoIP traffic using DAR



```
# Guaranteeing VoIP traffic using DAR on 128kbps PPP link over ISDN
# Central office configuration

set system name=central
set system territory=europe

# Configure ISDN and PPP
add user=remote password=rempass login=no telnet=no
add isdn call=office number=0 prec=in dir=in searchsub=local
create ppp=0 over=isdn-office authentication=chap echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.1.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.2.0 int=ppp0 next=0.0.0.0

# Create a DAR object for VoIP, to match sessions initiated by SIP signalling
create sqos dar=1 prot=sip

# The DAR does not match the signaling traffic itself, so create a separate
# classifier to match on the SIP signaling traffic
create  class=1 udpdport=5060

# Put the DAR onto the PPP interface so it will recognise when a phone call is set
# up over the PPP interface
add sqos interface=ppp0 dar=1

# Create a QoS policy
create sqos policy=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1,2

# Add the DAR object and SIP classifier to the traffic classes
add sqos trafficclass=1 dar=1
add sqos trafficclass=2 class=1

# Use the policy on ppp0
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

```
# Guaranteeing VoIP traffic using DAR on 128kbps PPP link over ISDN
# Remote office configuration

set system name=remote
set system territory=europe

# Configure ISDN and PPP
add isdn call=office number=your-central-office-number prec=out outsub=local
create ppp=0 over=isdn-office username=remote passw=rempass echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.2.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.1.0 int=ppp0 next=0.0.0.0

# Create a DAR object for VoIP, to match sessions initiated by SIP signalling
create sqos dar=1 prot=sip

# The DAR does not match the signaling traffic itself, so create a separate
# classifier to match on the SIP signaling traffic
create class=1 udpdport=5060

# Put the DAR onto the PPP interface so it will recognise when a phone call is set
# up over the PPP interface
add sqos interface=ppp0 dar=1

# Create a QoS policy
create sqos policy=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1,2

# Add the DAR object and SIP classifier to the traffic classes
add sqos trafficclass=1 dar=1
add sqos trafficclass=2 class=1

# Use the policy on ppp0
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

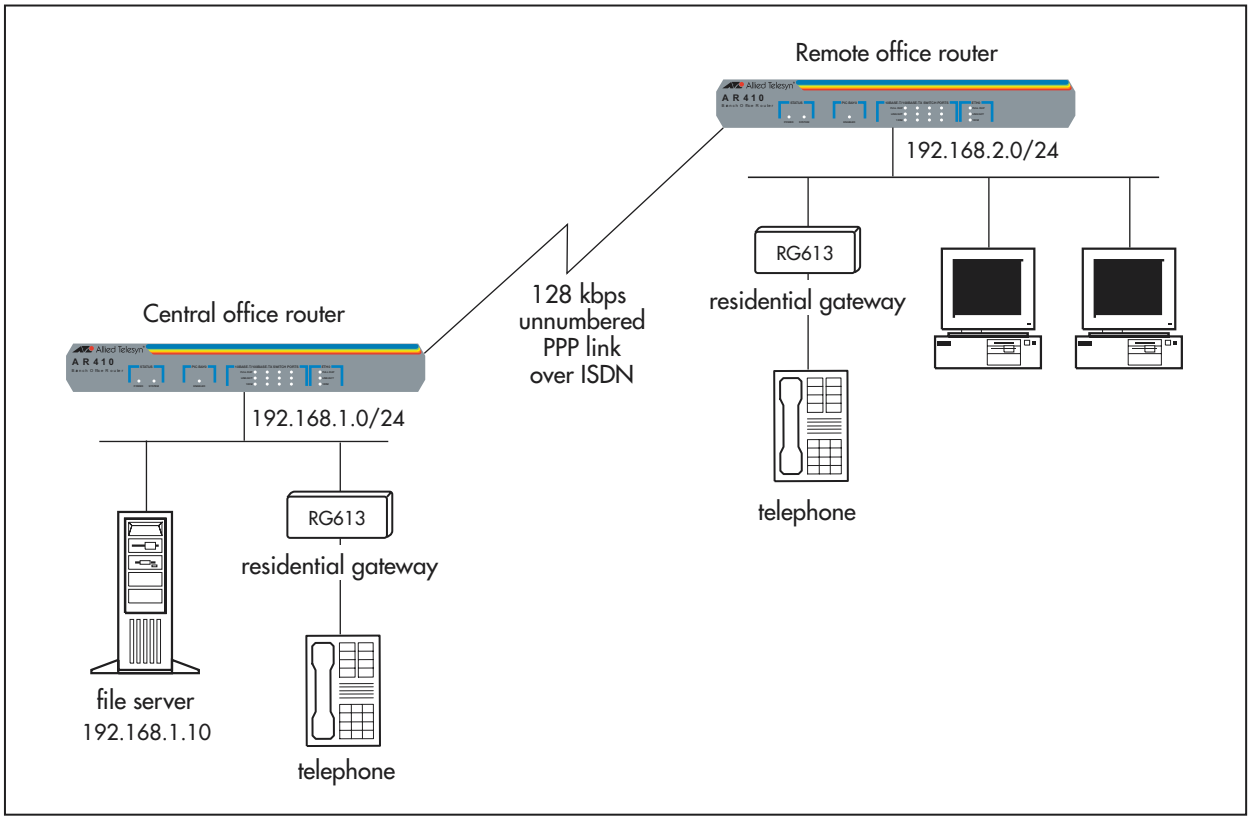
3: Guaranteeing VoIP Traffic While Maintaining File Server Traffic

This scenario expands on scenario 1 by improving the quality of service for file server traffic. The network set-up is identical to scenario 1.

In this scenario (Figure 3-25):

- VoIP traffic has the highest priority possible and a short queue, so the switches always transmit VoIP traffic with low drop rate, delay, and jitter.
- the switches identify VoIP traffic by the UDP ports it uses.
- File server traffic has the next highest priority. The switches use a medium RED curve set to drop file server traffic as necessary and control the TCP flows. We recommend RED because the traffic class may include multiple simultaneous flows to and from the file server.

Figure 3-25: Configuration for guaranteeing VoIP traffic while maintaining file server traffic



```
# Guaranteeing VoIP traffic and maintaining file server downloads
# Central office configuration

set system name=central
set system territory=europe

# Configure ISDN and PPP
add user=remote password=rempass login=no telnet=no
add isdn call=office number=0 prec=in dir=in searchsub=local
create ppp=0 over=isdn-office authentication=chap echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.1.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.2.0 int=ppp0 next=0.0.0.0

# Create a classifier for VoIP (This example assumes voice traffic uses UDP ports
# between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a classifier for traffic from the file server
create class=3 ipsadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Create a traffic class for file server traffic, give it the next highest priority
# and use a medium RED curve set
create sqos trafficclass=3 priority=13 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-3

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1
add sqos trafficclass=2 classifier=2
add sqos trafficclass=3 classifier=3

# Use the policy on ppp0
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

```
# Guaranteeing VoIP traffic and maintaining file server downloads
# Remote office configuration

set system name=remote
set system territory=europe

# Configure ISDN and PPP
add isdn call=office number=your-central-office-number prec=out outsub=local
create ppp=0 over=isdn-office username=remote passw=rempass echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.2.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.1.0 int=ppp0 next=0.0.0.0

# Create a classifier for VoIP (This example assumes voice traffic uses UDP ports
# between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a classifier for traffic to the file server
create class=3 ipdadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Create a traffic class for file server traffic, give it the next highest priority
# and use a medium RED curve set
create sqos trafficclass=3 priority=13 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-3

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1
add sqos trafficclass=2 classifier=2
add sqos trafficclass=3 classifier=3

# Use the policy on ppp0
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

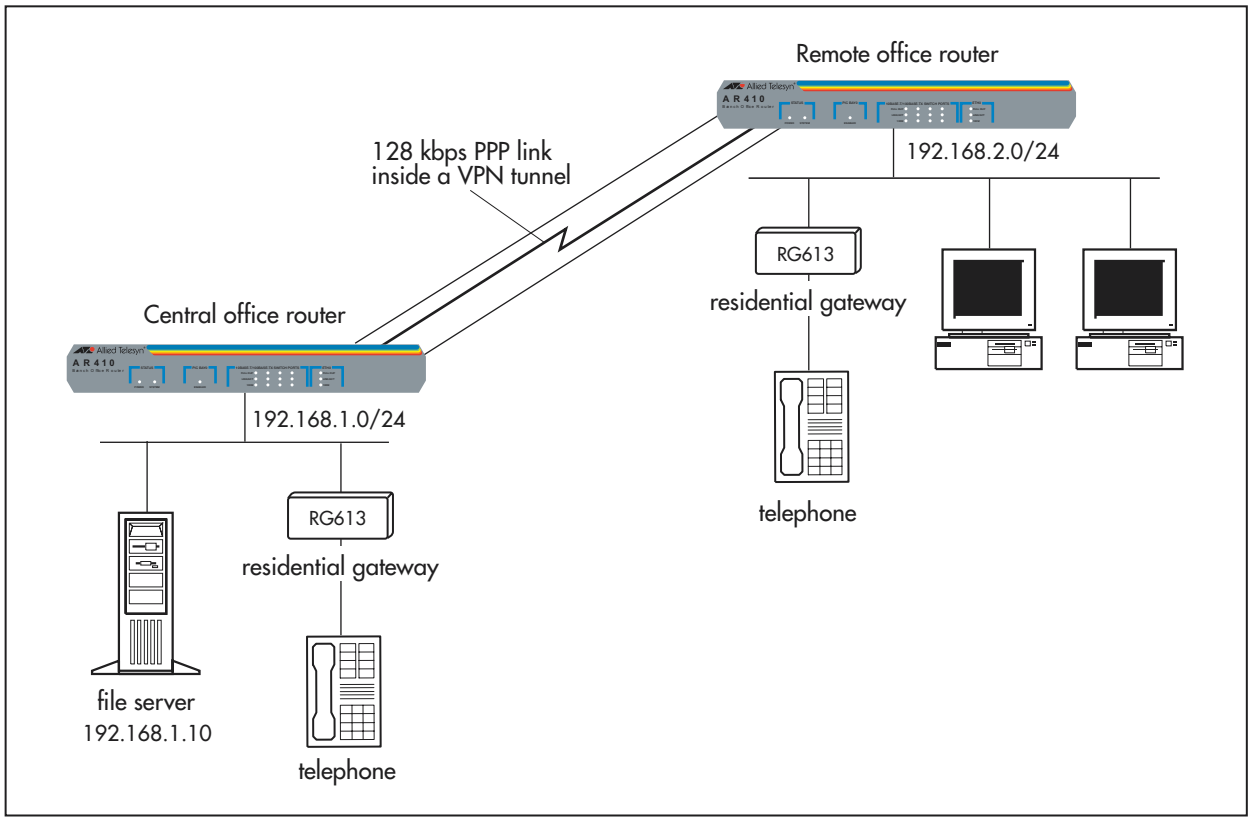
4: Guaranteeing VoIP Traffic over a VPN Tunnel

This scenario expands on scenario 3 by sending all traffic between the central office and the remote office securely over a VPN tunnel.

In this scenario (Figure 3-26):

- VoIP traffic has the highest priority possible and a short queue, so the switches always transmit VoIP traffic with low drop rate, delay, and jitter.
- the switches identify VoIP traffic by the UDP ports it uses.
- File server traffic has the next highest priority. The switches use a medium RED curve set to drop file server traffic as necessary and control the TCP flows. We recommend RED because the traffic class may include multiple simultaneous flows to and from the file server.

Figure 3-26: Configuration for software QoS for traffic over a VPN tunnel



```
# Guaranteeing VoIP traffic and maintaining file server downloads over a VPN tunnel
# Central office configuration

set system name=central
set system territory=europe

# Create a user with Security Officer privilege and enable secure mode
add user=secoff pass=verysecret priv=securityOfficer lo=yes
set user=secoff telnet=no netmask=255.255.255.255
enable system security_mode
set user securedelay=600

# create an encryption key for ISAKMP to protect and authenticate its messages
create enco key=1 type=general value=123456789

# Configure ISDN and PPP
add user=remote password=rempass login=no telnet=no
add isdn call=office number=0 prec=in dir=in searchsub=local
create ppp=0 over=isdn-office authentication=chap echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip int=vlan1 ip=192.168.1.254
add ip int=ppp0 ip=10.0.0.1
add ip rou=192.168.2.0 int=ppp0 next=10.0.0.2

# Configure ISAKMP
create isakmp pol=office pe=10.0.0.2 key=1
enable isakmp

# Configure IPsec
create ipsec sas=1 key=isakmp prot=esp enc=des hasha=sha
create ipsec bund=1 key=isakmp string=1

# Create policies which allow ISAKMP to bypass IPsec processing, but other
# traffic to be processed by IPSec
create ipsec pol=isakmp int=ppp0 ac=permit lp=500 rp=500
create ipsec pol=office key=isakmp isa=office int=ppp0 ac=ipsec bund=1 peer=10.0.0.2
set ipsec pol=office lad=192.168.1.0 lma=255.255.255.0 rad=192.168.2.0
rma=255.255.255.0
enable ipsec

# Create a classifier for VoIP (This example assumes voice traffic uses UDP ports
# between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a classifier for traffic from the file server
create class=3 ipsadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10
```

This configuration continues on the next page.

.

```
# Guaranteeing VoIP traffic and maintaining file server downloads over a VPN tunnel
# Central office configuration continued

# Create a traffic class for the SIP signalling traffic and give it the second
# highest priority
create sqos trafficclass=2 priority=14

# Create a traffic class for file server traffic, give it the next highest priority
# and use a medium RED curve set
create sqos trafficclass=3 priority=13 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-3

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1
add sqos trafficclass=2 classifier=2
add sqos trafficclass=3 classifier=3

# Use the policy on the IPSec tunnel
set sqos interface=ipsec-office tunnelpolicy=1

# Enable software QoS
enable sqos
```

```
# Guaranteeing VoIP traffic and maintaining file server downloads over a VPN tunnel
# Remote office configuration

set system name=remote
set system territory=europe

# Create Security Officer priviledge user
add user=secoff pass=friend priv=securityOfficer lo=yes
set user=secoff telnet=no netmask=255.255.255.255
enable system security_mode
set user securedelay=600

# Create an encryption key for ISAKMP to protect and authenticate its messages
create enco key=1 type=general value=123456789

# Configure ISDN and PPP
add isdn call=office number=your-central-office-number prec=out outsub=local
create ppp=0 over=isdn-office username=remote passw=rempass echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.2.254
add ip interface=ppp0 ip=10.0.0.2
add ip route=192.168.1.0 int=ppp0 next=10.0.0.1

# Configure ISAKMP
create isakmp pol=office pe=10.0.0.1 key=1
enable isakmp

# Configure IPsec
create ipsec sas=1 key=isakmp prot=esp enc=des hasha=sha
create ipsec bund=1 key=isakmp string=1

# Create policies which allow ISAKMP to bypass IPsec processing, but other
# traffic to be processed by IPsec
create ipsec pol=isakmp int=ppp0 ac=permit lp=500 rp=500
create ipsec pol=office key=isakmp isa=office int=ppp0 ac=ipsec bund=1 peer=10.0.0.1
set ipsec pol=office lad=192.168.2.0 lma=255.255.255.0 rad=192.168.1.0
rma=255.255.255.0
enable ipsec

# Create a classifier for VoIP (This example assumes voice traffic uses UDP ports
# between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a classifier for traffic to the file server
create class=3 ipdadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10
```

This configuration continues on the next page.

```
# Guaranteeing VoIP traffic and maintaining file server downloads over a VPN tunnel
# Remote office configuration continued

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Create a traffic class for file server traffic, give it the next highest priority
# and use a medium RED curve set
create sqos trafficclass=3 priority=13 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-3

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1
add sqos trafficclass=2 classifier=2
add sqos trafficclass=3 classifier=3

# Use the policy on the ipsec tunnel
set sqos interface=ipsec-office tunnelpolicy=1

# Enable software QoS
enable sqos
```

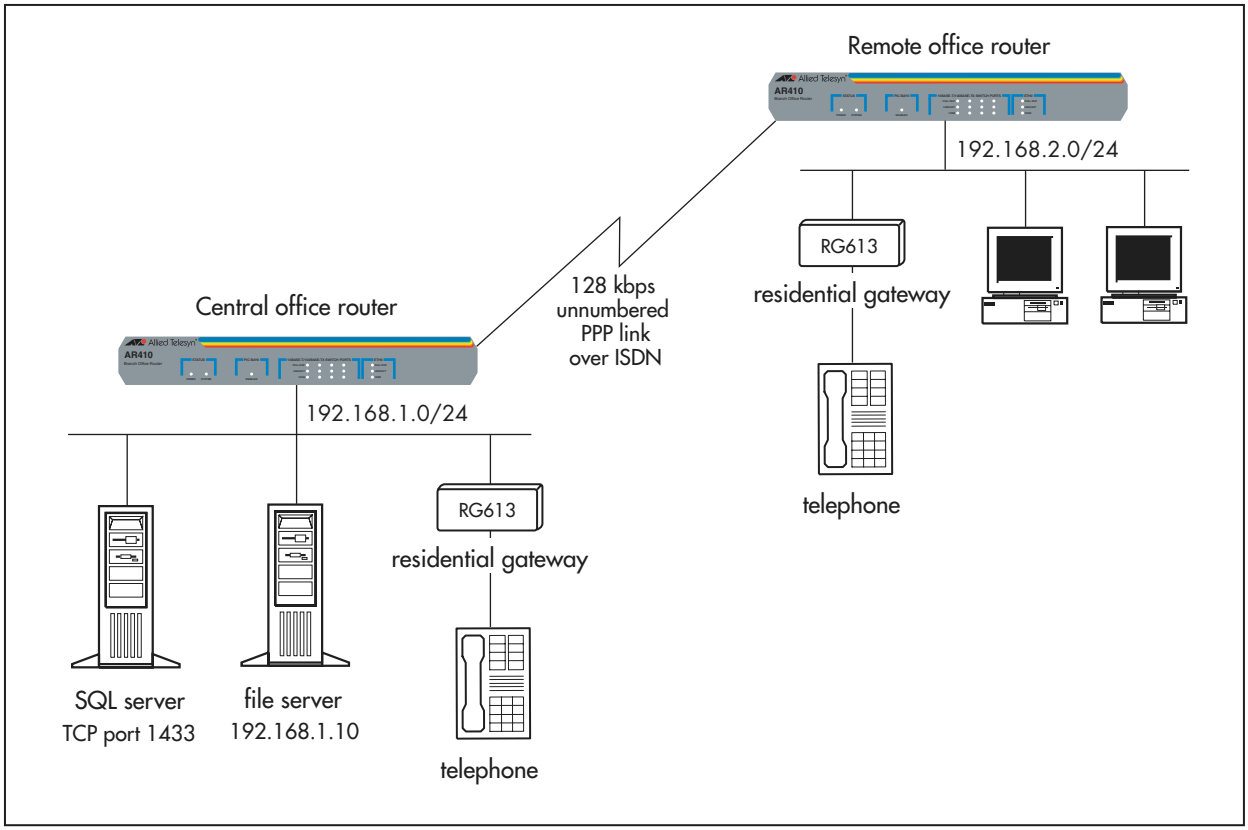
5: VoIP, Critical Database, and File Server Traffic

This scenario expands on scenario 3 by adding a critical SQL database.

In this scenario (Figure 3-27):

- VoIP traffic has the highest priority possible and a short queue, so the switches always transmit VoIP traffic with low drop rate, delay, and jitter.
- the switches identify VoIP traffic by the UDP ports it uses.
- SQL traffic has the next highest priority. The switches use a medium RED curve set to drop SQL server traffic as necessary. We recommend RED because the traffic class may include multiple simultaneous flows to and from the server.
- File server traffic has the next highest priority, so is only sent if there is no VoIP or SQL traffic waiting. The switches use a medium RED curve set to drop file server traffic as necessary and control the TCP flows. We recommend RED because the traffic class may include multiple simultaneous flows to and from the file server.

Figure 3-27: Configuration for VoIP, critical database, and file server traffic



```
# Guaranteeing VoIP traffic and maintaining SQL and file server downloads
# Central office configuration

set system name=central
set system territory=europe

# Configure ISDN and PPP
add user=remote password=rempass login=no telnet=no
add isdn call=office number=0 prec=in dir=in searchsub=local
create ppp=0 over=isdn-office authentication=chap echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.1.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.2.0 int=ppp0 next=0.0.0.0

# Create classifiers for VoIP (this example assumes voice traffic uses UDP ports
# between 16300 and 16320), and to match the SIP signaling traffic
create class=1 udpdport=16300-16320
create class=2 udpdport=5060

# Create a classifier for traffic from the SQL server
create class=3 tcpsport=1433

# Create a classifier for traffic from the file server
create class=4 ipsadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Create a traffic class for SQL server traffic, give it the next highest priority
# and use a medium RED curve set
create sqos trafficclass=3 priority=13 red=1

# Create a traffic class for file server traffic, give it a lower priority
# and use a medium RED curve set
create sqos trafficclass=4 priority=12 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-4

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1
add sqos trafficclass=2 classifier=2
add sqos trafficclass=3 classifier=3
add sqos trafficclass=4 classifier=4

# Use the policy on ppp
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

```
# Guaranteeing VoIP traffic and maintaining SQL and file server downloads
# Remote office configuration

set system name=remote
set system territory=europe

# Configure ISDN and PPP
add isdn call=office number=your-central-office-number prec=out outsub=local
create ppp=0 over=isdn-office username=remote passw=rempass echo=30 lqr=off bap=off

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.2.254
add ip interface=ppp0 ip=0.0.0.0 mask=0.0.0.0
add ip route=192.168.1.0 int=ppp0 next=0.0.0.0

# Create classifiers for VoIP (this example assumes voice traffic uses UDP ports
# between 16300 and 16320), and to match the SIP signaling traffic
create class=1 udpdport=16300-16320
create class=2 udpdport=5060

# Create a classifier for traffic to the SQL server
create class=3 tcpdport=1433

# Create a classifier for traffic to the file server
create class=4 ipdadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create a traffic class for VoIP traffic and give it the highest priority and a
# short queue
create sqos trafficclass=1 priority=15 maxqlen=10

# Create a traffic class for the SIP signalling traffic and give it the second highest
# priority
create sqos trafficclass=2 priority=14

# Create a traffic class for SQL server traffic, give it the next highest priority
# and use a medium RED curve set
create sqos trafficclass=3 priority=13 red=1

# Create a traffic class for file server traffic, give it a lower priority
# and use a medium RED curve set
create sqos trafficclass=4 priority=12 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-4

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1
add sqos trafficclass=2 classifier=2
add sqos trafficclass=3 classifier=3
add sqos trafficclass=4 classifier=4

# Use the policy on ppp
set sqos interface=ppp0 outpolicy=1

# Enable software QoS
enable sqos
```

6: Multiple Applications over Frame Relay

This scenario shows multiple applications running over a 1.5 Mbps frame relay link, including voice, video conferencing, network monitoring, and server traffic.

In this scenario (Figure 3-28 on page 3-81):

- A traffic class tree prioritises voice, video conferencing, and network monitoring traffic, higher than server downloads, while weighting the server downloads so that both SQL and file server data can be sent (Figure 3-29 on page 3-82).
- VoIP traffic has the highest priority possible and a short queue, so the switches always transmit VoIP traffic with low drop rate, delay, and jitter. Voice has a minimum bandwidth guarantee of 300 kbps, which allows 6-8 calls at once, and is limited to 500 kbps.
- Video conferencing traffic has the next highest priority. It is limited to 250 kbps, which only allows one video conference at a time.
- Network monitoring traffic (SNMP) has the next highest priority.
- SQL and file server traffic, combined, have the next highest priority, so is only sent if there is no VoIP, video, or network monitoring traffic waiting. SQL and file server traffic share a weighted intermediate traffic class, with SQL traffic having 70% of the weight and file server traffic 30%. This prevents the SQL traffic from throttling the file server downloads.
- The switches use a medium RED curve set to drop SQL and file server traffic as necessary. We recommend RED because the traffic classes may include multiple simultaneous flows to and from the servers.

Figure 3-28: Configuration for software QoS with multiple applications

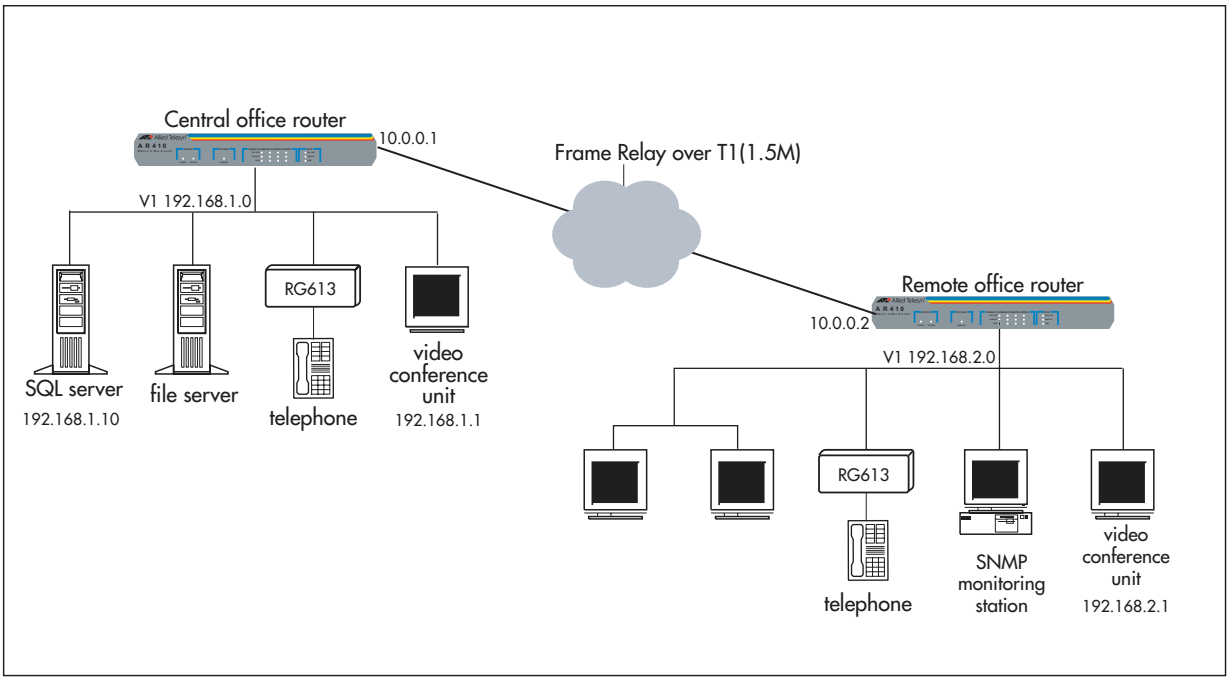
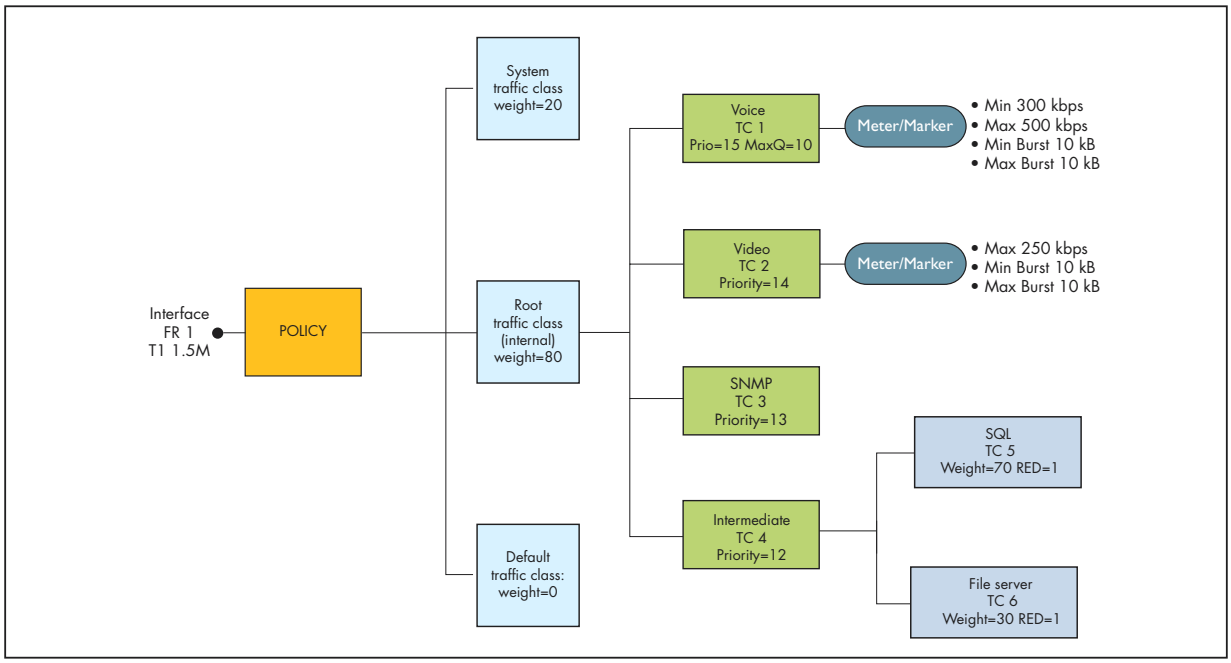


Figure 3-29: Traffic class tree for multiple applications over a frame relay link




```
# VoIP, Video, SNMP and Data traffic between two offices using FR/T1 (1.5M) link
# Central office configuration

set system name=central

# Configure the Frame Relay interface over the T1 (1.5M) PRI link
set pri=bay0.pri0 mode=tdm
set pri=bay0.pri0 cl=int

create tdm group=office interface=bay0.pri0 unstructured

create fr=1 over=office lmischeme=none
add fr=1 li=301 type=ptp
add fr=1 dlc=301
set fr=1 dlc=301 li=301

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.1.254
add ip interface=fr1.301 ip=10.0.0.1
add ip route=192.168.2.0 int=fr1.301 next=10.0.0.2

# Create a classifier for VoIP (This example assumes voice traffic uses UDP ports
# between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a classifier for the video conferencing traffic
create class=3 ipsadd=192.168.1.1

# Create a classifier for the SNMP traffic
create class=4 udpdport=161
```

This configuration continues on the next page.

```
# VoIP, Video, SNMP and Data traffic between two offices using FR/T1 (1.5M) link
# Central office configuration continued

# Create a classifier for traffic from the SQL server
create class=5 tcpsport=1433

# Create a classifier for traffic from the file server
create class=6 ipsadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create meters for the voice and video traffic
create sqos meter=1 descrip=voice type=trtcm min=300K max=500K minbu=10K maxbu=10K
create sqos meter=2 descrip=video type=srtcm max=250K minbu=10K maxbu=10K

# Create a traffic class for VoIP/SIP traffic and give it the highest priority and
# a short queue. Assign the meter and set it to drop traffic over the maximum
# bandwidth setting
create sqos trafficclass=1 priority=15 maxqlen=10 meter=1 bwclass3action=drop

# Create a traffic class for video traffic and give it the second highest priority.
# Assign the meter and set it to drop traffic over the maximum bandwidth setting
create sqos trafficclass=2 priority=14 meter=2 bwclass3action=drop

# Create a traffic class for SNMP traffic and give it the next highest priority
# after the real-time voice and video
create sqos trafficclass=3 priority=13

# Create an intermediate traffic class which will have sub traffic classes for SQL
# and file server traffic attached.  Give it a lower priority and set it to use WRR
# to send SQL and file server traffic
create sqos trafficclass=4 priority=12 weightscheduler=wrr

# Create a traffic class for SQL server traffic, give it a higher weight than the
# file server traffic class and use a medium RED curve set
create sqos trafficclass=5 weight=70 red=1

# Create a traffic class for file server traffic, give it a lower weight than the
# SQL traffic class and use a medium RED curve set
create sqos trafficclass=6 weight=30 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-4

# Add the sub traffic classes for SQL and File server traffic to intermediate traffic
# class 4
add sqos trafficclass=4 subclass=5,6

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1,2
add sqos trafficclass=2 classifier=3
add sqos trafficclass=3 classifier=4
add sqos trafficclass=5 classifier=5
add sqos trafficclass=6 classifier=6

# Use the policy on FR1
set sqos interface=fr1 outpolicy=1

# Enable software QoS
enable sqos
```

```
# VoIP, Video, SNMP and Data traffic between two offices using FR/T1 (1.5M) link
# Remote office configuration

set system name=remote

# Configure the Frame Relay interface over the T1 (1.5M) PRI link
set pri=bay0.pri0 mode=tdm

create tdm group=office interface=bay0.pri0 unstructured

create fr=1 over=office lmischeme=none
add fr=1 li=301 type=ptp
add fr=1 dlc=301
set fr=1 dlc=301 li=301

# Configure IP
enable ip
add ip interface=vlan1 ip=192.168.1.254
add ip interface=fr1.301 ip=10.0.0.2
add ip route=192.168.1.0 int=fr1.301 next=10.0.0.1

# Create a classifier for VoIP (This example assumes voice traffic uses UDP ports
# between 16300 and 16320)
create class=1 udpdport=16300-16320

# Create a classifier to match on the SIP signaling traffic
create class=2 udpdport=5060

# Create a classifier for the video conferencing traffic
create class=3 ipsadd=192.168.2.1

# Create a classifier for the SNMP traffic
create class=4 udpdport=161

# Create a classifier for traffic to the SQL server
create class=5 tcpdport=1433

# Create a classifier for traffic to the file server
create class=6 ipdadd=192.168.1.10

# Create a QoS policy
create sqos poli=1

# Create meters for the voice and video traffic
create sqos meter=1 descrip=voice type=trtcm min=300K max=500K minbu=10K maxbu=10K
create sqos meter=2 descrip=video type=srtcm max=250K minbu=10K maxbu=10K

# Create a traffic class for VoIP/SIP traffic and give it the highest priority and
# a short queue. Assign the meter and set it to drop traffic over the maximum
# bandwidth setting
create sqos trafficclass=1 priority=15 maxqlen=10 meter=1 bwclass3action=drop

# Create a traffic class for Video traffic and give it the second highest priority.
# Assign the meter and set it to drop traffic over the maximum bandwidth setting
create sqos trafficclass=2 priority=14 meter=2 bwclass3action=drop
```

This configuration continues on the next page.

```
# VoIP, Video, SNMP and Data traffic between two offices using FR/T1 (1.5M) link
# Remote office configuration continued

# Create a traffic class for the SNMP traffic, giving it the next highest priority
# after the real-time voice and video
create sqos trafficclass=3 priority=13

# Create an intermediate traffic class which will have sub traffic classes for SQL
# and data traffic attached.Give it a lower priority and set it to use WRR to send
# SQL and data
create sqos trafficclass=4 priority=12 weightscheduler=wrr

# Create a traffic class for SQL server traffic, give it a higher weight than the
# file server traffic class and use a medium RED curve set
create sqos trafficclass=5 weight=70 red=1

# Create a traffic class for File server traffic, give it a lower weight than the
# SQL traffic class and use a medium RED curve set
create sqos trafficclass=6 weight=30 red=1

# Add the traffic classes to the policy
add sqos policy=1 trafficclass=1-4

# Add the sub traffic classes for SQL and file server traffic to intermediate traffic
# class 4
add sqos trafficclass=4 subclass=5,6

# Add the classifiers to the traffic classes
add sqos trafficclass=1 classifier=1,2
add sqos trafficclass=2 classifier=3
add sqos trafficclass=3 classifier=4
add sqos trafficclass=5 classifier=5
add sqos trafficclass=6 classifier=6

# Use the policy on FR1
set sqos interface=fr1 outpolicy=1

# Enable software QoS
enable sqos
```

Command Reference

This section describes the commands available for configuring and monitoring Software QoS on the switch.

For each parameter and option, the shortest string you can enter is shown in capital letters in the Syntax section.

See Conventions in the Preface for additional conventions used to describe command syntax. See Appendix A, Messages for a complete list of messages and meanings.

add sqos interface dar

Syntax `ADD SQOS INTerface=`*`interface`* `DAR=`*`id-list`*

Description This command adds one or more DAR (Dynamic Application Recognition) objects to the interface, which should be the interface at which voice or video session initiation control messages arrive. The switch uses the DAR object to identify matching session initiation control messages, and creates dynamic classifiers for the associated voice or video flow.

Parameter	Description
INTerface	The interface or tunnel to add the DAR object to. Valid entry types are: Layer 1 and 2 interfaces: ● eth (e.g. eth0) ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● PPP (e.g. ppp0) ● the switch instance (swi0) ● the switch instance on AR750S routers (swi0) Layer 3 tunnels: ● GRE (e.g. gre1) ● IPv6 6-to-4 virtual interface (e.g. virt9) ● the name of an IPSec policy (ipsec-*policyname*) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command.
DAR	The DAR object to add to the interface. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. The DAR objects must already exist. Each DAR object can only belong to one interface.
* The shortest string you can enter is shown in capital letters.	

Example To attach DAR object 0 to ppp0, use the command:

```
add sqos int=ppp0 dar=0
```

To attach DAR object 0 to the IPsec policy *central*, use the command:

```
add sqos int=ipsec-central dar=0
```

Related Commands

add sqos trafficclass dar
create sqos dar
delete sqos interface dar
destroy sqos dar
set sqos dar
set sqos interface
show sqos dar

add sqos policy trafficclass

Syntax `ADD SQOS POLIcy=0..9999 TRafficclass=`*`id-list`*

Description This command adds one or more traffic classes to the specified QoS policy.

Parameter	Description
POLIcy	The policy to add the traffic class to. An integer in the range 0 to 9999. The policy must already exist.
TRafficclass	The traffic class to add to the policy. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. The traffic classes must already exist. Each traffic class can only belong to one policy.

* The shortest string you can enter is shown in capital letters.

Example To create a traffic class hierarchy (shown in Figure 3-30 on page 3-91) in which:

- policy 1 contains leaf traffic class 1 and intermediate traffic class 2
- intermediate traffic class 2 contains intermediate traffic class 3 and leaf traffic class 4
- intermediate traffic class 3 contains leaf traffic classes 5 and 6

first create the traffic classes and the policy, then combine them into the hierarchy using the commands:

```
add sqos poli=1 tr=1,2
add sqos tr=2 subc=3,4
add sqos tr=3 subc=5,6
```

Related Commands

create sqos policy
create sqos trafficclass
delete sqos policy trafficclass
destroy sqos policy
destroy sqos trafficclass
set sqos policy
set sqos trafficclass
show sqos policy
show sqos trafficclass

add sqos trafficclass classifier

Syntax `ADD SQOS TRafficclass=0..9999 CLASSifier=`*`id-list`*

Description This command adds one or more classifiers to the specified traffic class. The traffic class must be a leaf traffic class; you cannot also add sub traffic classes to it.

Parameter	Description
TRafficclass	The leaf traffic class to add the classifier to. An integer in the range 0 to 9999. The traffic class must already exist.
CLASSifier	The classifier to add to the traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. The classifiers must already exist. Each classifier can only belong to one traffic class.

* The shortest string you can enter is shown in capital letters.

Example To add classifiers 1, 2, 4, 5, and 6 to leaf traffic class 3, use the command:

```
add sqos tr=3 class=1,2,4-6
```

Related Commands

add sqos trafficclass classifier
create sqos trafficclass
delete sqos trafficclass classifier
destroy sqos trafficclass
set sqos trafficclass
show sqos trafficclass

add sqos trafficclass dar

Syntax `ADD SQOS TRafficclass=0..9999 DAR=`*`id-list`*

Description This command associates one or more DAR (Dynamic Application Recognition) objects with the specified traffic class. The traffic class will process the voice or video traffic flows that the DAR objects identify.

The traffic class must be a leaf traffic class; you cannot also add sub traffic classes to it.

Parameter	Description
TRafficclass	The leaf traffic class to add the DAR object to. An integer in the range 0 to 9999. The traffic class must already exist.
DAR	The DAR object to add to the traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. The DAR objects must already exist. Each DAR object can only belong to one traffic class.

* The shortest string you can enter is shown in capital letters.

Example To add DAR objects 1, 2, 4, 5, and 6 to leaf traffic class 3, use the command:

```
add sqos tr=3 dar=1,2,4-6
```

Related Commands

add sqos interface dar
create sqos dar
create sqos trafficclass
delete sqos interface dar
delete sqos trafficclass dar
destroy sqos dar
set sqos dar
set sqos trafficclass
show sqos dar
show sqos trafficclass

add sqos trafficclass subclass

Syntax `ADD SQOS TRafficclass=0..9999 SUBClass=`*`id-list`*

Description This command adds one or more traffic classes to the specified traffic class, to create a traffic class hierarchy.

Parameter	Description
TRafficclass	The intermediate traffic class to add the sub traffic class to. An integer in the range 0 to 9999. The intermediate traffic class must already exist and must be attached to a policy.
SUBClass	The sub traffic class to add to the intermediate traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. The sub traffic classes must already exist. Each sub traffic class can only belong to one intermediate traffic class.
* The shortest string you can enter is shown in capital letters.	

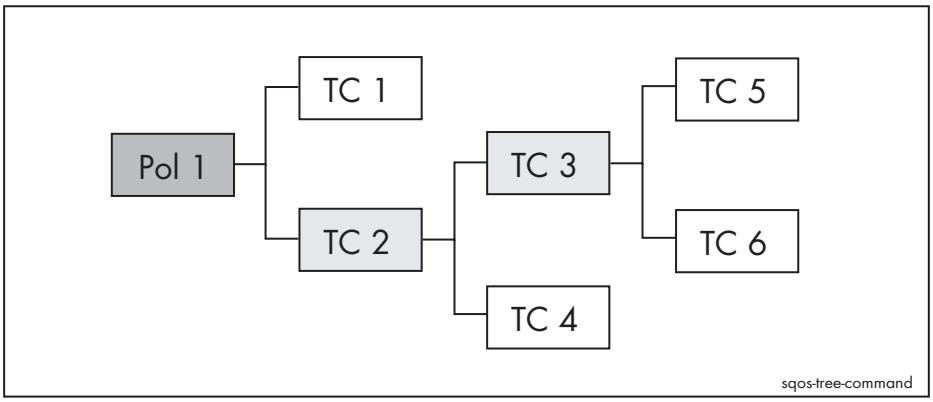
Example To create a traffic class hierarchy (shown in Figure 3-30) in which:

- policy 1 contains leaf traffic class 1 and intermediate traffic class 2
- intermediate traffic class 2 contains intermediate traffic class 3 and leaf traffic class 4
- intermediate traffic class 3 contains leaf traffic classes 5 and 6

first create the traffic classes and the policy, then combine them into the hierarchy using the commands:

```
add sqos poli=1 tr=1,2
add sqos tr=2 subc=3,4
add sqos tr=3 subc=5,6
```

Figure 3-30: An example of a traffic class tree



Related Commands

add sqos policy trafficclass
add sqos trafficclass classifier
add sqos trafficclass dar
create sqos trafficclass
delete sqos trafficclass subclass
destroy sqos trafficclass
set sqos trafficclass
show sqos trafficclass

create sqos dar

Syntax

```
CREate SQOS DAR=id-list [CODEC={AUDio|VIDeo|ANY}]
   [DESCription=description]
   [DSTIp={ipadd[/0..32]|ipv6add[/0..128]}]
   [SRCIp={ipadd[/0..32]|ipv6add[/0..128]}]
   [INACtivetimeout={1..3600|NONE}]
   [PROTocol={SIP|RTSp|H323|ALL}]
   [H323Port=1..65535] [RTSPPort=1..65535]
   [SIPPort=1..65535]
```

Description

This command creates one or more Dynamic Application Recognition objects. The switch identifies voice or video packets that match the settings in the DAR object and dynamically creates a classifier for that traffic flow.

You can create up to 64 DAR objects.

After you have created the DAR object, use the **add sqos trafficclass dar** command on page 3-90 to assign it to a leaf traffic class, and the **add sqos interface dar** command on page 3-86 to assign it to the interface at which voice or video session initiation control packets arrive.

Parameter	Description	
DAR	The ID number of the new DAR object. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
CODEC	The coder/decoder for the DAR object to use to match packets. Default: **any**	
	AUDio	Matches traffic flows that use any audio codec.
	VIDeo	Matches traffic flows that use any video codec.
	ANY	The DAR object ignores the codec. A DAR object with **codec=any** will match any sessions set up by the specified protocol, not just voice and video sessions.
DESCription	A description of the DAR object, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
DSTIp	The destination IPv4 or IPv6 address or subnet. The DAR object only matches voice or video traffic flows to that address or network. IPv4 addresses are specified in dotted decimal notation. IPv6 addresses are specified as eight pairs of octets, separated by colons. The CIDR mask for IPv4 and prefix length for IPv6 are optional. For IPv4, if you specify a subnet address without specifying a mask, the default mask for that subnet will be used. For IPv6, the default prefix length is 128. If you also specify **srcip**, either both must be IPv4 addresses or both must be IPv6 addresses. Default: no default (ignores destination IP address)	
H323Port	The TCP port that H.323 session control messages are received on. Default: **1720**	

Parameter	Description
INACtivetimeout	A time in the range 1 to 3600 seconds (up to 60 minutes). If a classified flow is idle for this length of time, its entry is deleted. Default: **600**
PROTocol	The protocol for the DAR object to use to match packets. Default: **all** (ignores protocol)
RTSPPort	The TCP port that RTSP session control messages are received on. Default: **554**
SIPPort	The UDP port that SIP messages are received on. Default: **5060**
SRCIp	The source IPv4 or IPv6 address or subnet. The DAR object only matches voice or video traffic flows from that address or network. IPv4 addresses are specified in dotted decimal notation. IPv6 addresses are specified as eight pairs of octets, separated by colons. The CIDR mask for IPv4 and prefix length for IPv6 are optional. For IPv4, if you specify a subnet address without specifying a mask, the default mask for that subnet will be used. For IPv6, the default prefix length is 128. If you also specify **dstip**, either both must be IPv4 addresses or both must be IPv6 addresses. Default: no default (ignores source IP address)
* The shortest string you can enter is shown in capital letters.	

Example To create a DAR object to identify and classify voice packets destined for the 192.168.1.0 subnet, use the command:

```
cre sqos dar=0 codec=audio dsti=192.168.1.0/24
```

Related Commands

add sqos interface dar
add sqos trafficclass dar
delete sqos interface dar
destroy sqos dar
set sqos dar
show sqos dar

create sqos dscpmap

Syntax `CREate SQOS DSCPMap=`*`id-list`* `[DESCription=`*`description`*`]`

Description This command creates one or more DSCP maps, with the default settings.

DSCP maps consist of a premarking table and a remarking table. Software QoS uses the premarking table before the metering stage, to map a packet's DSCP value to a new DSCP and/or bandwidth class. The map's default settings are to map all DSCPs to bandwidth class 1 (green). Software QoS uses the remarking table after the metering stage. It uses the bandwidth class the meter assigned the packet to, and the packet's DSCP value, to give the packet a new DSCP and/or bandwidth class. The map's default settings are to leave all values unchanged. Use the **set sqos dscpmap** command on page 3-126 to change these defaults.

You can create up to 64 maps.

Parameter	Description
DSCPMap	The ID number of the new DSCP map. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 1,3,4-9). An integer cannot appear in the list more than once.
DESCription	A description of the DSCP map, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default

* The shortest string you can enter is shown in capital letters.

Example To create a DSCP map to use to remark non-conformant traffic with a DSCP of 10, use the commands:

```
cre dscpm=1 desc=nonconformant_10
set dscpm=1 table=rem bwc=3 newd=10
```

Related Commands

create sqos trafficclass
destroy sqos dscpmap
set sqos dscpmap
show sqos dscpmap

create sqos meter

Syntax

```
CREate SQOS METer=id-list [DESCription=description]
   [IGNorebwclass={Yes|No}]
   [MINbandwidth=rate[Kbps|Mbps|Gbps]]
   [MAXbandwidth=rate[Kbps|Mbps|Gbps]]
   [MINBUrstsize=burstsize[Bytes|Kbytes|Mbytes|Gbytes]]
   [MAXBUrstsize=burstsize[Bytes|Kbytes|Mbytes|Gbytes]]
   [TYPE={SRtcm|TRtcm}]
```

Description

This command creates one or more Three Colour Marker meters, as described in RFC 2697, *A Single Rate Three Color Marker*, September 1999 and RFC 2698, *A Two Rate Three Color Marker*, September 1999. The meter measures how much bandwidth the packets in a traffic flow use, and how well the bandwidth use conforms with the bandwidth specifications for the traffic class that the flow belongs to. It assigns the packet to a bandwidth class depending on its conformance (Table 3-4 on page 3-21).

You can create up to 64 meters.

Parameter	Description	
METer	The ID number of the new meter. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
DESCription	A description of the meter, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
IGNorebwclass	Whether the meter acknowledges any previous bandwidth class assigned to packets. A meter that acknowledges previous conformance is called *colour aware*. Default: **no** (the meter is colour aware)	
	Yes	The metering function is colour blind and ignores any bandwidth class previously assigned to packets. It sets the meter bandwidth class according to only the metered conformance level of the flow.
	No	The metering function is colour aware and uses any bandwidth class previously assigned to packets, as well as the metered conformance level, to set the bandwidth class. Packets previously labelled non-conformant (bandwidth class 3, red) will remain in bandwidth class 3. Packets previously labelled partially-conformant (bandwidth class 2, yellow) will be assigned to bandwidth class 2 or 3, depending on metered conformance.

Parameter	Description
MAXbandwidth	For the single rate meter of RFC 2697, the highest rate at which a steady stream of packets can arrive at the meter and be assigned to bandwidth class 1 (conformant, green). This is the Committed Information Rate (CIR) of the RFC. For the two rate meter of RFC 2698, the Peak Information Rate (PIR) of the RFC. See "Metering: Bandwidth conformance" on page 3-22 for a description of PIR. It must equal or exceed **minbandwidth**. *rate* is in the range 0 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Default: **1Mbps**
MAXBUrstsize	For the single rate meter of RFC 2697, the amount by which a packet can exceed **maxbandwidth** plus **minburstsize** and still possibly be assigned to bandwidth class 2. This is the Excess Burst Size (EBS) of the RFC. For the two rate meter of RFC 2698, the amount by which a packet can exceed **maxbandwidth** and still possibly be assigned to bandwidth class 2. This is the Peak Burst Size (PBS) of the RFC. *burstsize* is in the range 0 to 16777216 bytes (16 MB), specified in bytes, kbytes, Mbytes, or Gbytes (in upper or lower case). If you do not specify a unit, it uses bytes. If you specify kB, MB or GB, *burstsize* may contain a decimal fraction, for example, 1.25M. For a single rate meter, at least one of **minburstsize** and **maxburstsize** should equal or exceed the size of the largest IP packet you expect on the metered flow. For a two rate meter, both **minburstsize** and **maxburstsize** must exceed 0 (zero), and the RFC recommends that both values equal or exceed the size of the largest IP packet you expect on the metered flow. To create a single rate two colour meter (green and red), set **maxburstsize** to 0 (zero). Default: **10kbytes**
MINbandwidth	For the two rate meter of RFC 2698, the Committed Information Rate (CIR) of the RFC. See "Metering: Bandwidth conformance" on page 3-22 for a description of CIR. It must not exceed **maxbandwidth**. *rate* is in the range 0 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Only valid if **type**=**trtcm.** Default: **1Mbps**

Parameter	Description
MINBUrstsize	For the single rate meter of RFC 2697, the amount by which a packet can exceed **maxbandwidth** and still possibly be assigned to bandwidth class 1. This is the Committed Burst Size (CBS) of the RFC. For the two rate meter of RFC 2698, the amount by which a packet can exceed **minbandwidth** and still possibly be assigned to bandwidth class 1. This is the Committed Burst Size (CBS) of the RFC. *burstsize* is in the range 0 to 16777216 bytes (16 MB), specified in bytes, kbytes, Mbytes, or Gbytes (in upper or lower case). If you do not specify a unit, it uses bytes. If you specify kB, MB or GB, *burstsize* may contain a decimal fraction, for example, 1.25M. For a single rate meter, at least one of **minburstsize** and **maxburstsize** should equal or exceed the size of the largest IP packet you expect on the metered flow. For a two rate meter, both **minburstsize** and **maxburstsize** must exceed 0 (zero), and the RFC recommends that both values equal or exceed the size of the largest IP packet you expect on the metered flow. For a single rate meter, if you set **minburstsize** to 0 (zero) the meter will assign all packets to bandwidth class 2 or 3. Default: **10kbytes**
TYPE	The type of meter. "Metering: Bandwidth conformance" on page 3-22 describes the meters. Default: SRTCM
	SRTCM The Single Rate Three Colour Marker of RFC 2697.
	TRTCM The Two Rate Three Colour Marker of RFC 2698.
* The shortest string you can enter is shown in capital letters.	

Example To create a colour-blind single-rate meter with default settings otherwise, use the command:

```
cre sqos met=0 ign=y
```

Related Commands

create sqos trafficclass
destroy sqos meter
set sqos meter
show sqos meter

create sqos policy

Syntax

```
CREate SQOS POLIcy=id-list
    [BWClass3action={DROP|PAUSE|NONE}]
    [DEFaulttrafficclass={0..9999|NONE}]
    [DESCription=description] [DSCPMap={0..9999|NONE}]
    [IGNOREPrenatinfo={YES|NO}] [METer={0..9999|NONE}]
    [PAUSEAction={NONE|LOg|TRap|BOth}] [PAUSETIme={1..30}]
    [REMarking={0..63|USEDscpmap|NONE}]
    [REMARKVlanpri={0..7|NONE}] [SYSTEMTraffic={5..50}]
    [VIRTbw={bandwidth[Kbps|Mbps|Gbps]|NONE}]
    [WEIGhtscheduler={WRr|DWrr}]
```

Description

This command creates one or more QoS policies. A policy defines the overall QoS processing for an interface. You can create up to 64 policies.

After you have created the policy, use the **set sqos interface** command on page 3-128 to assign it to the required interface.

Parameter	Description	
POLIcy	The ID number of the new policy. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
BWClass3action	The action the switch takes on Bandwidth Class 3 packets (red coloured packets). These are packets that exceed their allocated bandwidth, as determined at the metering stage. Default: **none**	
	DROP	The switch drops non-conformant packets.
	PAUSE	The switch drops non-conformant packets and stops dequeuing packets from the flow for **pausetime** seconds.
	NONE	The switch sends non-conformant packets to the next processing stage.
DEFaulttrafficclass	The traffic class that the switch applies to unclassified traffic on the policy's interface. It must be a leaf traffic class. Default: no default	
	0..9999	The traffic class ID.
	NONE	No user-nominated default traffic class. The switch uses the default traffic class that it made when you created the policy.
DEscription	A description of the policy, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
DScpmap	The DSCP map to assign to the policy. An integer in the range 0 to 9999. Default: **none**	

Parameter	Description	
IGNOREPrenatinfo	Whether classifiers attached to the policy use pre-NAT IP settings for classification because these contain the distinguishing information. Default: **no** (uses pre-NAT settings)	
MEter	The meter to assign to the policy. An integer in the range 0 to 9999. The meter determines a new bandwidth class (colour) for packets that are processed using this policy. You can configure the policy or a traffic class to drop or queue the packets on the basis of the new bandwidth class. Default: **none**	
PAUSEAction	The notification action taken by the switch when it pauses a non-conformant traffic flow that belongs to this policy. Only valid if **bwclass3action**=**pause**. Default: **none**	
	LOg	The switch generates a log message.
	TRap	The switch generates an SNMP trap.
	BOth	The switch generates both a log message and an SNMP trap.
	NONE	The switch does not generate a notification.
PAUSETIme	The length of time, in the range 1 to 30 seconds, for which the switch does not dequeue packets from a paused flow. Only valid if **bwclass3action**=**pause**. Default: **10**	
REMarking	How the switch sets the bandwidth class and/or DSCP value in the packet header's Differentiated Services field after metering. Default: **none**	
	0..63	The switch writes the specified value into the DSCP bits in the packet header.
	USEDscpmap	The switch uses the metered bandwidth class and current DSCP value, in conjunction with the policy's DSCP map, to determine the new DSCP value and/or bandwidth class. You must also specify the **dscpmap** parameter.
	NONE	The switch does not modify the DSCP value or metered bandwidth class.
REMARKVlanpri	What the switch sets the 802.1p VLAN priority field of the frame's Ethernet header to. Default: **none**	
	0..7	The switch writes the specified value into the 802.1p VLAN priority field of the Ethernet header.
	NONE	The switch does not modify the 802.1p VLAN priority field of the Ethernet header.
SYSTEMTraffic	The percentage of the interface's maximum bandwidth that the switch reserves for system traffic, in the range 5 to 50%. Default: **20**	

Parameter	Description	
VIRTbw	The maximum bandwidth available to the policy. **Virtbw** determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media. This rate is not equivalent to the transmission rate for data seen on the line, because the actual transmission rate includes the transmission of bits for the inter-frame-gap and the preamble of the layer 2 headers. For example, 10 Mbps of data leaving the internal queues is not equivalent to 10 Mbps of data transmitted on the line. *rate* is in the range 1 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Default: **none** (bandwidth is not limited)	
WEIGhtscheduler	The queue scheduling method for weighted traffic classes that belong to the policy. Weighted traffic classes assign weights to flows instead of priorities. Default: **wrr**	
	WRr	The switch uses a weighted round robin scheme to empty the queues of weighted traffic classes.
	DWrr	The switch uses a deficit weighted round robin scheme to empty the queues of weighted traffic classes. DWRR is less biased towards large packets than WRR.
* The shortest string you can enter is shown in capital letters.		

Example To create a policy that allocates 15% of the available bandwidth to system traffic, use the command:

```
cre sqos poli=0 systemt=15
```

To create a policy that uses meter 1 to measure bandwidth, drops non-conformant packets, and uses DSCP map 1 to remark conformant packets, use the command:

```
cre sqos poli=0 bwc=drop dscpm=1 rem=used
```

Related Commands

add sqos policy trafficclass
delete sqos policy trafficclass
destroy sqos policy
set sqos interface
set sqos policy
show sqos policy

create sqos red

Syntax

```
CREate SQOS RED=id-list
   [AVERaging=0..99] [DESCription=description]
   [START1=0..100] [STOP1=0..100] [DROP1=0..100]
   [START2=0..100] [STOP2=0..100] [DROP2=0..100]
   [START3=0..100] [STOP3=0..100] [DROP3=0..100]
```

Description

This command creates one or more sets of RED curves. Red curve sets 0-2 exist by default, and cannot be modified or deleted. You can create up to 61 more RED curve sets. Table 3-6 on page 3-27 shows the properties of the default red curve sets.

Parameter	Description
RED	The ID number of the new RED curve. *id-list* is an integer in the range 3 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 3,4-9). An integer cannot appear in the list more than once.
AVERaging	A weight used in the moving averaging estimation of queue length for the RED curve algorithm. The estimated queue length is frequently updated, and is calculated by taking a weighted average of the previous average and the current instantaneous queue length. **Averaging** is the weight given to the previous average in this weighted calculation. If **averaging** is too high, the estimated average queue size responds too slowly to transient congestion. If **averaging** is too low, the estimated average queue size tracks the instantaneous queue size too closely and you lose the benefits of RED. RED works best when the estimated average queue length responds as slowly as possible while preventing the queue from becoming full. To achieve this, set **averaging** to a lower value if the queue constantly becomes full, so that the estimated average queue size more closely tracks the actual queue size. To check how often the queue becomes full, use the **trafficclass** parameter of the **show sqos counters** command on page 3-145 and check the queue counters, or set **qlimitexceedaction** and check the log messages or SNMP traps. Default: **98**
DESCription	An optional description of the RED curve set, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default
START1 START2 START3	The percentage of the queue length at which the RED algorithm starts to drop packets, for packets in bandwidth class 1, 2 and 3 respectively. A percentage in the range 1 to 100. Default: **35**

Parameter	Description
STOP1 STOP2 STOP3	The percentage of the queue length at which the RED algorithm is dropping **drop** percent of the packets, for packets in bandwidth class 1, 2 and 3 respectively. Beyond this point, 100% of the packets are dropped. This value must be larger than **start** and is a percentage in the range 1 to 100. Default: **65**
DROP1 DROP2 DROP3	The probability that a packet will be dropped at the **stop** queue length, for packets in bandwidth class 1, 2 and 3 respectively. A percentage in the range 1 to 100. Default: **30**

* The shortest string you can enter is shown in capital letters.

Example To create a moderately-passive RED curve set, use the command:

```
cre red=3 desc=mod-passive
   start1=65 stop1=85 drop1=20
   start2=40 stop2=65 drop2=30
   start3=30 stop3=45 drop3=40
```

Related Commands

create sqos trafficclass
destroy sqos red
set sqos red
show sqos red

create sqos trafficclass

Syntax

```
CREate SQOS TRafficclass=id-list
   [BWClass3action={DRop|PAuse|NONE}]
   [DESCription=description] [MAXQlen=1..1023]
   [METer={0..9999|NONE}]
   [PAUSEAction={NONE|LOg|TRap|BOth}] [PAUSETIme={1..30}]
   [PREMARKBwcl={1..3|USEDscpmap}]
   [PREMARKDscp={0..63|USEDscpmap|NONE}]
   [{PRIOrity=0..15|WEIght=0..100}]
   [QLIMITExceedaction={NONE|LOg|TRap|BOth}]
   [QUEUEDrop={Head|Tail}] [RED={0..9999|NONE}]
   [REMarking=0..63|USEDscpmap|NONE}]
   [REMARKVlanpri={0..7|NONE}]
   [VIRTbw={bandwidth[Kbps|Mbps|Gbps]|NONE}]
   [WEIGhtscheduler={WRr|DWrr}]
```

Description

This command creates one or more traffic classes. A traffic class specifies the QoS actions for a set of flows. You can create up to 1024 traffic class.

After you have created the traffic class, use the **add sqos trafficclass subclass** command on page 3-91 to assign it to an intermediate traffic class, or the **add sqos policy trafficclass** command on page 3-88 to assign it to a policy.

Parameter	Description	
TRafficclass	The ID number of the new traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
DEscription	A description of the traffic class, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
BWClass3action	The action the switch takes on Bandwidth Class 3 packets (red coloured packets). These are packets that exceed their allocated bandwidth, as determined at the metering stage. Default: **none**	
	DRop	The switch drops non-conformant packets.
	PAuse	The switch drops non-conformant packets and stops dequeuing packets from the flow for **pausetime** seconds.
	NONE	The switch sends non-conformant packets to the next processing stage.
MAxqlen	The maximum queue length, in the range 1 to 1023 packets, for the traffic class. The switch drops packets that would exceed the maximum queue length. If you shape traffic by specifying a virtual bandwidth for a policy or traffic class (intermediate or leaf), give the appropriate leaf traffic classes large maximum queue lengths. This enables them to buffer bursts of packets and avoids packet loss. **maxqlen** is only valid on leaf traffic classes. Default: **64**	

Parameter	Description	
MEter	The meter to assign to the traffic class. An integer in the range 0 to 9999. The meter determines a new bandwidth class (colour) for packets that are processed using this traffic class. You can configure the traffic class, or the policy it is attached to, to drop or queue the packets on the basis of the new bandwidth class. Default: **none**	
PAUSEAction	The notification action taken by the switch when it pauses a non-conformant traffic flow that belongs to this traffic class. Only valid if **bwclass3action**=**pause**. Default: **none**	
	LOg	The switch generates a log message.
	TRap	The switch generates an SNMP trap.
	BOth	The switch generates both a log message and an SNMP trap.
	NONE	The switch does not generate a notification.
PAUSETIme	The length of time, in the range 0 to 30 seconds, for which the switch does not dequeue packets from a paused flow. If you specify 0, the switch will take the action in **pauseaction**, but not pause the flow. Only valid if **bwclass3action**=**pause**. Default: **10**	
QLIMITExceedaction	The notification action taken by the switch when a traffic flow exceeds the maximum queue length of the traffic class. Default: **none**	
	LOg	The switch generates a log message.
	TRap	The switch generates an SNMP trap.
	BOth	The switch generates both a log message and an SNMP trap.
	NONE	The switch does not generate a notification.
PREMARKBwcl	How the switch assigns the packet to a bandwidth class at the start of the QoS processing (before metering). The switch can use the assigned value in metering, marking and RED processing. You can only specify premarking in leaf traffic classes. Default: **1**	
	1..3	The switch assigns the packet to the specified bandwidth class.
	USEDscpmap	The switch uses the current DSCP value in conjunction with the policy's DSCP map to determine the bandwidth class. You must also specify the DSCP map by using the **dscpmap** parameter in the **create sqos policy** command on page 3-99 or the **set sqos policy** command on page 3-133.

Parameter	Description	
PREMARKDscp	How the switch changes the DSCP value in the packet header at the start of the QoS processing (before metering). The switch can use the assigned value in metering, marking and RED processing. You can only specify premarking in leaf traffic classes. Default: **none**	
	0..63	The switch writes the specified DSCP value into the packet header.
	USEDscpmap	The switch uses the current DSCP value in conjunction with the policy's DSCP map to determine the new DSCP. You must also specify the DSCP map by using the **dscpmap** parameter in the **create sqos policy** command on page 3-99 or the **set sqos policy** command on page 3-133.
	NONE	The switch does not change the packet DSCP value.
PRIOrity	The priority of the traffic class, an integer in the range 0 to 15. Specifying **priority** in traffic classes sets their policy (or intermediate traffic class) to schedule queues according to the relative priorities of all its traffic classes. The switch services the queue from the traffic class with the highest value for **priority** first. **Priority** and **weight** are mutually exclusive. Use the **priority** parameter to create a hierarchy based on the priority of flows, for strict priority queuing. Use the **weight** parameter to create a hierarchy with weighted flows, for WRR or DWRR queuing. If you create a mixed hierarchy the priority queues are emptied first, giving low latency queuing behaviour. Default: **1**	
QUEUEDrop	Whether packets are dropped from the head or tail of the queue when the queue becomes full. Tail dropping drops the newest packets; head dropping drops the oldest. Default: **tail**	
RED	The RED curve set that the switch uses for early dropping of packets. An integer in the range 0 to 9999. Default: **none**	
REMarking	How the switch sets the bandwidth class and/or the DSCP value in the packet header's Differentiated Services field after metering. Default: **none**	
	0..63	The switch writes the specified value into the DSCP bits in the packet header.
	USEDscpmap	The switch uses the metered bandwidth class and current DSCP value, in conjunction with the policy's DSCP map, to determine the new DSCP value and/or bandwidth class.You must also specify the **dscpmap** parameter in the **create sqos policy** command on page 3-99 or the **set sqos policy** command on page 3-133.
	NONE	The switch does not modify the DSCP value or metered bandwidth class.

Parameter	Description	
REMARKVlanpri	What the switch sets the 802.1p VLAN priority field of the frame's Ethernet header to. Default: **none**	
	0..7	The switch writes the specified value into the 802.1p VLAN priority field of the Ethernet header.
	NONE	The switch does not modify the 802.1p VLAN priority field of the Ethernet header.
VIRTbw	The maximum bandwidth available to the traffic class. **Virtbw** determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media. This rate is not equivalent to the transmission rate for data seen on the line, because the actual transmission rate includes the transmission of bits for the inter-frame-gap and the preamble of the layer 2 headers. For example, 10 Mbps of data leaving the internal queues is not equivalent to 10 Mbps of data transmitted on the line. *rate* is in the range 1 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Default: **none** (bandwidth is not limited)	
WEIght	The weight given to the traffic class, in the range 0 to 100. Specifying **weight** in traffic classes sets their policy (or intermediate traffic class) to schedule queues according to the relative weights of all its traffic classes. If a traffic class has a weight of 0 (zero), the switch only empties its queue once the queues of all its sibling traffic classes are empty. **Priority** and **weight** are mutually exclusive. Use the **priority** parameter to create a hierarchy based on the priority of flows, for strict priority queuing. Use the **weight** parameter to create a hierarchy with weighted flows, for WRR or DWRR queuing. If you create a mixed hierarchy the priority queues are emptied first, giving low latency queuing behaviour. Default: no default, because the default behaviour is priority-based hierarchies.	
WEIGhtscheduler	The queue scheduling method that the switch uses to schedule this traffic class' weighted sub traffic classes. This parameter is only valid if the sub traffic classes specify the **weight** parameter. Default: **wrr**	
	WRr	The switch uses a weighted round robin scheme to empty the queues of weighted sub traffic classes.
	DWrr	The switch uses a deficit weighted round robin scheme to empty the queues of weighted sub traffic classes. DWRR is less biased towards large packets than WRR.

* The shortest string you can enter is shown in capital letters.

Example

To create a trafficclass that uses meter 1 to measure bandwidth, drops non-conformant packets, and uses the policy's DSCP map to remark conformant packets, use the command:

```
cre sqos tr=0 bwc=dr rem=used
```

To create a traffic class with a moderately-high priority, and use DWRR to schedule the queues of the traffic class' sub classes, use the command:

```
cre sqos tr=1 prio=10 weig=dw
```

Related Commands

add sqos policy trafficclass
add sqos trafficclass classifier
add sqos trafficclass dar
add sqos trafficclass subclass
delete sqos policy trafficclass
delete sqos trafficclass classifier
delete sqos trafficclass dar
delete sqos trafficclass subclass
destroy sqos trafficclass
set sqos trafficclass
show sqos trafficclass

delete sqos interface dar

Syntax `DELete SQOS INTerface=`*`interface`* `DAR={`*`id-list`*`|ALL}`

Description This command removes one or more DAR (Dynamic Application Recognition) objects from the interface. It does not destroy the DAR objects.

Parameter	Description
INTerface	The interface or tunnel to remove the DAR object from. Valid entry types are: Layer 1 and 2 interfaces: ● eth (e.g. eth0) ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● PPP (e.g. ppp0) ● the switch instance (swi0) ● the switch instance on AR750S routers (swi0) Layer 3 tunnels: ● GRE (e.g. gre1) ● IPv6 6-to-4 virtual interface (e.g. virt9) ● the name of an IPSec policy (ipsec-*policyname*) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command.
DAR	The DAR object to remove from the interface. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default

* The shortest string you can enter is shown in capital letters.

Example To stop using DAR object 0 to identify voice or video sessions that are setup on ppp0, use the command:

```
del sqos int=ppp0 dar=0
```

To stop using any DAR objects to identify voice or video sessions that are setup on ppp0, use the command:

```
del sqos int=ppp0 dar=all
```

To stop using any DAR objects to identify voice or video sessions that are setup over the IPsec policy *central*, use the command:

```
del sqos int=ipsec-central dar=all
```

Related Commands

add sqos interface dar
create sqos dar
delete sqos trafficclass dar
destroy sqos dar
set sqos dar
show sqos dar

delete sqos policy trafficclass

Syntax `DELete SQOS POLIcy=0..9999 TRafficclass={`*`id-list`*`|ALL}`

Description This command removes one or more traffic classes from the specified QoS policy. It does not destroy the traffic classes, or detach any sub traffic classes from the traffic classes.

Parameter	Description
POLIcy	The policy to remove the traffic class from. An integer in the range 0 to 9999.
TRafficclass	The traffic class to remove from the policy. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default

* The shortest string you can enter is shown in capital letters.

Example To remove leaf traffic class 1 and intermediate traffic class 2 from policy 1, use the command:

```
del sqos poli=1 tr=1,2
```

To remove all traffic classes from policy 1, use the command:

```
del sqos poli=1 tr=all
```

Related Commands

add sqos policy trafficclass
create sqos policy
create sqos trafficclass
destroy sqos policy
destroy sqos trafficclass
set sqos policy
set sqos trafficclass
show sqos policy
show sqos trafficclass

delete sqos trafficclass classifier

Syntax `DELete SQOS TRafficclass=0..9999 CLASSifier={`*`id-list`*`|ALL}`

Description This command removes one or more classifiers from the specified leaf traffic class. It does not destroy the classifiers.

Parameter	Description
TRafficclass	The leaf traffic class to remove the classifier from. An integer in the range 0 to 9999.
CLASSifier	The classifier to remove from the traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default

* The shortest string you can enter is shown in capital letters.

Example To remove classifiers 1, 2, 4, 5, and 6 from leaf traffic class 3, use the command:

```
del sqos tr=3 class=1,2,4-6
```

To remove all classifiers from leaf traffic class 3, use the command:

```
del sqos tr=3 class=all
```

Related Commands

add sqos trafficclass classifier
delete sqos policy trafficclass
delete sqos trafficclass subclass
destroy sqos trafficclass
set sqos trafficclass
show sqos trafficclass

delete sqos trafficclass dar

Syntax `DELete SQOS TRafficclass=0..9999 DAR={`*`id-list`*`|ALL}`

Description This command disassociates one or more DAR (Dynamic Application Recognition) objects from the specified traffic class. It does not destroy the DAR object.

Parameter	Description
TRafficclass	The leaf traffic class to remove the DAR object from. An integer in the range 0 to 9999.
DAR	The DAR object to remove from the traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default

* The shortest string you can enter is shown in capital letters.

Example To remove DAR objects 1, 2, 4, 5, and 6 from leaf traffic class 3, use the command:

```
del sqos tr=3 dar=1,2,4-6
```

To remove all DAR objects from leaf traffic class 3, use the command:

```
del sqos tr=3 dar=all
```

Related Commands

add sqos interface dar
add sqos trafficclass dar
create sqos dar
create sqos trafficclass
delete sqos interface dar
delete sqos trafficclass classifier
delete sqos trafficclass subclass
destroy sqos dar
destroy sqos trafficclass
set sqos dar
set sqos trafficclass
show sqos dar
show sqos trafficclass

delete sqos trafficclass subclass

Syntax `DELete SQOS TRafficclass=0..9999 SUBClass={`*`id-list`*`|ALL}`

Description This command removes one or more sub traffic classes from the specified intermediate traffic class. It does not destroy the traffic classes.

Parameter	Description
TRafficclass	The intermediate traffic class to remove the sub traffic class from. An integer in the range 0 to 9999.
SUBClass	The sub traffic class to remove from the intermediate traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default

* The shortest string you can enter is shown in capital letters.

Example To remove leaf traffic classes 2 and 3 from intermediate traffic class 1, use the command:

```
del sqos tr=1 subc=2,3
```

To remove all sub traffic classes from intermediate traffic class 1, use the command:

```
del sqos tr=1 subc=all
```

Related Commands

add sqos trafficclass subclass
create sqos trafficclass
delete sqos trafficclass classifier
destroy sqos trafficclass
set sqos trafficclass
show sqos trafficclass

destroy sqos dar

Syntax `DESTroy SQOS DAR=`*`id-list`*

Description This command destroys one or more DAR objects. You cannot destroy a DAR object if an interface or traffic class uses it.

The *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.

Example To destroy DAR objects 1, 2 and 3, use the command:

```
dest sqos dar=1-3
```

Related Commands
add sqos interface dar
add sqos trafficclass dar
create sqos dar
delete sqos interface dar
delete sqos trafficclass dar
set sqos dar
show sqos dar

destroy sqos dscpmap

Syntax `DESTroy SQOS DSCPMap=`*`id-list`*

Description This command destroys one or more DSCP maps. You cannot destroy a DSCP map if a policy uses it.

The *id-list* is an integer in the range 1 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 1,3,4-9). An integer cannot appear in the list more than once.

Example To destroy DSCP maps 1, 2 and 3, use the command:

```
dest sqos dscpm=1-3
```

Related Commands
create sqos dscpmap
create sqos trafficclass
set sqos dscpmap
show sqos dscpmap

destroy sqos meter

Syntax `DESTroy SQOS METer=`*`id-list`*

Description This command destroys one or more meters. You cannot destroy a meter if a policy or traffic class uses it.

The *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.

Example To destroy meters 1, 2 and 3, use the command:

```
dest sqos met=1-3
```

Related Commands
create sqos trafficclass
create sqos meter
set sqos meter
show sqos meter

destroy sqos policy

Syntax `DESTroy SQOS POLIcy=`*`id-list`*

Description This command destroys one or more policies. You cannot destroy a policy if it is attached to an interface or if any traffic classes are attached to it.

The *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.

Example To destroy policies 1, 2 and 3, use the command:

```
dest sqos poli=1-3
```

Related Commands
add sqos policy trafficclass
create sqos policy
delete sqos policy trafficclass
set sqos policy
show sqos policy

destroy sqos red

Syntax `DESTroy SQOS RED=`*`id-list`*

Description This command destroys one or more RED curve sets. You cannot destroy a RED curve set if a traffic class uses it. You cannot destroy the default RED curve sets (0-2).

The *id-list* is an integer in the range 3 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 3,4-9). An integer cannot appear in the list more than once.

Example To destroy RED curve set 3, use the command:

```
dest sqos red=3
```

Related Commands
create sqos red
create sqos trafficclass
set sqos red
show sqos red

destroy sqos trafficclass

Syntax `DESTroy SQOS TRafficclass=`*`id-list`*

Description This command destroys one or more traffic classes. You cannot destroy a traffic class if it is attached to a policy, or if any sub traffic classes or classifiers are attached to it.

The *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma-separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.

Example To destroy traffic classes 1, 2 and 3, use the command:

```
dest sqos tr=1-3
```

Related Commands
delete sqos policy trafficclass
delete sqos trafficclass classifier
delete sqos trafficclass dar
delete sqos trafficclass subclass
create sqos trafficclass
set sqos trafficclass
show sqos trafficclass

disable sqos

Syntax `DISable SQOS`

Description This command disables software QoS. Software QoS is disabled by default.

Example To disable software QoS, use the command:

```
dis sqos
```

Related Commands
disable sqos debug
enable sqos
enable sqos debug
purge sqos
show sqos
show sqos counters

disable sqos debug

Syntax

```
DISable SQOS
   DEBug={ALL|DAR|DARDATA|ENGine|ERRor|INFo|MARk|PKT}
```

Description

This command disables software QoS debugging. Debugging is disabled by default.

Parameter	Description	
DEBug	The debug mode to disable. Default: no default	
	ALL	All debugging modes.
	DAR	Notifications when DAR objects and instances are created or destroyed.
	DARDATA	More detailed information about voice and video data.
	ENGine	Debugging information related to the packet conditioning engine.
	ERRor	Critical error debugging information, including a stack trace.
	INFo	General debugging information.
	MARk	Packet marking debugging information.
	PKT	Packet debugging.
* The shortest string you can enter is shown in capital letters.		

Example

To disable packet debugging, use the command:

```
dis sqos deb=pkt
```

Related Commands

disable sqos
enable sqos debug
enable sqos
purge sqos
show sqos
show sqos counters

enable sqos

Syntax `ENAble SQOS`

Description This command enables software QoS. Software QoS is disabled by default.

Example To enable software QoS, use the command:

```
ena sqos
```

Related Commands
disable sqos
disable sqos debug
enable sqos debug
purge sqos
show sqos
show sqos counters

enable sqos debug

Syntax

```
ENAble SQOS
   DEBug={ALL|DAR|DARDATA|ENGine|ERRor|INFo|MARk|PKT}
```

Description

This command enables software QoS debugging. Debugging is disabled by default.

Parameter	Description	
DEBug	The debug mode to enable. Default: no default	
	ALL	All debugging modes.
	DAR	Notifications when DAR objects and instances are created or destroyed.
	DARDATA	More detailed information about SIP and RTSP data.
	ENGine	Debugging information related to the packet conditioning engine.
	ERRor	Critical error debugging information, including a stack trace.
	INFo	General debugging information.
	MARk	Packet marking debugging information.
	PKT	Packet debugging.
* The shortest string you can enter is shown in capital letters.		

Example

To enable packet debugging, use the command:

```
ena sqos deb=pkt
```

Related Commands

disable sqos
disable sqos debug
enable sqos
purge sqos
show sqos
show sqos counters

purge sqos

Syntax `PURge SQOS {DAR|INTerface|POLIcy|TRafficclass}`

Description This command destroys all or a section of software QoS configuration. If you specify **purge sqos** with no other parameters, all software QoS configuration is destroyed.

Parameter	Description
DAR	Destroy all DAR objects.
INTerface	Remove software QoS policies from all interfaces. The interfaces and policies are not destroyed.
POLIcy	Destroy all software QoS policies. This parameter destroys the traffic class hierarchy by also detaching all sub traffic classes from intermediate traffic classes.
TRafficclass	Destroy all traffic classes.

* The shortest string you can enter is shown in capital letters.

Example To destroy all software QoS configuration, use the command:

```
pur sqos
```

To destroy all software QoS policies and the traffic class hierarchy, while leaving the traffic classes intact, use the command:

```
pur sqos poli
```

Related Commands

disable sqos
disable sqos debug
enable sqos
enable sqos debug
show sqos
show sqos counters

reset sqos counters

Syntax

```
RESET SQOS COUnters POLIcy[=id-list]
   [DIrection={In|OUT|TUNnel}] [INTerface=interface]

RESET SQOS COUnters TRafficclass[=id-list]
   [DIrection={In|OUT|TUNnel|ALL}] [INTerface=interface]

RESET SQOS COUnters CLASSifier[=id-list]
   [DIrection={In|OUT|TUNnel|ALL}] [INTerface=interface]
   [TRafficclass=id-list]

RESET SQOS COUnters DAR[=id-list]
   [DIrection={In|OUT|TUNnel}] [INTerface=interface]
```

Description This command resets counter information for the specified software QoS objects.

Parameter	Description	
CLASSifier	The classifier to reset the counters for. An integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default (classifier counters are not reset)	
DAR	The DAR object to reset the counters for. An integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default (DAR object counters are not reset)	
DIrection	A filter that restricts the command, so that the switch only resets counters for software QoS objects with this direction. Default: no default (not filtered by direction)	
	IN	Reset counters for software QoS objects that act on the packet at ingress.
	OUT	Reset counters for software QoS objects that act on the packet at egress.
	TUNnel	Reset counters for software QoS objects that act on tunnelled packets.
	ALL	Reset counters for software QoS objects that act on packets in any direction. **All** is only valid with **trafficclass** and **classifier**.

Parameter	Description
INTerface	A filter that restricts the command, so that the switch only resets counters for software QoS objects that are associated with this interface or tunnel. Valid entry types are: Layer 1 and 2 interfaces: ● eth (e.g. eth0) ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● PPP (e.g. ppp0) ● the switch instance (swi0) ● the switch instance on AR750S routers (swi0) Layer 3 tunnels: ● GRE (e.g. gre1) ● IPv6 6-to-4 virtual interface (e.g. virt9) ● the name of an IPSec policy (ipsec-*policyname*) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command on page 6-104 of Chapter 6, Interfaces. Default: no default (not filtered by interface)
POLIcy	The policy to reset the counters for. An integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default (policy counters are not reset)
TRafficclass	The traffic class. An integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. If you specify both **classifier** and **trafficclass** the switch resets the classifier counters for that traffic class; otherwise it resets the traffic class counters. Default: no default
* The shortest string you can enter is shown in capital letters.	

Example To reset all classifier counters for traffic class 1, use the command:

```
reset sqos cou class tr=1
```

To reset ingress software QoS policy counters on ppp0, use the command:

```
reset sqos cou poli di=in int=ppp0
```

Related Commands

disable sqos debug
enable sqos debug
show sqos
show sqos counters

set sqos dar

Syntax

```
SET SQOS DAR=id-list [CODEC={AUDio|VIDeo|ANY}]
   [DESCription=description]
   [DSTIp={ipadd[/0..32]|ipv6add[/0..128]}]
   [SRCIp={ipadd[/0..32]|ipv6add[/0..128]}]
   [INACtivetimeout={1..3600|NONE}]
   [PROTocol={SIP|RTSp|H323|ALL}]
   [H323Port=1..65535] [RTSPPort=1..65535]
   [SIPPort=1..65535]
```

Description This command modifies one or more Dynamic Application Recognition objects.

Parameter	Description	
DAR	The ID number of the DAR object. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
CODEC	The coder/decoder for the DAR object to use to match packets. Default: **any**	
	AUDio	Matches traffic flows that use any audio codec.
	VIDeo	Matches traffic flows that use any video codec.
	ANY	The DAR object ignores the codec. A DAR object with **codec=any** will match any sessions set up by the specified protocol, not just voice and video sessions.
DESCription	A description of the DAR object, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
DSTIp	The destination IPv4 or IPv6 address or subnet. The DAR object only matches voice or video traffic flows to that address or network. IPv4 addresses are specified in dotted decimal notation. IPv6 addresses are specified as eight pairs of octets, separated by colons. The CIDR mask for IPv4 and prefix length for IPv6 are optional. For IPv4, if you specify a subnet address without specifying a mask, the default mask for that subnet will be used. For IPv6, the default prefix length is 128. If you also specify **srcip**, either both must be IPv4 addresses or both must be IPv6 addresses. Default: no default (ignores destination IP address)	
H323Port	The TCP port that H.323 session control messages are received on. Default: **1720**	
INACtivetimeout	A time in the range 1 to 3600 seconds (up to 60 minutes). If a classified flow is idle for this length of time, its entry is deleted. Default: **600**	
PROTocol	The protocol for the DAR object to use to match packets. Default: **all** (ignores protocol)	
RTSPPort	The TCP port that RTSP session control messages are received on. Default: **554**	

Parameter	Description
SIPPort	The UDP port that SIP messages are received on. Default: **5060**
SRCIp	The source IPv4 or IPv6 address or subnet. The DAR object only matches voice or video traffic flows from that address or network. IPv4 addresses are specified in dotted decimal notation. IPv6 addresses are specified as eight pairs of octets, separated by colons. The CIDR mask for IPv4 and prefix length for IPv6 are optional. For IPv4, if you specify a subnet address without specifying a mask, the default mask for that subnet will be used. For IPv6, the default prefix length is 128. If you also specify **dstip**, either both must be IPv4 addresses or both must be IPv6 addresses. Default: no default (ignores source IP address)

* The shortest string you can enter is shown in capital letters.

Example To modify DAR object 0 so that it identifies and classifies voice packets destined for the 192.168.1.0 subnet, use the command:

```
set sqos dar=0 codec=audio dsti=192.168.1.0/24
```

Related Commands

add sqos interface dar
add sqos trafficclass dar
create sqos dar
delete sqos interface dar
delete sqos trafficclass dar
destroy sqos dar
show sqos dar

set sqos dscpmap

Syntax To set the premarking table:

```
SET SQOS DSCPMap=id-list TAble=PREmark
   [DESCription=description] [DSCP=dscp-list]
   [NEWBwclass=1..3] [NEWDscp=0..63]
```

To set the remarking table:

```
SET SQOS DSCPMap=id-list TAble=REMark
   [DESCription=description] [BWClass=bwclass-list]
   [DSCP=dscp-list] [NEWBwclass=1..3] [NEWDscp=0..63]
```

To change only the description:

```
SET SQOS DSCPMap=id-list [DESCription=description]
```

Description This command configures one or more DSCP maps. Each map consists of a premarking table and a remarking table. To modify both, use separate commands.

Parameter	Description
DSCPMap	The ID number of the DSCP map. An integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 1,4-9). An integer cannot appear in the list more than once.
BWClass	The bandwidth class to use as an index into the remarking table. An integer in the range 1 to 3, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 1,3). The switch writes the **newdscp** and/or **newbwclass** into the entries in the table that have this bandwidth class and the specified **dscp** (if any). If you specify neither **dscp** or **bwclass**, the switch gives the **newdscp** and/or **newbwclass** to all remarking table entries**.** Default: no default
DESCription	A description of the DSCP map, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default
DSCP	The DSCP value to use as an index into the premarking or remarking table. An integer in the range 0 to 63, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 1,4-9). The switch writes the **newdscp** and/or **newbwclass** into the entries in the table that have this DSCP and for the remarking table the specified **bwclass** (if any). If you specify neither **dscp** or, for the remarking table, **bwclass**, the switch gives the **newdscp** and/or **newbwclass** to all entries**.**
NEWBwclass	The new bandwidth class for entries with the specified **dscp** and/or, for the remarking table, **bwclass**. An integer in the range 1 to 3. Default: no default
NEWDscp	The new DSCP value for entries with the specified **dscp** and/or, for the remarking table, **bwclass**. An integer in the range 0 to 63. Default: no default

Parameter	Description	
TAble	The table in the DSCP map to configure. Default: no default	
	PREmark	Used before the metering stage.
	REMark	Used after metering has taken place, after the packet has been dequeued from the leaf traffic class.
* The shortest string you can enter is shown in capital letters.		

Example To set DSCP map 1 so that the remark table gives non-conformant traffic a DSCP of 10, use the command:

```
set dscpm=1 table=rem bwc=3 newd=10
```

Related Commands

create sqos dscpmap
create sqos trafficclass
destroy sqos dscpmap
show sqos dscpmap

set sqos interface

Syntax

```
SET SQOS INTerface=interface [INpolicy={0..9999|NONE}]
   [OUtpolicy={0..9999|NONE}]
   [TUNnelpolicy={0..9999|NONE}]
```

Description This command associates one or more software QoS policies with a layer 1 or 2 interface or layer 3 tunnel.

Parameter	Description
INTerface	The interface or tunnel to associate the policy with. Valid entry types are: Layer 1 and 2 interfaces: ● eth (e.g. eth0) ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● PPP (e.g. ppp0) ● the switch instance (swi0) ● the switch instance on AR750S routers (swi0) Layer 3 tunnels: ● GRE (e.g. gre1) ● IPv6 6-to-4 virtual interface (e.g. virt9) ● the name of an IPSec policy (ipsec-*policyname*) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command.
INpolicy	The software QoS policy that the switch applies to ingress traffic on the interface. **Inpolicy** is only valid for layer 1 and 2 interfaces. Default: no default
	0..9999 The policy ID.
	NONE No policy. You can use this option to deactivate software QoS on ingress traffic for the interface.
OUtpolicy	The software QoS policy that the switch applies to egress traffic on the interface. **Outpolicy** is only valid for layer 1 and 2 interfaces. Default: no default
	0..9999 The policy ID.
	NONE No policy. You can use this option to deactivate software QoS on egress traffic for the interface.
TUNnelpolicy	The software QoS policy that the switch applies to tunnelled traffic on the interface. **Tunnelpolicy** is only valid for tunnels. The switch performs QoS processing on traffic before it enters the tunnel (in other words, before it is encapsulated by the tunnelling protocol). Default: no default
	0..9999 The policy ID.
	NONE No policy. You can use this option to deactivate software QoS on tunnelled traffic for the interface.

* The shortest string you can enter is shown in capital letters.

Example To apply policy 1 to egress traffic on ppp0 use the command:

```
set sqos int=ppp0 ou=1
```

To apply software QoS policy 2 to traffic that the IPsec policy *central* processes, use the command:

```
set sqos int=ipsec-central tun=2
```

Related Commands

create sqos policy
delete sqos policy trafficclass
destroy sqos policy
set sqos policy
show sqos interface
show sqos policy

set sqos meter

Syntax

```
SET SQOS METer=id-list [DESCription=description]
   [IGNorebwclass={Yes|No}]
   [MINbandwidth=rate[Kbps|Mbps|Gbps]]
   [MAXbandwidth=rate[Kbps|Mbps|Gbps]]
   [MINBUrstsize=burstsize[Bytes|Kbytes|Mbytes|Gbytes]]
   [MAXBUrstsize=burstsize[Bytes|Kbytes|Mbytes|Gbytes]]
   [TYPE={SRtcm|TRtcm}]
```

Description

This command modifies one or more Three Colour Marker meters, as described in RFC 2697, *A Single Rate Three Color Marker*, September 1999 and RFC 2698, *A Two Rate Three Color Marker*, September 1999. The meter measures how much bandwidth the packets in a traffic flow use, and how well the bandwidth use conforms with the bandwidth specifications for the traffic class that the flow belongs to. It assigns the packet to a bandwidth class depending on its conformance (Table 3-4 on page 3-21).

Parameter	Description	
METer	The ID number of the meter. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
DESCription	A description of the meter, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
IGNorebwclass	Whether the meter acknowledges any previous bandwidth class assigned to packets. A meter that acknowledges previous conformance is called *colour aware*. Default: **no** (the meter is colour aware)	
	Yes	The metering function is colour blind and ignores any bandwidth class previously assigned to packets. It sets the meter bandwidth class according to only the metered conformance level of the flow.
	No	The metering function is colour aware and uses any bandwidth class previously assigned to packets, as well as the metered conformance level, to set the bandwidth class. Packets previously labelled non-conformant (bandwidth class 3, red) will remain in bandwidth class 3. Packets previously labelled partially-conformant (bandwidth class 2, yellow) will be assigned to bandwidth class 2 or 3, depending on metered conformance.

Parameter	Description
MAXbandwidth	For the single rate meter of RFC 2697, the highest rate at which a steady stream of packets can arrive at the meter and be assigned to bandwidth class 1 (conformant, green). This is the Committed Information Rate (CIR) of the RFC. For the two rate meter of RFC 2698, the Peak Information Rate (PIR) of the RFC. See "Metering: Bandwidth conformance" on page 3-22 for a description of PIR. It must equal or exceed **minbandwidth**. *rate* is in the range 0 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Default: **1Mbps**
MAXBUrstsize	For the single rate meter of RFC 2697, the amount by which a packet can exceed **maxbandwidth** plus **minburstsize** and still possibly be assigned to bandwidth class 2. This is the Excess Burst Size (EBS) of the RFC. For the two rate meter of RFC 2698, the amount by which a packet can exceed **maxbandwidth** and still possibly be assigned to bandwidth class 2. This is the Peak Burst Size (PBS) of the RFC. *burstsize* is in the range 0 to 16777216 bytes (16 MB), specified in bytes, kbytes, Mbytes, or Gbytes (in upper or lower case). If you do not specify a unit, it uses bytes. If you specify kB, MB or GB, *burstsize* may contain a decimal fraction, for example, 1.25M. For a single rate meter, at least one of **minburstsize** and **maxburstsize** should equal or exceed the size of the largest IP packet you expect on the metered flow. For a two rate meter, both **minburstsize** and **maxburstsize** must exceed 0 (zero), and the RFC recommends that both values equal or exceed the size of the largest IP packet you expect on the metered flow. To create a single rate 2-colour meter (green and red), set this to 0 (zero). Default: **10kbytes**
MINbandwidth	For the two rate meter of RFC 2698, the Committed Information Rate (CIR) of the RFC. See "Metering: Bandwidth conformance" on page 3-22 for a description of CIR. It must not exceed **maxbandwidth**. *rate* is in the range 0 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Only valid if **type=trtcm.** Default: **1Mbps**

Parameter	Description	
MINBUrstsize	For the single rate meter of RFC 2697, the amount by which a packet can exceed **maxbandwidth** and still possibly be assigned to bandwidth class 1. This is the Committed Burst Size (CBS) of the RFC. For the two rate meter of RFC 2698, the amount by which a packet can exceed **minbandwidth** and still possibly be assigned to bandwidth class 1. This is the Committed Burst Size (CBS) of the RFC. *burstsize* is in the range 0 to 16777216 bytes (16 MB), specified in bytes, kbytes, Mbytes, or Gbytes (in upper or lower case). If you do not specify a unit, it uses bytes. If you specify kB, MB or GB, *burstsize* may contain a decimal fraction, for example, 1.25M. For a single rate meter, at least one of **minburstsize** and **maxburstsize** should equal or exceed the size of the largest IP packet you expect on the metered flow. For a two rate meter, both **minburstsize** and **maxburstsize** must exceed 0 (zero), and the RFC recommends that both values equal or exceed the size of the largest IP packet you expect on the metered flow. For a single rate meter, if you set **minburstsize** to 0 (zero) the meter will assign all packets to bandwidth class 2 or 3. Default: **10kbytes**	
TYPE	The type of meter. "Metering: Bandwidth conformance" on page 3-22 describes the meters. Default: **srtcm**	
	SRtcm	The Single Rate Three Colour Marker of RFC 2697.
	TRtcm	The Two Rate Three Colour Marker of RFC 2698.
* The shortest string you can enter is shown in capital letters.		

Example To make meter 0 a colour-blind meter, use the command:

```
set sqos met=0 ign=y
```

Related Commands

create sqos meter
create sqos trafficclass
destroy sqos meter
show sqos meter

set sqos policy

Syntax

```
SET SQOS POLIcy=id-list
   [BWClass3action={DROP|PAUSE|NONE}]
   [DEFaulttrafficclass={0..9999|NONE}]
   [DESCription=description] [DSCPMap={0..9999|NONE}]
   [IGNOREPrenatinfo={YES|NO}] [METer={0..9999|NONE}]
   [PAUSEAction={NONE|LOg|TRap|BOth}] [PAUSETIme={1..30}]
   [REMarking={0..63|USEDscpmap|NONE}]
   [REMARKVlanpri={0..7|NONE}] [SYSTEMTraffic={5..50}]
   [VIRTbw={bandwidth[Kbps|Mbps|Gbps]|NONE}]
   [WEIGhtscheduler={WRr|DWrr}]
```

Description This command modifies one or more QoS policies. A policy defines the overall QoS processing for an interface.

Parameter	Description	
POLIcy	The ID number of the policy. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
BWClass3action	The action the switch takes on Bandwidth Class 3 packets (red coloured packets). These are packets that exceed their allocated bandwidth, as determined at the metering stage. Default: **none**	
	DROP	The switch drops non-conformant packets.
	PAUSE	The switch drops non-conformant packets and stops dequeuing packets from the flow for **pausetime** seconds.
	NONE	The switch sends non-conformant packets to the next processing stage.
DEFaulttrafficclass	The traffic class that the switch applies to unclassified traffic on the policy's interface. It must be a leaf traffic class. Default: no default	
	0..9999	The traffic class ID.
	NONE	No user-nominated default traffic class. The switch uses the default traffic class that it made when you created the policy.
DEscription	A description of the policy, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
DScpmap	The DSCP map to assign to the policy. An integer in the range 0 to 9999. Default: **none**	
IGNOREPrenatinfo	Whether classifiers attached to the policy use pre-NAT IP settings for classification because these contain the distinguishing information. Default: **no** (uses pre-NAT settings)	

Parameter	Description	
MEter	The meter to assign to the policy. An integer in the range 0 to 9999. The meter determines a new bandwidth class (colour) for packets that are processed using this policy. You can configure the policy or a traffic class to drop or queue the packets on the basis of the new bandwidth class. Default: **none**	
PAUSEAction	The notification action taken by the switch when it pauses a non-conformant traffic flow that belongs to this policy. Only valid if **bwclass3action**=**pause**. Default: **none**	
	LOg	The switch generates a log message.
	TRap	The switch generates an SNMP trap.
	BOth	The switch generates both a log message and an SNMP trap.
	NONE	The switch does not generate a notification.
PAUSETIme	The length of time, in the range 1 to 30 seconds, for which the switch does not dequeue packets from a paused flow. Only valid if **bwclass3action**=**pause**. Default: **10**	
REMarking	How the switch sets the bandwidth class and/or DSCP value in the packet header's Differentiated Services field after metering. Default: **none**	
	0..63	The switch writes the specified value into the DSCP bits in the packet header.
	USEDscpmap	The switch uses the metered bandwidth class and current DSCP value, in conjunction with the policy's DSCP map, to determine the new DSCP value and/or bandwidth class. You must also specify the **dscpmap** parameter.
	NONE	The switch does not modify the DSCP value or metered bandwidth class.
REMARKVlanpri	What the switch sets the 802.1p VLAN priority field of the frame's Ethernet header to. Default: **none**	
	0..7	The switch writes the specified value into the 802.1p VLAN priority field of the Ethernet header.
	NONE	The switch does not modify the 802.1p VLAN priority field of the Ethernet header.
SYSTEMTraffic	The percentage of the interface's maximum bandwidth that the switch reserves for system traffic, in the range 5 to 50%. Default: **20**	

Parameter	Description	
VIRTbw	The maximum bandwidth available to the policy. **Virtbw** determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media. This rate is not equivalent to the transmission rate for data seen on the line, because the actual transmission rate includes the transmission of bits for the inter-frame-gap and the preamble of the layer 2 headers. For example, 10 Mbps of data leaving the internal queues is not equivalent to 10 Mbps of data transmitted on the line. *rate* is in the range 1 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Default: **none** (bandwidth is not limited)	
WEIGhtscheduler	The queue scheduling method for weighted traffic classes that belong to the policy. Weighted traffic classes assign weights to flows instead of priorities. Default: **wrr**	
	WRr	The switch uses a weighted round robin scheme to empty the queues of weighted traffic classes.
	DWrr	The switch uses a deficit weighted round robin scheme to empty the queues of weighted traffic classes. DWRR is less biased towards large packets than WRR.
* The shortest string you can enter is shown in capital letters.		

Example To modify policy 0 so that it allocates 15% of the available bandwidth to system traffic, use the command:

```
set sqos poli=0 systemt=15
```

Related Commands

add sqos policy trafficclass
create sqos policy
delete sqos policy trafficclass
destroy sqos policy
set sqos interface
show sqos policy

set sqos red

Syntax

```
SET SQOS RED=id-list
   [AVERaging=0..99] [DESCription=description]
   [START1=0..100] [STOP1=0..100] [DROP1=0..100]
   [START2=0..100] [STOP2=0..100] [DROP2=0..100]
   [START3=0..100] [STOP3=0..100] [DROP3=0..100]
```

Description

This command modifies one or more sets of RED curves. Red curve sets 0-2 exist by default, and cannot be modified or deleted. Table 3-6 on page 3-27 shows the properties of the default red curve sets.

Parameter	Description
RED	The ID number of the RED curve set. *id-list* is an integer in the range 3 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 3,4-9). An integer cannot appear in the list more than once.
AVERaging	A weight used in the moving averaging estimation of queue length for the RED curve algorithm. The estimated queue length is frequently updated, and is calculated by taking a weighted average of the previous average and the current instantaneous queue length. **Averaging** is the weight given to the previous average in this weighted calculation. If **averaging** is too high, the estimated average queue size responds too slowly to transient congestion. If **averaging** is too low, the estimated average queue size tracks the instantaneous queue size too closely and you lose the benefits of RED. RED works best when the estimated average queue length responds as slowly as possible while preventing the queue from becoming full. To achieve this, set **averaging** to a lower value if the queue constantly becomes full, so that the estimated average queue size more closely tracks the actual queue size. To check how often the queue becomes full, use the **trafficclass** parameter of the **show sqos counters** command on page 3-145 and check the queue counters, or set **qlimitexceedaction** and check the log messages or SNMP traps. Default: **98**
DESCription	An optional description of the RED curve set, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default
START1 START2 START3	The percentage of the queue length at which the RED algorithm starts to drop packets, for packets in bandwidth class 1, 2 and 3 respectively. A percentage in the range 1 to 100. Default: **35**

Parameter	Description
STOP1 STOP2 STOP3	The percentage of the queue length at which the RED algorithm is dropping **drop** percent of the packets, for packets in bandwidth class 1, 2 and 3 respectively. Beyond this point, 100% of the packets are dropped. This value must be larger than **start** and is a percentage in the range 1 to 100. Default: **65**
DROP1 DROP2 DROP3	The probability that a packet will be dropped at the **stop** queue length, for packets in bandwidth class 1, 2 and 3 respectively. A percentage in the range 1 to 100. Default: **30**

* The shortest string you can enter is shown in capital letters.

Example To set the drop probability to 20% for bandwidth class 1 packets in RED curve set 3, use the command:

```
set red=3 drop1=20
```

Related Commands

create sqos red
create sqos trafficclass
destroy sqos red
show sqos red

set sqos trafficclass

Syntax

```
SET SQOS TRafficclass=id-list
   [BWClass3action={DRop|PAuse|NONE}]
   [DESCription=description] [MAXQlen=1..1023]
   [METer={0..9999|NONE}]
   [PAUSEAction={NONE|LOg|TRap|BOth}] [PAUSETIme={1..30}]
   [PREMARKBwcl={1..3|USEDscpmap}]
   [PREMARKDscp={0..63|USEDscpmap|NONE}]
   [{PRIOrity=0..15|WEIght=0..100}]
   [QLIMITExceedaction={NONE|LOg|TRap|BOth}]
   [QUEUEDrop={Head|Tail}] [RED={0..9999|NONE}]
   [REMarking=0..63|USEDscpmap|NONE}]
   [REMARKVlanpri={0..7|NONE}]
   [VIRTbw={bandwidth[Kbps|Mbps|Gbps]|NONE}]
   [WEIGhtscheduler={WRr|DWrr}]
```

Description This command modifies one or more traffic classes. A traffic class specifies the QoS actions for a set of flows.

Parameter	Description	
TRafficclass	The ID number of the traffic class. *id-list* is an integer in the range 0 to 9999, a range of integers separated by a hyphen, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.	
DEscription	A description of the traffic class, which has no effect on its operation. A string 1 to 100 characters long. All printable characters are valid. If *description* contains spaces, it must be in double quotes. Default: no default	
BWClass3action	The action the switch takes on Bandwidth Class 3 packets (red coloured packets). These are packets that exceed their allocated bandwidth, as determined at the metering stage. Default: **none**	
	DRop	The switch drops non-conformant packets.
	PAuse	The switch drops non-conformant packets and stops dequeuing packets from the flow for **pausetime** seconds.
	NONE	The switch sends non-conformant packets to the next processing stage.
MAxqlen	The maximum combined queue length, in the range 1 to 1023 packets, for the traffic class. The switch drops packets that would exceed the maximum queue length. If you shape traffic by specifying a virtual bandwidth for a policy or traffic class (intermediate or leaf), give the appropriate leaf traffic classes large maximum queue lengths. This enables them to buffer bursts of packets and avoids packet loss. **maxqlen** is only valid on leaf traffic classes. Default: **64**	

Parameter	Description	
MEter	The meter to assign to the traffic class. An integer in the range 0 to 9999. The meter determines a new bandwidth class (colour) for packets that are processed using this traffic class. You can configure the traffic class, or the policy it is attached to, to drop or queue the packets on the basis of the new bandwidth class. Default: **none**	
PAUSEAction	The notification action taken by the switch when it pauses a non-conformant traffic flow that belongs to this traffic class. Only valid if **bwclass3action**=**pause**. Default: **none**	
	LOg	The switch generates a log message.
	TRap	The switch generates an SNMP trap.
	BOth	The switch generates both a log message and an SNMP trap.
	NONE	The switch does not generate a notification.
PAUSETIme	The length of time, in the range 0 to 30 seconds, for which the switch does not dequeue packets from a paused flow. If you specify 0, the switch will take the action in **pauseaction**, but not pause the flow. Only valid if **bwclass3action**=**pause**. Default: **10**	
QLIMITExceedaction	The notification action taken by the switch when a traffic flow exceeds the maximum queue length of the traffic class. Default: **none**	
	LOg	The switch generates a log message.
	TRap	The switch generates an SNMP trap.
	BOth	The switch generates both a log message and an SNMP trap.
	NONE	The switch does not generate a notification.
PREMARKBwcl	How the switch assigns the packet to a bandwidth class at the start of the QoS processing (before metering). The switch can use the assigned value in metering, marking and RED processing. You can only specify premarking in leaf traffic classes. Default: **1**	
	1..3	The switch assigns the packet to the specified bandwidth class.
	USEDscpmap	The switch uses the current DSCP value in conjunction with the policy's DSCP map to determine the bandwidth class. You must also specify the DSCP map by using the **dscpmap** parameter in the **create sqos policy** command on page 3-99 or the **set sqos policy** command on page 3-133.

Parameter	Description	
PREMARKDscp	How the switch changes the DSCP value in the packet header at the start of the QoS processing (before metering). The switch can use the assigned value in metering, marking and RED processing. You can only specify premarking in leaf traffic classes. Default: **none**	
	0..63	The switch writes the specified DSCP value into the packet header.
	USEDscpmap	The switch uses the current DSCP value in conjunction with the policy's DSCP map to determine the new DSCP. You must also specify the DSCP map by using the **dscpmap** parameter in the **create sqos policy** command on page 3-99 or the **set sqos policy** command on page 3-133.
	NONE	The switch does not change the packet DSCP value.
PRIOrity	The priority of the traffic class, an integer in the range 0 to 15. Specifying **priority** in traffic classes sets their policy (or intermediate traffic class) to schedule queues according to the relative priorities of all its traffic classes. The switch services the queue from the traffic class with the highest value for **priority** first. **Priority** and **weight** are mutually exclusive. Use the **priority** parameter to create a hierarchy based on the priority of flows, for strict priority queuing. Use the **weight** parameter to create a hierarchy with weighted flows, for WRR or DWRR queuing. If you create a mixed hierarchy the priority queues are emptied first, giving low latency queuing behaviour. Default: **1**	
QUEUEDrop	Whether packets are dropped from the head or tail of the queue when the queue becomes full. Tail dropping drops the newest packets; head dropping drops the oldest. Default: **tail**	
RED	The RED curve set that the switch uses for early dropping of packets. An integer in the range 0 to 9999. Default: **none**	
REMarking	How the switch sets the bandwidth class and/or the DSCP value in the packet header's Differentiated Services field after metering. Default: **none**	
	0..63	The switch writes the specified value into the DSCP bits in the packet header.
	USEDscpmap	The switch uses the metered bandwidth class and current DSCP value, in conjunction with the policy's DSCP map, to determine the new DSCP value and/or bandwidth class.You must also specify the **dscpmap** parameter in the **create sqos policy** command on page 3-99 or the **set sqos policy** command on page 3-133.
	NONE	The switch does not modify the DSCP value or metered bandwidth class.

Parameter	Description	
REMARKVlanpri	What the switch sets the 802.1p VLAN priority field of the frame's Ethernet header to. Default: **none**	
	0..7	The switch writes the specified value into the 802.1p VLAN priority field of the Ethernet header.
	NONE	The switch does not modify the 802.1p VLAN priority field of the Ethernet header.
VIRTbw	The maximum bandwidth available to the traffic class. **Virtbw** determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media. This rate is not equivalent to the transmission rate for data seen on the line, because the actual transmission rate includes the transmission of bits for the inter-frame-gap and the preamble of the layer 2 headers. For example, 10 Mbps of data leaving the internal queues is not equivalent to 10 Mbps of data transmitted on the line. *rate* is in the range 1 to 16000000 kilobits per second, specified in Kbps, Mbps or Gbps (in upper or lower case). If you do not specify a unit, it uses kbps. If you specify Mbps or Gbps, *rate* may contain a decimal fraction with up to 3 decimal places, for example, 1.25 Mbps. Default: **none** (bandwidth is not limited)	
WEIght	The weight given to the traffic class, in the range 0 to 100. Specifying **weight** in traffic classes sets their policy (or intermediate traffic class) to schedule queues according to the relative weights of all its traffic classes. If a traffic class has a weight of 0 (zero), the switch only empties its queue once the queues of all its sibling traffic classes are empty. **Priority** and **weight** are mutually exclusive. Use the **priority** parameter to create a hierarchy based on the priority of flows, for strict priority queuing. Use the **weight** parameter to create a hierarchy with weighted flows, for WRR or DWRR queuing. If you create a mixed hierarchy the priority queues are emptied first, giving low latency queuing behaviour. Default: no default, because the default behaviour is priority-based hierarchies.	
WEIGhtscheduler	The queue scheduling method that the switch uses to schedule this traffic class' weighted sub traffic classes. This parameter is only valid if the sub traffic classes specify the **weight** parameter. Default: **wrr**	
	WRr	The switch uses a weighted round robin scheme to empty the queues of weighted sub traffic classes.
	DWrr	The switch uses a deficit weighted round robin scheme to empty the queues of weighted sub traffic classes. DWRR is less biased towards large packets than WRR.

* The shortest string you can enter is shown in capital letters.

Example To modify traffic class 1 so that it has a moderately-high priority, and use DWRR to schedule the queues of the traffic class' sub classes, use the command:

```
set sqos tr=1 prio=10 weig=dw
```

Related Commands

add sqos policy trafficclass
add sqos trafficclass classifier
add sqos trafficclass dar
add sqos trafficclass subclass
create sqos trafficclass
delete sqos policy trafficclass
delete sqos trafficclass classifier
delete sqos trafficclass dar
delete sqos trafficclass subclass
destroy sqos trafficclass
show sqos trafficclass

show sqos

Syntax `SHow SQOS`

Description This command displays general information about software QoS.

Figure 3-31: Example output from the **show sqos** command

```
Software QoS Module
------------------------------------------------------------
  Status:
    SQoS Module Enabled.. YES
  Number of:
    Policies.............. 3
    Traffic Classes...... 7
    Meters................ 3
    RED Curves............ 3
    DSCP Maps............. 3
    Interfaces............ 2
    DAR Objects........... 3
  Debug Information:
    Debug Device......... 16
    Debug Flags.......... ERROR
```

Table 3-28: Parameters in the output of the **show sqos** command.

Parameter	Meaning
SQoS Module Enabled	Whether or not software QoS is enabled.
Number of:	The total number of each type of software QoS object.
Policies	The total number of software QoS policies.
Traffic Classes	The total number of software QoS traffic classes.
Meters	The total number of software QoS meters.
RED Curves	The total number of software QoS RED curve sets, including the 3 default RED curve sets.
DSCP Maps	The total number of software QoS DSCP maps, including the default map.
Interfaces	The total number of layer 1 and 2 interfaces and layer 3 tunnels that have software QoS policies and/or DAR objects attached.
DAR Objects	The total number of software QoS Dynamic Application Recognition objects.
Debug information:	Information about debugging settings, if debugging is enabled.
Debug Device	The device to which debug messages are sent.
Debug Flags	The types of debugging that are enabled.

Example To find out how many software QoS policies you have created, and other general information, use the command:

```
sh sqos
```

Related Commands

disable sqos
disable sqos debug
enable sqos
enable sqos debug
show sqos counters

show sqos counters

Syntax

```
SHow SQOS COUnters CLASSifier[=id-list]
   [DIrection={In|OUT|TUNnel|ALL}] [INTerface=interface]
   [TRafficclass=id-list]

SHow SQOS COUnters DAR[=id-list]
   [DIrection={In|OUT|TUNnel}] [INTerface=interface]

SHow SQOS COUnters POLIcy[=id-list]
   [DIrection={In|OUT|TUNnel}] [INTerface=interface]

SHow SQOS COUnters TRafficclass[={id-list|DEFault|SYStem}]
   [DIrection={In|OUT|TUNnel|ALL}] [INTerface=interface]
```

Description This command displays counter information for the specified software QoS objects.

Parameter	Description	
CLASSifier	The classifier to display the counters for. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. If the classifier is used in more than one traffic class, the command shows a set of counters for each use of the classifier, unless you limit the display to one traffic class. Default: no default (classifier counters are not displayed)	
DAR	The DAR object to display the counters for. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: no default (DAR object counters are not displayed)	
DIrection	A filter that restricts the command, so that the switch only displays counters for software QoS objects with this direction. Default: no default (not filtered by direction)	
	In	Display counters for software QoS objects that act on the packet at ingress.
	OUT	Display counters for software QoS objects that act on the packet at egress.
	TUNnel	Display counters for software QoS objects that act on tunnelled packets.
	ALL	Display counters for software QoS objects that act on packets in any direction. **All** is only valid with **trafficclass** and **classifier**.

Parameter	Description
INTerface	A filter that restricts the command, so that the switch only displays counters for software QoS objects that are associated with this interface or tunnel. Valid entry types are Layer 1 and 2 interfaces: ● eth (e.g. eth0) ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● PPP (e.g. ppp0) ● the switch instance (swi0) ● the switch instance on AR750S routers (swi0) Layer 3 tunnels: ● GRE (e.g. gre1) ● IPv6 6-to-4 virtual interface (e.g. virt9) ● the name of an IPSec policy (ipsec-*policyname*) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command. Default: no default (not filtered by interface)
POLIcy	The policy to display the counters for. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. If the policy is used on more than one interface, the command shows a set of counters for each use of the policy, unless you limit the display to one interface. Default: no default (policy counters are not displayed)
TRafficclass	The traffic class to display the counters for. If you specify both **classifier** and **trafficclass** the switch displays the classifier counters for that traffic class; otherwise it displays the traffic class counters. If the traffic class is used on a policy that is attached to more than one interface, the command shows a set of counters for each use of the traffic class, unless you limit the display to one interface. Default: no default
	id-list — An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once.
	DEFault — The default traffic class.
	SYStem — The system traffic class.
* The shortest string you can enter is shown in capital letters.	

Figure 3-32: Example output from the **show sqos counters classifiers** command

```
Classifier 8: (Interface=eth0 TC=6 Direction=Egress)
-------------------------------------------------------------
  Hit Counters
    Packets Matched.. 0
    Bytes Matched.... 0
```

Table 3-29: Parameters in the output of the **show sqos counters classifiers** command.

Parameter	Meaning
Classifier	The ID of the classifier that the counters apply to.
Interface	The interface that this set of classifier counters applies to.
Traffic Class	The traffic class that this set of classifier counters applies to.
Direction	Whether this set of classifier counters applies to ingress traffic (In), egress traffic (Out) or tunnelled traffic (Tunnel).
Hit Counters	Information about the classifier matches since the switch last restarted or the counters were last reset.
Packets Matched	The number of packets that were classified by the classifier.
Bytes Matched	The number of bytes of data that were classified by the classifier.

Figure 3-33: Example output from the **show sqos counters dar** command

```
DAR Object 1
-------------------------------------------------------------------------------
  Session Counters (by protocol)
    Total Sessions Recognised.. 3
    RTSP Sessions Recognised... 0
    SIP Sessions Recognised.... 3
    H323 Sessions Recognised... 0
  Session Counters (by media)
    Active Sessions............ 3
    Voice Sessions Started..... 1
    Video Sessions Started..... 1
  Dynamic Classifiers
    Classifer=10002 tc=2 ip=10.33.25.17/32 port=69-69
    Classifer=10000 tc=2 ip=10.33.25.17/32 port=49170-49171
    Classifer=10001 tc=2 ip=10.33.25.17/32 port=51372-51373
```

Table 3-30: Parameters in the output of the **show sqos counters dar** command.

Parameter	Meaning
DAR Object	The ID of the DAR object that the counters apply to.
Session Counters (by protocol)	Information about the number of sessions the DAR object recognised since the switch last restarted or the counters were last reset, sorted according to the protocol they used for session setup.
Total Sessions Recognised	The total number of sessions that were recognised by the DAR object.
RTSP Sessions Recognised	The number of Real Time Streaming Protocol sessions that were recognised by the DAR object.
SIP Sessions Recognised	The number of Session Initiation Protocol sessions that were recognised by the DAR object.
H323 Sessions Recognised	The number of H.323 sessions that were recognised by the DAR object.
Session Counters (by media)	Information about the number of sessions the DAR object recognised since the switch last restarted or the counters were last reset, sorted according to the type of data they carried.
Active Sessions	The number of sessions that are currently successfully connected.
Voice Sessions Started	The total number of successful VoIP connections.
Video Sessions Started	The total number of successful video connections.
Dynamic Classifiers	Information about the dynamic classifiers created by the DAR object.
Classifier	The ID number of the classifier. Within a traffic class, classifiers with a lower number take precedence.

Table 3-30: Parameters in the output of the **show sqos counters dar** command.

Parameter	Meaning
tc	The traffic class that the classifier is assigned to.
ip	The source or destination IP address and mask that the classifier uses to identify traffic.
port	The UDP or TCP port that the classifier uses to identify traffic.

Figure 3-34: Example output from the **show sqos counters policy** command

```
Policy 2: (Interface=eth0 Direction=Egress)
-------------------------------------------------------------------------------
  Packets Processed
    Passed (Packets)... 0                          (Bytes).. 0
    Dropped (Packets).. 0                          (Bytes).. 0
```

Table 3-31: Parameters in the output of the **show sqos counters policy** command.

Parameter	Meaning
Policy	The ID of the policy that the counters apply to.
Interface	The interface that this set of policy counters applies to.
Direction	Whether this set of policy counters applies to ingress traffic, egress traffic or tunnelled traffic.
Packets Processed	Information about the packets processed by the policy since the switch last restarted or the counters were last reset.
Packets Passed	The number of packets processed and forwarded by the policy.
Packets Dropped	The number of packets dropped by the policy for any reason.
Bytes Passed	The number of bytes of data processed and forwarded by the policy.
Bytes Dropped	The number of bytes of data dropped by the policy for any reason.

Figure 3-35: Example output from the **show sqos counters trafficclass** command

```
Traffic Class 4: (Interface=eth0 Direction=Egress)
-------------------------------------------------------------------------------
  Packets Processed
    Passed (Packets)..................... 0            (Bytes).. 0
    Dropped (Packets).................... 0            (Bytes).. 0
    Classifiers.......................... 2
  Queue Counters
    Current Queue Length (Packets)....... 0            (Bytes).. 0
    Avg Queue Length (Last Sec) (Packets)... 0         (Bytes).. 0
    Avg Queue Length (Last Min) (Packets)... 0         (Bytes).. 0
    Avg Queue Length (Last Hour) (Packets).. 0         (Bytes).. 0
    Avg Latency (microseconds)........... 0
  Meter Counters
    Meter................................ 1
    BWC 1 (Packets)...................... 0            (Bytes).. 0
    BWC 2 (Packets)...................... 0            (Bytes).. 0
    BWC 3 (Packets)...................... 0            (Bytes).. 0
  RED Curve Counters
    Red Curve............................ 0
    BWC 1 Dropped (Packets).............. 0            (Bytes).. 0
    BWC 2 Dropped (Packets).............. 0            (Bytes).. 0
    BWC 3 Dropped (Packets).............. 0            (Bytes).. 0
```

Table 3-32: Parameters in the output of the **show sqos counters trafficclass** command.

Parameter	Meaning
Traffic Class	The ID of the traffic class that the counters apply to.
Interface	The interface that this set of traffic class counters applies to.
Direction	Whether this set of traffic class counters applies to ingress traffic, egress traffic or tunnelled traffic.
Packets Processed	Information about the packets processed by the traffic class since the switch last restarted or the counters were last reset.
Packets Passed	The number of packets processed and forwarded by the traffic class.
Packets Dropped	The number of packets dropped by the traffic class for any reason.
Bytes Passed	The number of bytes of data processed and forwarded by the traffic class.
Bytes Dropped	The number of bytes of data dropped by the traffic class for any reason.
Classifiers	The classifiers attached to the traffic class.
Queue Counters	Information about the traffic class queue since the switch last restarted or the counters were last reset.
Current Queue Length	The number of packets currently queued by the traffic class.
Avg Queue Length (Last Sec)	The average number of packets and bytes queued by the traffic class at any one time, averaged over the last second.
Avg Queue Length (Last Min)	The average number of packets and bytes queued by the traffic class at any one time, averaged over the last minute.
Avg Queue Length (Last Hour)	The average number of packets and bytes queued by the traffic class at any one time, averaged over the last hour.
Avg Latency	The average length of time that packets spend in software QoS queues, in microseconds. Dropped packets are not counted. Latency is averaged since the switch last restarted or the counters were last reset.
Meter Counters	Information about the packets metered by the traffic class to measure their bandwidth use and conformance, since the switch last restarted or the counters were last reset.
Meter	The ID of the meter that the traffic class uses.
BWC 1 Packets	The number of packets that the meter assigned to bandwidth class 1 (green).
BWC 1 Bytes	The number of bytes of data that the meter assigned to bandwidth class 1 (green).
BWC 2 Packets	The number of packets that the meter assigned to bandwidth class 2 (yellow).
BWC 2 Bytes	The number of bytes of data that the meter assigned to bandwidth class 2 (yellow).
BWC 3 Packets	The number of packets that the meter assigned to bandwidth class 3 (red).
BWC 3 Bytes	The number of bytes of data that the meter assigned to bandwidth class 3 (red).
RED Curve Counters	Information about the packets dropped by the traffic class' RED curve, since the switch last restarted or the counters were last reset.
RED Curve	The ID of the RED curve that the traffic class uses.
BWC 1 Packets Dropped	The number of packets in bandwidth class 1 (green) that were dropped.
BWC 1 Bytes Dropped	The number of bytes of data in bandwidth class 1 (green) that were dropped.
BWC 2 Packets Dropped	The number of packets in bandwidth class 2 (yellow) that were dropped.
BWC 2 Bytes Dropped	The number of bytes of data in bandwidth class 2 (yellow) that were dropped.
BWC 3 Packets Dropped	The number of packets in bandwidth class 3 (red) that were dropped.
BWC 3 Bytes Dropped	The number of bytes of data in bandwidth class 3 (red) that were dropped.

Example To display the number of packets and bytes of data processed by policy 1 on ppp0, use the command:

```
sh sqos cou poli=1 int=ppp0
```

Related Commands

disable sqos debug
enable sqos debug
reset sqos counters
show sqos

show sqos dar

Syntax SHow SQOS DAR[={*id-list*|ALL}] [FULl|SUMmary]
[SHOwunused={Yes|No}]

Description This command displays information about one or more DAR objects.

Parameter	Description
DAR	The DAR object to display information about. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: **all**
FULl	Detailed information about the DAR object. This is the default if you specify a single DAR object.
SUMmary	A summary table of information about the DAR object. This is the default if you specify multiple DAR objects.
SHOwunused	Whether the output displays an entry for a parameter if the DAR object has no value for that parameter (for example, displays **Dst IP** when you have not specified a destination IP address in the DAR object). Default: **no**
* The shortest string you can enter is shown in capital letters.	

Figure 3-36: Summary example output from the **show sqos dar summary** command

```
Id     Src IP             Dst IP             Protocol   Codec   Interfaces
-------------------------------------------------------------------------------
   1   192.168.1.0/24     192.168.100.0/24        ALL     ANY   eth0
   2   192.168.2.0/24     192.168.200.0/24        ALL     ANY   eth1
```

Figure 3-37: Full example output from the **show sqos dar=1** command

```
Id..................... 1
    Src IP............. 2001:0DB8::1/32
    Protocol........... ALL
    Codec.............. ANY
    Inactivity Timeout.. 600
    SIP Port........... 5060
    RTSP Port.......... 554
    H323 Port.......... 1720
    Interfaces......... eth0
    Policies........... 1
    Traffic Classes..... 1
```

Table 3-33: Parameters in the output of the **show sqos dar** command.

Parameter	Meaning
ID	The ID number of the DAR object.
Description	The description of the DAR object, if it has one.
Src IP	The source IPv4 or IPv6 address, or subnet if it includes a CIDR mask or prefix length. The DAR object only matches traffic flows from that address or network.
Dst IP	The destination IPv4 or IPv6 address, or subnet if it includes a CIDR mask or prefix length. The DAR object only matches traffic flows destined for that address or network.
Protocol	The protocol that the DAR object uses to match packets. "All" indicates that the DAR object ignores the protocol.
Codec	The coder/decoder that the DAR object uses to match packets; one of "Any" (ignores the codec), "Audio" (matches traffic flows that use any audio codec), or "Video" (matches traffic flows that use any video codec).
Inactivity Timeout	A time in seconds. If a classified flow is idle for this length of time, its entry will be deleted. "None" indicates that the flow entry will never time out.
SIP Port	The UDP port that SIP messages are received on.
RTSP Port	The TCP port that RTSP session control messages are received on.
H323 Port	The TCP port that H.323 session control messages are received on.
Interfaces	The interfaces that the DAR object is attached to. The DAR object recognises session initiation packets on this interface and uses them to create classifiers for packets from those sessions.
Policies	The policies that the DAR object is attached to, through its traffic classes.
Traffic Classes	The traffic classes that the DAR object is attached to.

Example To display summary information about all DARs, use the command:

```
sh sqos dar
```

To display detailed information about all DARs, use the command:

```
sh sqos dar ful
```

Related Commands

add sqos interface dar
add sqos trafficclass dar
create sqos dar
delete sqos interface dar
delete sqos trafficclass dar
destroy sqos dar
set sqos dar

show sqos dscpmap

Syntax SHow SQOS DSCPMap[={*id-list*|ALL}] [FULl|SUMmary]

Description This command displays information about one or more DSCP maps.

Parameter	Description
DSCPMap	The DSCP map to display information about. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: **all**
FULl	Detailed information about the DSCP map. This is the default if you specify a single DSCP map.
SUMmary	A summary table of information about the DSCP map. This is the default if you specify multiple DSCP maps.
* The shortest string you can enter is shown in capital letters.	

Figure 3-38: Summary example output from the **show sqos dscpmap summary** command

```
Id
Map  Description                                              Policy Refs
-------------------------------------------------------------------------------
   1 Primary DSCP Map                                        1
   2 Secondary DSCP Map                                      2
   3 Auxilary DSCP Map
```

Figure 3-39: Full example output from the **show sqos dscpmap=1 full** command

```
                          Remarking
                           ------------------------------------------
               Premarking  BW Class 1     BW Class 2     BW Class 3
               ----------- ------------   ------------   ------------
Id     Old     New     New New     New    New     New    New     New
Map    DSCP    BWClass DSCP BWClass DSCP  BWClass DSCP  BWClass DSCP
-------------------------------------------------------------------------
   1
          1         1   63        1    1        2    1        3    1
          2         1   63        1    2        2    2        3    2
          3         1   63        1    3        2    3        3    3
          4         1   63        1    4        2    4        3    4
          5         1   63        1    5        2    5        3    5
          6         1   63        1    6        2    6        3    6
          7         1   63        1    7        2    7        3    7
          8         1   63        1    8        2    8        3    8
          9         1   63        1    9        2    9        3    9
         10         1   63        1   10        2   10        3   10
```

Table 3-34: Parameters in the output of the **show sqos dscpmap** command.

Parameter	Meaning
ID Map	The ID number of the DSCP map.
Description	The description of the DSCP map, if it has one.
Policy Refs	The policies that use this DSCP map.
Old DSCP	The packet's existing DSCP value. For premarking, this is the DSCP the packet had at ingress. For remarking, it is the DSCP the packet had at ingress unless premarking or earlier remarking changed it.
Premarking	The premarking table, which uses the ingress (old) DSCP to determine a bandwidth class and/or DSCP for the packet. Premarking happens before metering.
New BW Class	The bandwidth class to which the map assigns packets that have the given "Old DSCP" at ingress.
New DSCP	The DSCP which the map writes into packets that have the given "Old DSCP" at ingress.
Remarking	The remarking table, which uses the current (old) DSCP and bandwidth class to determine a new DSCP and/or bandwidth class for the packet. Remarking happens after metering, after the packet is dequeued from the leaf traffic class.
New BW Class	For each bandwidth class and old DSCP, the bandwidth class that the packet will be assigned to.
New DSCP	For each bandwidth class and old DSCP, the DSCP value that will be written into the packet header before egress.
BW Class 1 BW Class 2 BW Class 3	The current bandwidth class. Generally this is the bandwidth class that the meter assigned the packet to.

Example To display the information about DSCP map 1 that Figure 3-39 on page 3-153 illustrates, use the command:

```
show sqos dscpmap=1 full
```

This map is used to premark the DSCP value of all incoming packets to 63 and to assign them to bandwidth class 1.

Related Commands

create sqos dscpmap
create sqos trafficclass
destroy sqos dscpmap
set sqos dscpmap

show sqos interface

Syntax `SHow SQOS INTerface[={`*interface*`|ALL}]`

Description This command displays information about the software QoS policies attached to one or more layer 1 and 2 interfaces or layer 3 tunnels.

Parameter	Description
INTerface	The interface or tunnel to display information about. Valid entry types are: Layer 1 and 2 interfaces: ● eth (e.g. eth0) ● ATM channel (e.g. atm0.0) ● frame relay (e.g. fr0) ● PPP (e.g. ppp0) ● the switch instance (swi0) ● the switch instance on AR750S routers (swi0) Layer 3 tunnels: ● GRE (e.g. gre1) ● IPv6 6-to-4 virtual interface (e.g. virt9) ● the name of an IPSec policy (ipsec-*policyname*) To see a list of current valid Layer 1 and 2 interfaces, use the **show interface** command. Default: **all**

* The shortest string you can enter is shown in capital letters.

Figure 3-40: Example output from the **show sqos interface** command

```
Interface          In Policy     Out Policy    Tunnel Policy   DAR Objects
-------------------------------------------------------------------------
eth0                       1             21                        2
ppp0                       2             22                        3,5
ipsec-central                                          41    4
-------------------------------------------------------------------------
```

Table 3-35: Parameters in the output of the **show sqos interface** command.

Parameter	Meaning
Interface	The layer 1 or 2 interface, layer 3 tunnel or IPsec policy that the policies and DAR objects are acting on.
In Policy	The policy that the switch applies to ingress traffic on this layer 1 or 2 interface.
Out Policy	The policy that the switch applies to egress traffic on this layer 1 or 2 interface.
Tunnel Policy	The policy that the switch applies to traffic processed by this layer 3 tunnel or IPsec policy.
DAR Objects	The DAR objects that are attached to the interface. The DAR objects recognise session initiation packets on this interface and use them to create classifiers for packets from those sessions.

Example To find out which DAR object and policies are attached to each interface, use the command:

```
sh sqos int
```

Related Commands

create sqos policy
delete sqos policy trafficclass
destroy sqos policy
set sqos interface
set sqos policy
show sqos policy

show sqos meter

Syntax SHow SQOS METer[={*id-list*|ALL}] [FULl|SUMmary]

Description This command displays information about one or more meters. Meters measure bandwidth usage and conformance.

Parameter	Description
METer	The meter to display information about. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: **all**
FULl	Detailed information about the meter. This is the default if you specify a single meter.
SUMmary	A summary table of information about the meter. This is the default if you specify multiple meters.

* The shortest string you can enter is shown in capital letters.

Figure 3-41: Summary example output from the **show sqos meter summary** command

```
                Bandwidth       Burst Size
Id      Type    Min    Max      Min    Max      Traffic Classes
-------------------------------------------------------------------------
   1    SRTCM          1Mbps    10kB   10kB     4
   2    SRTCM          1Mbps    10kB   10kB
   3    TRTCM   1Mbps  1Mbps    10kB   10kB
```

Figure 3-42: Full example output from the **show sqos meter full** command

```
Id.................. 1
    Meter Type....... SRTCM
    Max Bandwidth.... 1Mbps
    Min Burst Size... 10kB
    Max Burst Size... 10kB
    Traffic Classes.. 4

Id.................. 2
    Meter Type....... SRTCM
    Max Bandwidth.... 1Mbps
    Min Burst Size... 10kB
    Max Burst Size... 10kB

Id.................. 3
    Meter Type....... TRTCM
    Min Bandwidth.... 1Mbps
    Max Bandwidth.... 1Mbps
    Min Burst Size... 10kB
    Max Burst Size... 10kB
```

Table 3-36: Parameters in the output of the **show sqos meter** command.

Parameter	Meaning
ID	The ID number of the meter.
Meter Type	Whether the meter is a Single Rate Three Colour Marker of RFC 2697 (SRTCM) or a Two Rate Three Colour Marker of RFC 2698 (TRTCM).
Description	The description of the meter, if it has one.
Min Bandwidth	For the two rate meter of RFC 2698, the Committed Information Rate (CIR) of the RFC. See "Metering: Bandwidth conformance" on page 3-22 for a description of CIR.
Max Bandwidth	For the single rate meter of RFC 2697, the highest rate at which a steady stream of packets can arrive at the meter and be assigned to bandwidth class 1 (conformant, green). This is the Committed Information Rate (CIR) of the RFC. For the two rate meter of RFC 2698, the Peak Information Rate (PIR) of the RFC. See "Metering: Bandwidth conformance" on page 3-22 for a description of PIR.
Min Burst Size	For the single rate meter of RFC 2697, the amount by which a packet can exceed **maxbandwidth** and still possibly be assigned to bandwidth class 1. This is the Committed Burst Size (CBS) of the RFC. For the two rate meter of RFC 2698, the amount by which a packet can exceed **minbandwidth** and still possibly be assigned to bandwidth class 1. This is the Committed Burst Size (CBS) of the RFC.
Max Burst Size	For the single rate meter of RFC 2697, the amount by which a packet can exceed **maxbandwidth** plus **minburstsize** and still possibly be assigned to bandwidth class 2. This is the Excess Burst Size (EBS) of the RFC. For the two rate meter of RFC 2698, the amount by which a packet can exceed **maxbandwidth** and still possibly be assigned to bandwidth class 2. This is the Peak Burst Size (PBS) of the RFC.
Traffic Classes	The traffic classes that use the meter.
Ignore Bandwidth Class	Whether the meter acknowledges any previous bandwidth class assigned to packets. "Yes" indicates that the metering function is colour blind and ignores any bandwidth class previously assigned to packets. It sets the meter bandwidth class according to only the metered conformance level of the flow. "No" indicates that the metering function is colour aware and uses any bandwidth class previously assigned to packets, as well as the metered conformance level, to set the bandwidth class.

Example To get a list of the available meters, use the command:

```
sh sqos met
```

Related Commands

create sqos trafficclass
create sqos meter
destroy sqos meter
set sqos meter

show sqos policy

Syntax `SHow SQOS POLIcy[={id-list|ALL}] [FULl|SUMmary|TREE] [SHOwunused={Yes|No}]`

Description This command displays information about one or more software QoS policies.

Parameter	Description
POLIcy	The policy to display information about. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: **all**
FULl	Detailed information about the policy. This is the default if you specify a single policy.
SUMmary	A summary table of information about the policy. This is the default if you specify multiple policies.
TREE	A tree diagram of the traffic class hierarchy that is attached to this policy.
SHOwunused	Whether the output displays an entry for a parameter if the policy has no value for that parameter (for example, displays **DSCP Map** when you have not specified a DSCP map for the policy). Default: **no**
* The shortest string you can enter is shown in capital letters.	

Figure 3-43: Summary example output from the **show sqos policy summary** command

```
              DSCP   Virt    Wt
Id     Mtr    Map    BW      Schd   Interfaces
-------------------------------------------------------------------------------
   1             1   -       WRR    eth0
   2             1   -       DWRR   eth1,eth2
```

Table 3-37: Parameters in the summary output of the **show sqos policy** command.

Parameter	Meaning
ID	The ID number of the policy.
Mtr	The meter that the policy uses.
DSCP map	The DSCP map that the policy uses.
Virt BW	The maximum bandwidth available to the policy. Virtual BW determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media.
Wt Schd	The queue scheduling method for weighted traffic classes that belong to the policy, one of Weighted Round Robin (WRR) or Deficit Weighted Round Robin (DWRR).
Interfaces	The layer 1 and 2 interfaces, layer 3 tunnels and IPsec policies that use this software QoS policy.

Figure 3-44: Full example output from the **show sqos policy full** command

```
Id.............................. 1
    Pause Time................... 10
    Maximum Queue Length (pkts).. 64
    Weight Scheduler............. WRR
    DSCP Map..................... 1
    System Traffic Weight........ 20
    Traffic Classes.............. 1 2 3
    DAR Objects.................. 1,2,3

Id.............................. 2
    Pause Time................... 10
    Maximum Queue Length (pkts).. 64
    Weight Scheduler............. WRR
    DSCP Map..................... 2
    Default Traffic Class........ 7
    System Traffic Weight........ 20
    Traffic Classes.............. 4 5 6 7*
    Interfaces (out)............. eth0,eth1

Id.............................. 3
    Pause Time................... 10
    Maximum Queue Length (pkts).. 64
    Weight Scheduler............. WRR
    System Traffic Weight........ 20
```

Table 3-38: Parameters in the full output of the **show sqos policy** command.

Parameter	Meaning
ID	The ID number of the policy.
Meter	The meter that the policy uses.
Description	The description of the policy, if it has one.
Pause Time	The length of time, in seconds, for which the switch does not dequeue packets from a paused flow.
Pause Action	The notification action taken by the switch when it pauses a non-conformant traffic flow that belongs to this policy, one of: Log: The switch generates a log message Trap: The switch generates an SNMP trap Both: The switch generates both a log message and an SNMP trap None: The switch does not generate a notification.
BW Class 3 Action	The action the switch takes on Bandwidth Class 3 packets (red coloured packets), one of: ● Drop: The switch drops non-conformant packets ● Pause: The switch drops non-conformant packets and stops dequeuing packets from the flow for Pause Time seconds. ● None: The switch sends non-conformant packets to the next processing stage.
Ignore Pre-NAT Information	Whether classifiers attached to the policy use pre-NAT IP settings for classification because these contain the distinguishing information, one of No (uses pre-NAT settings) or Yes (uses post-NAT settings).
Remark	How the switch sets the bandwidth class and/or DSCP value in the packet header's Differentiated Services field after metering.
Remark VLAN Priority	What the switch sets the 802.1p VLAN priority field of the frame's Ethernet header to. "None" indicates that the switch does not reset the VLAN priority.

Table 3-38: Parameters in the full output of the **show sqos policy** command. (Continued)

Parameter	Meaning
Virtual BW	The maximum bandwidth available to the policy. Virtual BW determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media.
Weight scheduler	The queue scheduling method for weighted traffic classes that belong to the policy, one of Weighted Round Robin (WRR) or Deficit Weighted Round Robin (DWRR).
DSCP Map	The DSCP map assigned to the policy.
System traffic weight	The percentage of the interface's maximum bandwidth that the switch reserves for system traffic, in the range 5 to 50%.
Traffic classes	The traffic classes that are attached to this policy. Parentheses show the hierarchy of sub traffic classes. If you have set a default class, it is indicated by an *.
Interfaces (in)	The layer 1 and 2 interfaces that use this policy for ingress traffic.
Interfaces (out)	The layer 1 and 2 interfaces that use this policy for egress traffic.
Interfaces (tunnel)	The layer 3 tunnels or IPsec policies that use this policy.
Default traffic class	The traffic class that the switch applies to unclassified traffic on the policy's interface. The class policies use by default is called "Default".

Figure 3-45: Tree example output from the **show sqos policy tree** command

```
                Actual                          Virt
Tree            Scheduler   Priority   Weight   BW      Classifiers
-------------------------------------------------------------------------------
1               PQ
  1             FIFO              15                   1
  2             FIFO              14
  3             FIFO              13

2               PQ
  4             FIFO              15                   2
  5             FIFO              14                   8
  7*            FIFO              1

3               FIFO
```

Table 3-39: Parameters in the tree output of the **show sqos policy** command.

Parameter	Meaning
Tree	The policy ID and its traffic class hierarchy. The first entry is the policy. The sub traffic classes belonging to a traffic class are shown indented below that traffic class (so in the example above, traffic classes 3 and 4 are attached to traffic class 2).
Actual Scheduler	The scheduling algorithm that the switch uses to schedule queues at this level of the hierarchy. For the top level of the tree, PQ indicates that only priority traffic classes are attached to the policy (apart from the system and default classes). WRR and DWRR indicate that only weighted classes are attached. PQ+WRR and PQ+WDRR indicate that a mix of priority and weighted traffic classes are attached.
Priority	For priority queue based traffic classes, the priority. The range is 1 to 15, and a higher value means a higher priority.
Weight	For weighted traffic classes, the weight. The range is 0 to 100, and a higher value means a higher weight.
Virt BW	The maximum bandwidth available to the policy. Virtual BW determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media.
Classifiers	The classifiers that each leaf traffic class uses.

Example To see the traffic class hierarchy that makes up policy 1 and information about the order in which it will dequeue packets, use the command:

```
sh sqos poli=1 tree
```

Related Commands

add sqos policy trafficclass
create sqos policy
delete sqos policy trafficclass
destroy sqos policy
set sqos policy

show sqos red

Syntax `SHow SQOS RED[={`*`id-list`*`|ALL}] [FULl|SUMmary] [SHOwunused={Yes|No}]`

Description This command displays information about one or more RED curve sets.

Parameter	Description
RED	The RED curve set to display information about. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: **all**
FULl	Detailed information about the RED curve set. This is the default if you specify a single curve set.
SUMmary	A summary table of information about the RED curve set. This is the default if you specify multiple curve sets.
SHOwunused	Whether or not the output displays an entry for a parameter if the RED curve set has no value for that parameter (for example, displays **Traffic Classes** when no traffic class uses the RED curve set). Default: **no**
* The shortest string you can enter is shown in capital letters.	

Figure 3-46: Summary example output from the **show sqos red summary** command

```
                 BW Class 1        BW Class 2        BW Class 3
                 ---------------   ---------------   ---------------
                 Start Stop Drop   Start Stop Drop   Start Stop Drop
Id     Average              Prob              Prob              Prob
-------------------------------------------------------------------
   0        98      35   50   20      20   35   30      10   20   40
   1        98      50   70   20      30   50   30      15   30   40
   2        98      80   95   20      60   80   30      40   60   40
```

Figure 3-47: Full example output from the **show sqos red=0,2 full** command

```
Id..................... 0
    Description......... Aggressive
    Averaging........... 98
    1 Start............. 35
      Stop.............. 50
      Drop Probability.. 20
    2 Start............. 20
      Stop.............. 35
      Drop Probability.. 30
    3 Start............. 10
      Stop.............. 20
      Drop Probability.. 40
    Traffic Classes..... 2

Id..................... 2
    Description......... Passive
    Averaging........... 98
    1 Start............. 80
      Stop.............. 95
      Drop Probability.. 20
    2 Start............. 60
      Stop.............. 80
      Drop Probability.. 30
    3 Start............. 40
      Stop.............. 60
      Drop Probability.. 40
    Traffic Classes..... 1,3
```

Table 3-40: Parameters in the output of the **show sqos red** command.

Parameter	Meaning
ID	The ID number of the RED curve set.
Description	The description of the RED curve set, if it has one.
Averaging	The weight used in the moving averaging estimation of queue length for the RED curve algorithm, expressed as a percentage of the current average. If Averaging is too high, the estimated average queue size responds too slowly to transient congestion. If Averaging is too low, the estimated average queue size tracks the instantaneous queue size too closely and you lose the benefits of RED.
For each of bandwidth class 1 (green), 2 (yellow), and 3 (red):	
Start	The percentage of the queue length at which the RED algorithm starts to drop packets.
Stop	The percentage of the queue length at which the RED algorithm is dropping Drop Probability percent of the packets. Beyond this point, 100% of the packets are dropped.
Drop Probability	The probability that a packet will be dropped at the Stop queue length.
Traffic Classes	The traffic classes that use this RED curve set

Example To see the cut-off values for each RED curve and their descriptions, use the command:

```
sh sqos red ful
```

Related Commands

create sqos red
create sqos trafficclass
destroy sqos red
set sqos red

show sqos trafficclass

Syntax `SHow SQOS TRafficclass[={id-list|ALL}] [FULl|SUMmary] [SHOwunused={Yes|No|ON|OFf}]`

Description This command displays information about one or more traffic classes.

Parameter	Description
TRafficclass	The traffic class to display information about. An integer in the range 0 to 9999, a range of integers separated by hyphens, or a comma separated list of integers and/or ranges (for example 0,3,4-9). An integer cannot appear in the list more than once. Default: **all**
FULl	Detailed information about the traffic class. This is the default if you specify a single traffic class.
SUMmary	A summary table of information about the traffic class. This is the default if you specify multiple traffic classes.
SHOwunused	Whether the output displays an entry for a parameter if the traffic class has no value for that parameter (for example, displays **Pause Mode** when you have not configured the traffic class to pause traffic). Default: **no**
* The shortest string you can enter is shown in capital letters.	

Figure 3-48: Summary example output from the **show sqos trafficclass summary** command

```
                                  Max
                Red       Virt    QLen      Wt
Id      Mtr      Curve     BW      (pkts)    Schd    Policy    Sub-classes
--------------------------------------------------------------------------------
   1       1         0     -           64    WRR          3         13,18
  13       2         1     -           64    WRR
  18       3         2     -           64    WRR
  23       4         3     -           64    WRR          6
```

Table 3-41: Parameters in the summary output of the **show sqos trafficclass** command.

Parameter	Meaning
ID	The ID number of the traffic class.
Mtr	The meter that the traffic class uses.
RED Curve	The RED curve that the traffic class uses.
Virt BW	The maximum bandwidth available to the traffic class. Virtual BW determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media.
Max Qlen (pkts)	The maximum queue length, in packets, for the traffic class. The switch drops packets that would exceed the maximum queue length.
Wt Schd	The queue scheduling method for weighted sub classes that belong to the traffic class, one of Weighted Round Robin (WRR) or Deficit Weighted Round Robin (DWRR).
Policy	The policy that uses the traffic class.
Sub-classes	The sub traffic classes that are attached to this traffic class.

Figure 3-49: Full example output from the **show sqos trafficclass=2** command

```
Id.............................. 2
    Description................. VoIP
    Pause Time.................. 10
    Pause Action................ None
    Premark BW Class............ 1
    Remark...................... USEDSCPMAP
    Remark VLAN Priority........ 6
    BW Class 3 Action........... Drop
    Virtual BW.................. None
    Maximum Queue Length (pkts).. 64
    Queue Limit Exceed Action.... None
    Weight Scheduler............ WRR
    Policy...................... 1
    Parent Class................ None
    Priority.................... 15
    Weight...................... None
    Sub Classes................. None
    Classifiers................. none
    DAR Objects.................1,2
```

Table 3-42: Parameters in the full output of the **show sqos trafficclass** command.

Parameter	Meaning
ID	The ID number of the traffic class.
Description	The description of the traffic class, if it has one.
Pause Time	The length of time, in seconds, for which the switch does not dequeue packets from a paused flow.
Pause Action	The notification action taken by the switch when it pauses a non-conformant traffic flow that belongs to this traffic class, one of: Log: The switch generates a log message Trap: The switch generates an SNMP trap Both: The switch generates both a log message and an SNMP trap None: The switch does not generate a notification.
Premark BW Class	How the switch assigns the packet to a bandwidth class at the start of the QoS processing (before metering).
Premark DSCP	How the switch changes the DSCP value in the packet header at the start of the QoS processing (before metering).
Remark	How the switch sets the bandwidth class and/or DSCP value in the packet header's Differentiated Services field after metering.
Remark VLAN Priority	What the switch sets the 802.1p VLAN priority field of the frame's Ethernet header to. "None" indicates that the switch does not reset the VLAN priority.
BW Class 3 Action	The action the switch takes on Bandwidth Class 3 packets (red coloured packets), one of: ● Drop: The switch drops non-conformant packets ● Pause: The switch drops non-conformant packets and stops dequeuing packets from the flow for Pause Time seconds. ● None: The switch sends non-conformant packets to the next processing stage.
Virtual BW	The maximum bandwidth available to the traffic class. Virtual BW determines the maximum rate at which data can leave the internal queues to be transmitted onto the physical media.
Max Queue Length (pkts)	The maximum queue length, in packets, for the traffic class. The switch drops packets that would exceed the maximum queue length.

Table 3-42: Parameters in the full output of the **show sqos trafficclass** command. (Continued)

Parameter	Meaning
Queue Limit Exceed Action	The notification action taken by the switch when a traffic flow exceeds the maximum queue length of the traffic class, one of: Log: The switch generates a log message Trap: The switch generates an SNMP trap Both: The switch generates both a log message and an SNMP trap None: The switch does not generate a notification.
Weight scheduler	The queue scheduling method for weighted sub classes that belong to the traffic class, one of Weighted Round Robin (WRR) or Deficit Weighted Round Robin (DWRR).
Policy	The policy that uses the traffic class.
Parent Class	The intermediate traffic class that this traffic class is attached to.
Priority	The priority of the traffic class, in the range 0 to 15. The policy (or intermediate traffic class) schedules queues of priority-based traffic classes according to the relative priorities of all its traffic classes. The switch services queues from the traffic class with the highest value for Priority first.
Weight	The weight given to the traffic class, in the range 0 to 100. The policy (or intermediate traffic class) schedules queues of weighted traffic classes according to the relative weights of all its traffic classes, using WRR or DWRR. If a traffic class has a weight of 0 (zero), the switch only empties its queue once the queues of all its sibling traffic classes are empty.
Sub Classes	The sub traffic classes that are attached to this traffic class.
Classifiers	The classifiers that are attached to this traffic class.
DARs	The DAR objects that are attached to this traffic class.

Related Commands

add sqos policy trafficclass
add sqos trafficclass classifier
add sqos trafficclass dar
add sqos trafficclass subclass
create sqos trafficclass
delete sqos policy trafficclass
delete sqos trafficclass classifier
delete sqos trafficclass dar
delete sqos trafficclass subclass
destroy sqos trafficclass
set sqos trafficclass

Chapter 4

Link Access Control Protocol (LACP)

Link Access Control Protocol (LACP) 4-2
Command Reference 4-3
- activate switch port 4-3
- add lacp port 4-4
- delete lacp port 4-6
- disable lacp 4-7
- disable lacp debug 4-7
- enable lacp 4-7
- enable lacp debug 4-8
- purge lacp 4-8
- reset lacp port counter 4-8
- set lacp port 4-9
- set lacp priority 4-10
- show lacp 4-11
- show lacp port 4-12
- show lacp trunk 4-14

Link Access Control Protocol (LACP)

The implementation of the Link Access Control Protocol (LACP) follows the IEEE Standard 802.3-2002, *CSMA/CD access method and physical layer specifications*.

On SwitchBlade switches, LACP runs on line cards that use the K1 silicon processor. Cards with this processor have a model number ending in v2. Alternatively you can see the silicon revision of each line card by executing the **show switch instance** command. The silicon revision is shown in the first line of this command's output

LACP operates where systems are connected over multiple communication links. In this configuration, links that LACP controls are constantly monitored and can be automatically added to or removed from trunk groups (or aggregated links).

Once LACP has been initially configured and enabled, it automatically creates trunk groups and assigns appropriate links to their membership. LACP continues to monitor these groups and dynamically adds or removes links to them as network changes occur.

LACP achieves this by determining the following:

- which ports are under LACP control
- whether each port is in *LACP active* or *LACP passive* mode
- which system has the highest LACP priority
- the LACP priority of ports
- whether the periodic timeout is fast or slow

Aggregation criteria

For individual links to be formed into an aggregated group they must meet the following criteria:

- originate on the same device
- terminate on the same device
- be members of the same VLANs
- have the same data rate
- share the same admin port key (assigned by using the **add lacp port adminkey** command)

The hardware must also be capable and have the capacity to handle the number of links to be aggregated.

Aggregated group identification

In order to identify particular aggregated groups, each group is assigned a link aggregation identifier called a *lag ID*. The lag ID comprises the following components for both the local system (called the Actor) followed by their equivalent components for the remote system (called the Partner):

- *system priority* - set by the **set lacp priority** command
- *system identifier* - the MAC address of the system
- *port key* - An identifier - created by the LACP software

- *port priority* - set by the **add lacp port priority** command
- *port number* - determined by the device connection

The lag ID can be displayed for each aggregated link by entering the **show lacp trunk** command.

Command Reference

This section describes the commands available to configure and manage LACP functions on the switch.

activate switch port

Syntax

```
ACTivate SWItch POrt={port-list|ALL} {AUTOnegotiate}
   {LOCK}
```

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports. On SwitchBlade switches, ports are identified either by the port number or the line card.port number.

Description

This command activates autonegotiation of port speed and duplex mode for a port or a group of ports.

The PORT parameter specifies the port for which autonegotiation is to be activated. Only ports in the list that are set to autonegotiate are affected by this command. Ports are not modified that have a fixed speed setting or that belong to a trunk group. A port added to LACP autonegotiates until it actively becomes part of an aggregated link (i.e. trunked), at which point it operates at the speed of the link.

A port that has been added to LACP autonegotiates until it actively becomes part of an aggregated link (i.e. trunked), when it then operates at the speed of the aggregated link.

The AUTONEGOTIATE parameter specifies that the port is to activate the autonegotiation process. The port begins to autonegotiate link speed and duplex mode.

The LOCK parameter manually locks the switch port before it reaches its learning limit so that no new addresses are automatically learned. The LEARN parameter for the port is set to the current number of learned MAC addresses.

Examples

To activate autonegotiation on ports 1-8 and port 10, use the command:

```
act swi po=1-8,10 auto
```

Related Commands

set lacp port
activate switch port

add lacp port

On SwitchBlade switches, LACP runs on line cards that use the K1 silicon processor. Cards with this processor have a model number ending in v2. Alternatively you can see the silicon revision of each line card by executing the **show switch instance** command. The silicon revision is shown in the first line of this command's output

Syntax

```
ADD LACP POrt=[{port-list|ALL}] [ADMinkey=key]
   [PRIOrity=priority] [MODe={ACTIve|PASsive}]
   [PERiodic={FASt|SLOw}]
```

where:

- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered port, including uplink ports. On SwitchBlade switches, ports are identified either by the port number or the line card.port number.
- *key* is an integer from 0 to 65535
- *priority* is an integer from 0 to 65535

Description

This command adds a port to LACP's control thus enabling LACP to put it into an aggregated link. By default, ports are added in the active mode. If a port is added in the active mode, and its link's requirements for trunking are met, then the port and it associated link are automatically aggregated without further configuration. The same situation applies for a port configured in passive mode but whose link connects to a remote port configured in active mode.

The **port** parameter specifies the ports whose parameters are to be modified. Where none of the ports specified are presently managed by LACP, the command takes effect if it can be applied to all the specified ports. Where some of the ports specified are already managed by LACP, and additional ports are added, by using the ALL parameter for example, then the LACP managed ports have their Key and other parameters changed and the command succeeds on all the specified ports.

In the following descriptions, references to an individual port refers to all ports selected by the **port** parameter.

The **adminkey** parameter specifies the Admin LACP port key. This affects the LACP port key that is generated but does not determine its value. You can use this parameter to prevent ports from being aggregated when they might otherwise form a trunk. By default all ports that can be aggregated are given the same LACP port key. The default for **adminkey** is 1.

The **priority** parameter specifies the *LACP port priority*. This priority assigned is used where the number of physical links connecting two devices is greater than the number that can be aggregated. The priority entered is then used to determine which ports are selected for aggregation. The default of 32,768 (0 being the highest priority) is applied to all ports.

Where the port priority is the same, the port number governs which ports are selected. Low port number have high priority. Excess ports are put into standby mode, which are effectively disabled but take the place of a link in a trunk that goes down.

The **mode** parameter specifies whether the port runs in LACP *passive* or *active* mode. A port in passive mode begins sending LACPDUs in response to a received LACPDU; whereas, a port in active mode always sends LACPDUs at regular intervals specified by the **periodic** parameter.

The **periodic** parameter specifies the requested rate that the LACP port receives LACPDU *update messages* from its partner port. A port in fast mode receives one LACPDU every second; in slow mode, a port receives one every thirty seconds.

Examples

To add ports 3 and 5 to LACP, use the command:

```
add lacp po=3,5
```

Related Commands

delete lacp port
set lacp port
show lacp port

delete lacp port

Syntax `DELete LACP POrt={`*`port-list`*`}`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered switch port, including uplink ports. On SwitchBlade switches, ports can be specified by either the port number or the line card.port number.

Description This command removes ports from LACP's control and LACP frames are no longer transmitted across the link. It is good practice to delete LACP from ports that are linked to non-LACP-capable devices.

The PORT parameter specifies switch ports to be deleted from LACP's control. Ports specified must be under the control of LACP. ALL is not a configurable option; to stop LACP on all ports, use the **disable lacp** command on page 4-7.

Examples To delete ports 3 and 5 from LACP, use the command:

```
del lacp po=3,5
```

Related Commands

add lacp port
disable lacp
set lacp port
show lacp port

disable lacp

Syntax `DISable LACP`

Description This command disables the LACP processes on the switch. A warning message, notification message, and log message are generated when this command is executed. LACP is disabled by default. Port settings that are changed while LACP is disabled take effect when LACP is re-enabled.

Related Commands **show lacp**

disable lacp debug

Syntax `DISable LACP DEBug=[MSG|PACket|STAte|TRAce|ALL]`

Description This command disables the LACP debugging process, which is enabled by default. The MSG option displays the decoded form of incoming and outgoing LACP packets. The PACKET option displays incoming and outgoing LACP packets in hex. The STATE option displays internal state machine changes. The TRACE option displays the function call tree.

Related Commands **enable lacp debug**
show lacp

enable lacp

Syntax `ENAble LACP`

Description This command enables LACP on the switch. A notification message and a log message file are generated when this command is executed. LACP is disabled by default.

On SwitchBlade switches, running this command enables LACP on the switch irrespective of its line card configuration, but the LACP operation itself runs only on line cards that use the K1 silicon processor. Cards with this processor have a model number ending in v2. Alternatively, you can see the silicon revision of each line card by executing the **show switch** command with the **instance** parameter. The silicon revision is shown in the first line of this command's output

Related Commands **disable lacp**
show lacp
reset lacp port counter

enable lacp debug

Syntax `ENAble LACP DEBug=[PACket|STAte|ALL]`

Description This command enables the LACP debugging facility, which is disabled by default. The PACKET option displays all incoming and outgoing LACP packets. The STATE option displays internal state machine changes. The ALL option displays both LACP packets and internal state machine changes.

Related Commands **disable lacp debug**
show lacp

purge lacp

Syntax `PURge LACP`

Description This command destroys all LACP configuration and restores the defaults to all the configurable parameters. The debug parameters for all ports are reset to their defaults. This command returns the LACP module to the status that existed when first powered on.

Related Commands **show lacp**

reset lacp port counter

Syntax `RESET LACP POrt[={`*`port-list`*`|ALL}] COUnter`

where *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports. On SwitchBlade switches, ports are identified either by the port number or the line card.port number.

Description This command resets all LACP counters for the specified switch ports.

Examples To reset the LACP counters for all ports, use the command:

```
reset lacp po cou
```

Related Commands **purge lacp**
show lacp

set lacp port

Syntax

```
SET LACP POrt=[{port-list|ALL}] [ADMinkey=key-number]
   [PRIOrity=priority] [MODe={ACTIve|PASsive}]
   [PERiodic={FASt|SLOw}]
```

where:

- *port-list* is a port number, range (specified as *n-m*), or comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports. On SwitchBlade switches, ports can be specified either by the port number or the line card.port number.
- *key-number* is a integer from 0 to 65535
- *priority* is a integer from 0 to 65535

Description

This command modifies the value of parameters for LACP ports.

The **port** parameter specifies the ports for which parameters are modified. If the command would succeed on a subset of the ports specified, but cause an error on the others, then the command as a whole fails and has no effect. Reference in the descriptions below to an individual port should be taken as a reference to all ports selected by the PORT parameter.

The **adminkey** parameter specifies the Admin LACP port key. This affects the LACP port key that is generated but does not determine its value. You can use this parameter to prevent ports from being aggregated when they might otherwise form a trunk. By default all ports that can be aggregated are given the same LACP port key. The default for **adminkey** is 1.

The **priority** parameter specifies the LACP port priority. This value is used to decide which ports should be selected when being added to a trunk group (where there are more links existing between the two devices than the switch is able to aggregate). The default is one. This means that port number governs which ports are selected (low port number equals high priority). Excess ports are put into a standby mode. In this mode they remain untrunked, but still able replace a link that goes down.

The **mode** parameter specifies whether the port runs in LACP passive or active mode. A port in passive mode sends an LACPDU in response to receiving one; whereas, a port in active mode sends LACPDUs at regular intervals as specified by the PERIODIC parameter.

The **periodic** parameter specifies the rate at which the LACP port transmits updates. A port in fast mode transmits one LACPDU every second; a port in slow mode transmits one LACPDU every thirty seconds.

Related Commands

delete lacp port
add lacp port
show lacp port

set lacp priority

Syntax `SET LACP PRIOrity=`*`priority`*

where *priority* is an integer from 0 to 65535

Description This command modifies the relative priority of LACP enabled partners.

The PRIORITY parameter specifies a numeric value that is used as part of the system priority calculation. When systems with multiple links connect and use LACP to control link aggregation, each system compares its system priority data identifiers to determine which system should control the links. A system identifier comprises a system priority component (configured by this parameter) followed the system's MAC address. Link control is assigned to the system with the numerically *lower* system priority data identifier. The default is 32768.

Examples System A is to connect to system B using LACP and System B is to control their aggregated links.

System A has a MAC address of 00-00-cd-00-0d-42 and has been assigned an LACP PRIORITY value of 500. System B has a MAC address of 00-00-cd-00-0d-52.

In order to ensure that System B controls the links, its LACP PRIORITY must be set to a value **lower** than 500. The LACP PRIORITY on System B is therefore set to 300. Note that system control is determined by the values set by the LACP Priority values because these have a greater numeric significance than MAC Addresses.

```
set lacp prio=300
```

Related Commands **show lacp**

show lacp

Syntax `SHow LACP`

Description This command displays the state of LACP on the switch.

Figure 4-1: Example output from the **enable switch lacp** command on a SwitchBlade switch

```
LACP Information
---------------------------------------------------------------
Status ......................... Enabled
   Actor System Priority ........... 80-00
   Actor System .................... 00-3e-0a-12-00-01
   LACP Ports ...................... 1.1,2.1-2.48,5.1,6.1
      Active ....................... 1.1,2.1-2.48
      Passive ...................... 5.1,6.1
```

Table 4-1: Parameters in output of the **show lacp** command

Parameter	Description
Status	Whether LACP is enabled.
Priority	User-configurable priority of the system. This parameter is concatenated with the Actor System parameter to generate the Actor System ID.
Actor System	MAC address of the local system.
LACP Ports	A list of ports currently under LACP control.
Active	A list of ports currently in LACP Active mode.
Passive	A list of ports currently in LACP Passive mode.

show lacp port

Syntax `SHow LACP POrt[={`*`port-list`*`|ALL}]`

where *port-list* is a port number, range (specified as *n-m*), or a comma-separated list of port numbers and/or ranges. Port numbers start at 1 and end at *m*, where *m* is the highest numbered Ethernet switch port, including uplink ports. On SwitchBlade switches, ports are identified either by the port number or the line card.port number.

Description This command displays LACP information about a specific switch port or all of them (Figure 4-2).

Figure 4-2: Example output from the **show lacp port** command

```
LACP Port Information
-------------------------------------------------------------------------------
Actor Port ....................... 1
  Trunk Group ..............  lacp1       Partner Information
  Selected ................. Selected     Partner System Priority ........ 8000
  Port Priority .................8000     Partner System      00-3e-0a-12-00-01
  LACP Port Number ............. 0001     Port Key ..........................4
  Port Key ... ..................6        Port Priority ................... 500
    Admin Key ................... 12      Port Number ..................... 0002
  Mode ...................... Active      Mode ......................... Active
  Periodic...................... Fast     Periodic........................ Fast
  Individual ..................... No     Individual ....................... No
  Synchronised ..................Yes      Synchronised .....................Yes
  Collecting ....................Yes      Collecting .......................Yes
  Distributing .................. Yes     Distributing ..................... Yes
  Defaulted ..................... No      Defaulted ........................ No
  Expired ....................... No      Expired .......................... No
  Actor Churn................. .....No
  Partner Churn................. ...No
-------------------------------------------------------------------------------
```

Table 4-2: Parameters in output of the **show lacp port** command

Parameter	Meaning
Port	Number of the port.
Trunk Group	Name of trunk group to which the port belongs. It is a name that LACP has automatically assigned to an aggregated link.You cannot manually create a trunk starting with the letters LACP. If LACP created, then the name has the prefix LACP followed by a numeric, such as LACP72. This number is the same as the new interface index shown by the **show interface** command.
Priority	User-configurable priority assigned to the port.
LACP Port Number	LACP encoded port number. On SwitchBlade switches, because dot-separated numbers cannot be transmitted (e.g. port 1.5), an alternative single number representation is used.
Port Key	Key that LACP has assigned to the port.
Admin Key	User-configurable key assigned to the port.

Table 4-2: Parameters in output of the **show lacp port** command (Continued)

Parameter	Meaning
Mode	The participation mode. If active, the port sends LACPDU packets regardless of the partner port's participation. If passive, the port sends LACPDU packets after receiving one from its partner port.
Periodic	User-configurable time period between transmission of periodic LACPDU packets; one of "Fast" (1 second) or "Slow" (30 seconds).
Individual	User-configurable setting that determines whether the port is an individual. If no, the port may be aggregated; if yes, it is not aggregated.
Synchronised	If yes, the port is considered to be in a synchronised state—the port has been correctly associated with an aggregator.
Collecting	Whether this port has been enabled to receive packets.
Distributing	Whether this port has been enabled to transmit packets.
Defaulted	Whether this system is using defaults for the partner information. If no, the values have been received from the partner via a LACPDU.
Expired	The port has not received a frame from its partner for 3 times the periodic time (3 or 90 seconds).
Actor Churn	Whether churning of the actor port has been detected.
Partner Churn	Whether churning of the partner port has been detected.
Partner Information	Information that has been received about the partner port. The partner port is the port on the connected device.
Partner System Priority	Partner's system priority.
Partner System	Partner's system identifier.
Port Key	Partner port's key.
Port Priority	Partner port's key priority.
Port Number	Partner port's port number.
Mode	Whether the mode is active or passive. If active, the partner port sends LACPDU packets regardless of this port's participation. If passive, the partner port sends LACPDU packets only after receiving one from this port.
Periodic	The setting of the partner port for the time period between transmission of periodic LACPDU packets; one of "Fast" (1 second) or "Slow" (30 seconds).
Individual	The setting of the partner port determining whether the port is an individual. If no, the partner port is not an individual and may be aggregated; if yes, it cannot be aggregated.
Synchronised	If yes, the partner system considers the partner port to be in a synchronised port—the port has been correctly associated with an aggregator; otherwise, no.
Collecting	Whether the partner port has been enabled for receiving packets.
Distributing	Whether the partner port has been enabled for transmitting packets.
Defaulted	Whether the partner system is using the defaults for this port's information. If no, the values have been received from this system via a LACPDU. If yes, the defaults are still in use.
Expired	When the partner port has not received a frame for 3 times the periodic time (3 or 90 seconds).

Examples To show the LACP port information for all ports, use the command:

```
sh lacp po
```

Related Commands add lacp port
set lacp port
show lacp

show lacp trunk

Syntax `SHow LACP TRunk`

Description This command displays the currently dynamically configured trunks for the LACP module.

Figure 4-3: Example output from the **show lacp trunk** command

```
LACP Dynamic Trunk Group Information
--------------------------------------------------------------------------------

Trunk group name .......... lacp53:
   Speed ................... 100 Mbps
   Ports in Trunk .......... 10,15
   LAG ID:
   [(8000,00-00-cd-03-00-79,0005,00,0000),(8000,00-00-cd-08-76-60,0002,00,0000)]
 --------------------------------------------------------------------------------
```

Related Commands show lacp trunk
show lacp

Chapter 5

Border Gateway Protocol version 4 (BGP-4)

Introduction 5-4
Overview of BGP-4 5-4
BGP Operation 5-6
    BGP Attributes 5-7
    BGP Route Selection 5-9
    Classless Inter-domain Routing (CIDR) and Aggregation 5-10
BGP Multi-Homing 5-11
BGP Route Filtering 5-13
    Route Maps 5-14
AS Confederations 5-16
Triggers 5-17
How to Configure BGP Peers 5-19
    How to Create a Basic BGP AS 5-19
    How to Create BGP Peers Using Peer Templates 5-23
    How to Modify BGP Peers (Without Templates) 5-24
    How to Use a Template to Modify BGP Peers 5-25
    How to Modify BGP Peers that Use a Template 5-26
    How to Delete BGP Peers 5-26
How to Filter Routes for BGP 5-27
    How to Configure AS Path Filters 5-27
    How to Configure Prefix Filters 5-29
    How to Configure Route Maps 5-30
How to Optimise BGP 5-36
    How to Minimise the Impact of Unstable EBGP Routes 5-36
    How to Withdraw Routes As Soon As they Fail 5-41
    How to Improve IBGP Scalability 5-42
    How to Handle Spikes in Memory Use 5-47
    How to Stop BGP from Overloading System Memory 5-48
    How to Avoid Leaking Private AS Numbers into Global BGP Tables 5-48
    How to Control Import of Static Routes 5-49
    How to Set the IP Address By Which the Switch Identifies Itself 5-50
Configuration Examples 5-51
    Example One 5-51
    Example Two 5-53
    Example Three 5-53
    Example Four 5-53
    Example Five 5-54
    Example Six 5-55
    Example Seven 5-55
    Example Eight 5-56
Command Reference 5-57

add bgp aggregate 5-57
add bgp confederationpeer 5-59
add bgp import 5-60
add bgp network 5-61
add bgp peer 5-62
add bgp peertemplate 5-70
add ip aspathlist 5-75
add ip communitylist 5-77
add ip prefixlist 5-79
add ip routemap 5-81
create bgp damping parameterset 5-86
delete bgp aggregate 5-88
delete bgp confederationpeer 5-89
delete bgp import 5-90
delete bgp network 5-91
delete bgp peer 5-92
delete bgp peertemplate 5-92
delete ip aspathlist 5-93
delete ip communitylist 5-94
delete ip prefixlist 5-94
delete ip routemap 5-95
destroy bgp damping parameterset 5-96
disable bgp autosoftupdate 5-96
disable bgp damping 5-97
disable bgp debug 5-98
disable bgp peer 5-99
enable bgp autosoftupdate 5-99
enable bgp damping 5-100
enable bgp debug 5-101
enable bgp peer 5-102
purge bgp damping 5-102
reset bgp damping 5-103
reset bgp peer 5-103
reset bgp peer soft 5-104
set bgp 5-105
set bgp aggregate 5-108
set bgp backoff 5-109
set bgp damping parameterset 5-110
set bgp import 5-112
set bgp memlimit 5-113
set bgp peer 5-114
set bgp peertemplate 5-121
set ip autonomous 5-126
set ip prefixlist 5-127
set ip routemap 5-129
show bgp 5-134
show bgp aggregate 5-136
show bgp confederation 5-137
show bgp backoff 5-138
show bgp counters 5-140
show bgp damping 5-144
show bgp damping routes 5-146
show bgp import 5-147
show bgp memlimit 5-148
show bgp memlimit scan 5-149
show bgp network 5-151
show bgp peer 5-152
show bgp peertemplate 5-157
show bgp route 5-159
show ip aspathlist 5-161

show ip communitylist 5-162
show ip prefixlist 5-163
show ip routemap 5-165

Introduction

This chapter describes the Border Gateway Protocol version 4 (BGP-4), how it is implemented on the switch, and how to configure the switch to use it.

BGP-4 is enabled with a special feature license that you can obtain by contacting an Allied Telesyn authorised distributor or reseller.

BGP-4 runs on hardware with 16 MB or more of RAM and is used with IPv4. See the Hardware Reference for your switch for memory details.

Overview of BGP-4

The Border Gateway Protocol version 4 (BGP-4) is an external gateway protocol. It allows two switches in different routing domains, known as *Autonomous Systems*, to exchange routing information to facilitate the forwarding of data across the borders of the routing domains. The basic operation of BGP-4 is described in RFC 1771, "A Border Gateway Protocol 4 (BGP-4)".

An Autonomous System (AS) is a set of switches under a single technical administration that uses:

- one or more internal gateway protocols (IGP)
- one or more sets of common metrics to route packets within its own AS
- an external gateway protocol (EGP) to route packets to other ASs

Every public AS is identified by an Autonomous System Number (ASN) in the range 1 to 64511. An ASN in this range is a globally unique number that the IANA assigns to every AS on the internet.

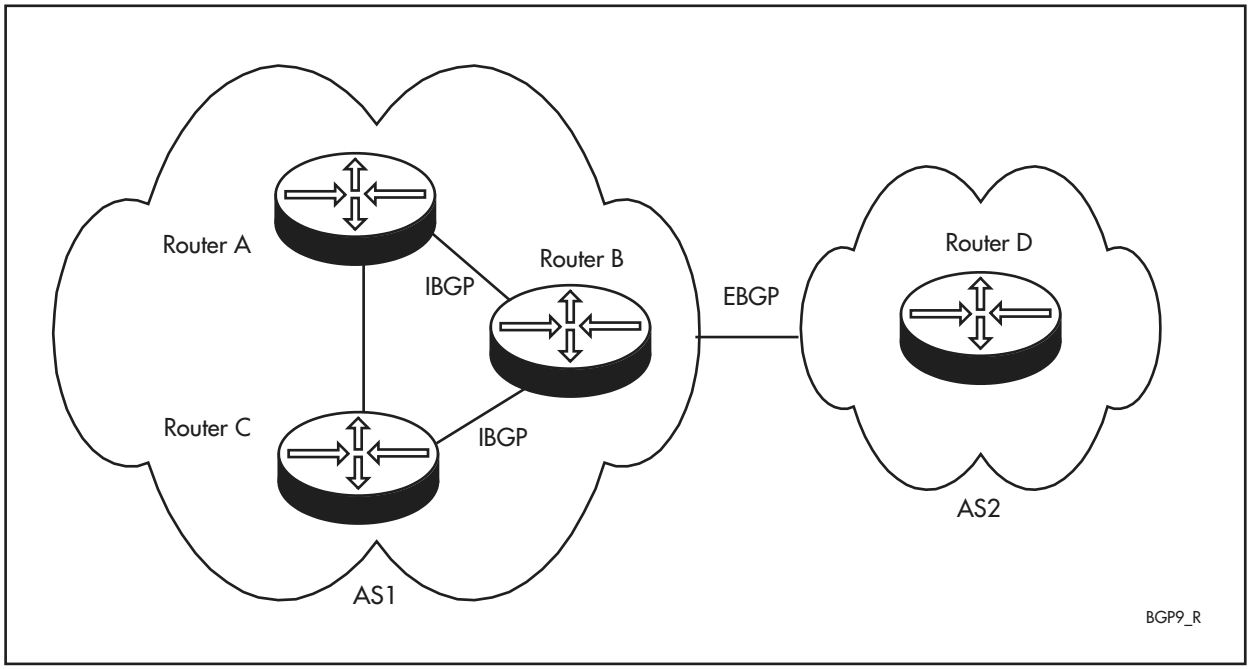
BGP lets switches learn multiple paths, choose the best path, and install it in the IP routing table. BGP-4 is based on distance vector (DV) protocol algorithms and uses TCP as its transport protocol on TCP port 179.

When BGP is used as an external gateway protocol to exchange routing information across AS borders, it is known as *External BGP (EBGP)*. EBGP connections are established between BGP *speakers* that have different AS numbers. A BGP speaker is any host that can use BGP to exchange routing information with another BGP-capable host. A BGP speaker does not necessarily have to be a switch – it could be a host that passes routes learned by BGP to a switch via another means, such as RIP.

A speaker may advertise any routes it knows to other speakers over an EBGP connection, as long as the speaker being advertised to has a different AS number from the speaker that is advertising.

The routes advertised over an EBGP connection can be learned by any means, for example IGP, EGP, or static assignment.

Figure 5-1: Example of the use of IBGP and EBGP



When BGP is used as an internal gateway protocol to transfer routing information within an AS, it is known as an *Internal BGP (IBGP)*. IBGP connections are established between BGP speakers that have the same AS number. Figure 5-1 on page 5-5 shows the use of IBGP and EBGP.

For speakers within an AS to learn the routes known by all the other speakers in the AS:

- Every speaker must have IBGP connections to all the other speakers in the AS (this is called *full mesh*), or
- Speakers need to be part of a confederation, or
- The AS must use route reflection. For information about route reflection, see "How to Improve IBGP Scalability" on page 5-42".

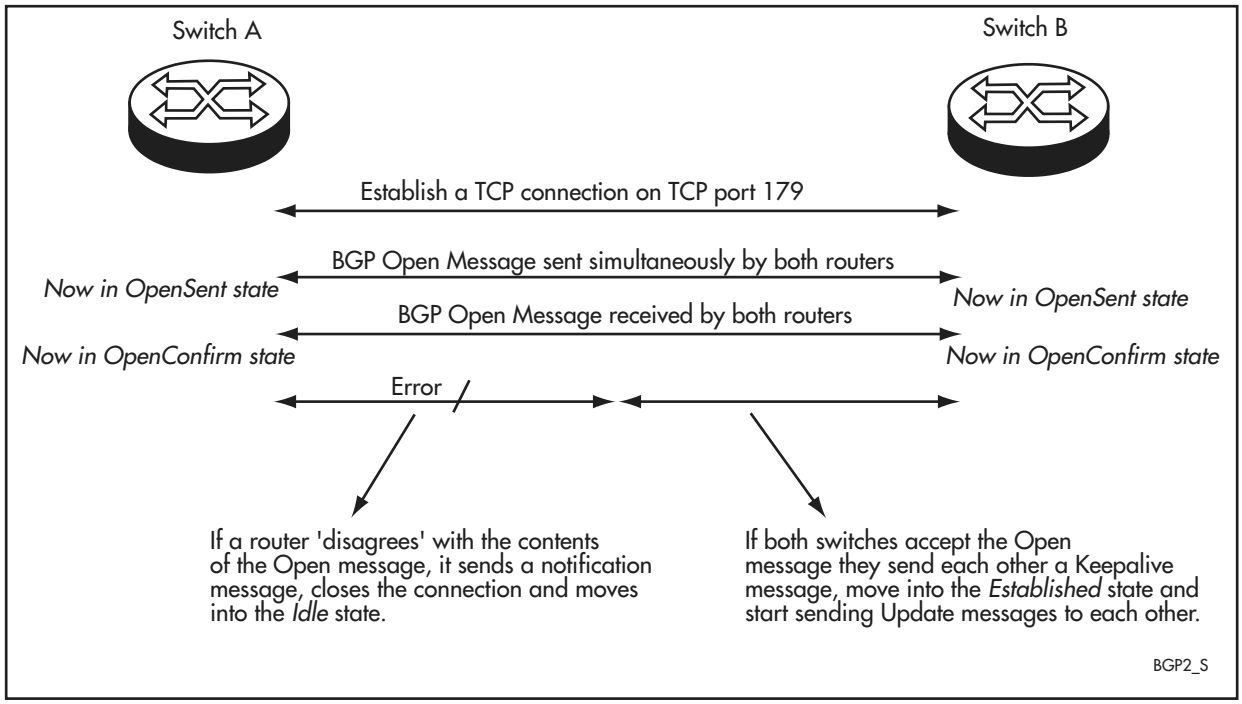
Without route reflection, a speaker cannot advertise routes learned over an IBGP connection to another speaker over another IBGP connection. However, a speaker can advertise routes learned from other means (for example, RIP and OSPF) to another speaker over an IBGP connection. A speaker can also advertise routes learned from an IBGP connection to another speaker over an EBGP connection.

An AS with one BGP speaker and a single external BGP connection is referred to as a *stub* AS.

BGP Operation

BGP is a protocol between two BGP speakers, which are called *peers*. Two switches become BGP peers when a TCP connection is established on Port 179 between them. The communication flow is illustrated in Figure 5-2.

Figure 5-2: Communication flow in a BGP session



Establishing a connection

BGP peer sessions start in the Idle state. In this state, BGP refuses all incoming BGP connections and does not allocate resources to peers. When you trigger a Start event—by enabling a peer—the switch initiates a TCP connection to the peer and moves that peer session into the Connect state.

If the TCP connection attempt to a peer fails, the session moves into the Active state, waits until its ConnectRetry time expires, then tries to establish the connection again.

When the TCP connection is established, BGP peers immediately identify themselves to each other by simultaneously sending *open* messages, and move into the OpenSent state. The open messages let the peers agree on various protocol parameters, such as timers, and negotiate shared capabilities.

When each switch receives an open message, it checks all the fields. If it "disagrees" with the contents of the open message, it sends a *notification* message, closes the connection and goes into the Idle state. If it finds no errors, it moves into the OpenConfirm state and sends back a *keepalive* message.

When both switches have received a keepalive message, they move into the Established state. The BGP session is now open. BGP sessions typically stay in the Established state most of the time. They only leave the Established state if an error occurs, or the hold time expires with no contact from the far end.

Exchanging routing information

Once a switch has established a BGP connection with a peer, it starts to exchange routing information with that peer. Initially the peers exchange a complete copy of their routing tables. After this, they exchange updates to this routing information, in update messages.

The routing information contained within an update message consists of:

- a set of attribute values

  Attributes describe properties of the routes the update message contains. They are described in detail in "BGP Attributes" below.

- a list of one or more prefixes

  A prefix is the network address and CIDR mask. Each prefix contained within an update message represents a network that can be reached through the IP address given in the NextHop attribute contained in the same update message.

  The attribute values contained in the attributes section of the update message apply to *all* the prefixes that are advertised in that update message. Update messages can advertise multiple routes by listing multiple prefixes, as long as all their attributes are the same.

Update messages can also list routes that are withdrawn from service. These routes do not have to have the same attributes as the advertised routes.

Maintaining and closing a connection

Peers regularly exchange keepalive messages to prevent sessions from expiring. These messages are sent every 1 to 21845 seconds, every 30 seconds by default.

When an error occurs during a BGP session, the switch that perceives the error sends a notification message to its peer identifying the error. It then closes the TCP connection and moves into the Idle state. Each switch stops using routing information it heard from the other.

If a BGP speaker receives neither an update message or a keepalive message from a peer for a configurable period of time, called the hold time, it resets the session and withdraws any routes it learned from that peer.

BGP Attributes

An important part of the BGP protocol operation is the set of *attributes* associated with prefixes. Each BGP update message contains a set of attributes (Table 5-1 on page 5-8). These attributes describe some of the properties of the routes, and can be used in making decisions about which routes to accept and which to reject, in the following ways:

- If the switch has multiple routes to a destination, it checks the attributes of each route to determine which one to use (see "BGP Route Selection" on page 5-9).
- You can create filters to reject or accept routes on the basis of their attributes (see "How to Filter Routes for BGP" on page 5-27).
- You can create route maps to change the attributes of particular update messages (see "How to Configure Route Maps" on page 5-30).

Table 5-1: BGP attributes

Attribute	Description
Origin	How the prefix came to be routed by BGP at the origin AS. The switch can learn prefixes from various sources and then put them into BGP. Sources include directly connected interfaces, manually configured static routes, and dynamic internal or external routing protocols. Values are IGP (internal protocols such as RIP and OSPF), EGP (other EGPs) and INCOMPLETE (static routes or other means). Every update message has this attribute.
AS_path	A list of the autonomous systems through which the announcement for the prefix has passed. As prefixes pass between autonomous systems, each autonomous system adds its Autonomous System Number (ASN) to the beginning of the list. This means the AS_path can be used to make routing decisions. Every update message has this attribute, although it may be empty.
Next_hop	The address of the next node to which the switch should send packets to get the packets closer to the destination. Every update message has this attribute.
Multi_Exit_Discriminator (MED)	A metric expressing the optimal path by which to reach a particular prefix in or behind a particular AS. One AS sets the value and a different AS uses that value when deciding which path to choose.
Local_preference	A metric used in IBGP so each host knows which path inside the AS it should use to reach the advertised prefix. EBGP peers do not send this value, and ignore it on receipt.
Atomic_aggregate	An attribute that allows BGP peers to inform each other about decisions they have made about overlapping routes. If Switch A receives overlapping routes, and selects the less specific (more general route) only, then it attaches the atomic_aggregate attribute. When one of its neighbours receives a prefix with the atomic_aggregate attribute set, that neighbour must not take the prefix and de-aggregate it into any more specific entries in BGP.
Aggregator	An attribute that can be attached to an aggregated prefix to specify the AS and IP address of the switch that performed the aggregation.
Community	Where the prefix is relevant to and should be advertised to. By default, all prefixes belong to the Internet community, which is the community of all BGP peers. Other communities have been globally defined that limit the scope of prefix advertisement or export, or you can identify a community by a community number.
Originator_ID	The router ID of the IBGP peer that first learned this route, either via an EBGP peer or by some other means such as importing it. This attribute is used by route reflection to prevent routing loops. EBGP peers do not send this value, and ignore it on receipt.
Cluster_list	A list of the cluster IDs of route reflectors who have reflected the corresponding route(s) within this AS. This attribute is used by route reflection to prevent routing loops. EBGP peers do not send this value, and ignore it on receipt.

BGP Route Selection

The route selection process involves selecting paths toward any prefix that should be put into a forwarding table. All routes that have been learned and accepted by the local system can be selected.

When there is only one route toward a particular prefix, it is the one used. When there are multiple routes for a particular prefix, then the system uses the following rules to decide which one to use. If a rule results in selection of a single route, the switch uses that route. If multiple routes still match, the switch goes to the next rule.

1. Selecting on local_preference

   The switch chooses the route with the highest local preference. How the switch determines the local preference depends on the source of the route:

   - for routes the switch learned via an EBGP session, or for routes it learned from sources such as an IGP or static configuration, the switch calculates the value of the preference itself.
   - for routes the switch learned from an IBGP peer, the switch uses the preference supplied by the peer—the update message for that route contains a local_preference attribute indicating the degree of preference.

2. Selecting on AS_path

   The switch chooses the route with the shortest AS path.

3. Selecting on the Multi_Exit_Discriminator value

   If the local system is configured to take into account the value of the Multi_Exit_Discriminator (MED), and if the multiple routes are learned from the same neighbouring AS, the switch chooses the route with the lowest MED value.

4. Selecting on the next_hop attribute

   The switch chooses the route that has the minimum cost to the next hop specified in the next_hop attribute. Deciding the cost involves looking into the IP route table.

5. Selecting on routes learned via EBGP

   If the switch learned only one of the possible routes via EBGP, it chooses that route. If the switch learned more than one of the possible routes via EBGP, it chooses the route with the lowest router ID.

6. Selecting on routes learned via IBGP

   When the switch learned all the possible routes via IBGP, it chooses the route that it learned from the IBGP neighbour with the lowest router ID.

Classless Inter-domain Routing (CIDR) and Aggregation

Interfaces, static routes, and routes learned via IGP (such as RIP and OSPF) can have a specific classful subnet mask (Class A, B, C or D). However, BGP routes are *classless*, (whereby IP addresses are not part of a class) so that shorter route tables can be exchanged and the number of advertised routes (prefixes) is less.

Figure 5-3 shows an example of inter-domain routing without CIDR. The service provider has many customers, all with Class C addresses that start with 204.71. The service provider announces each of the networks individually into the global Internet routing mesh.

Figure 5-3: Inter-domain routing without CIDR

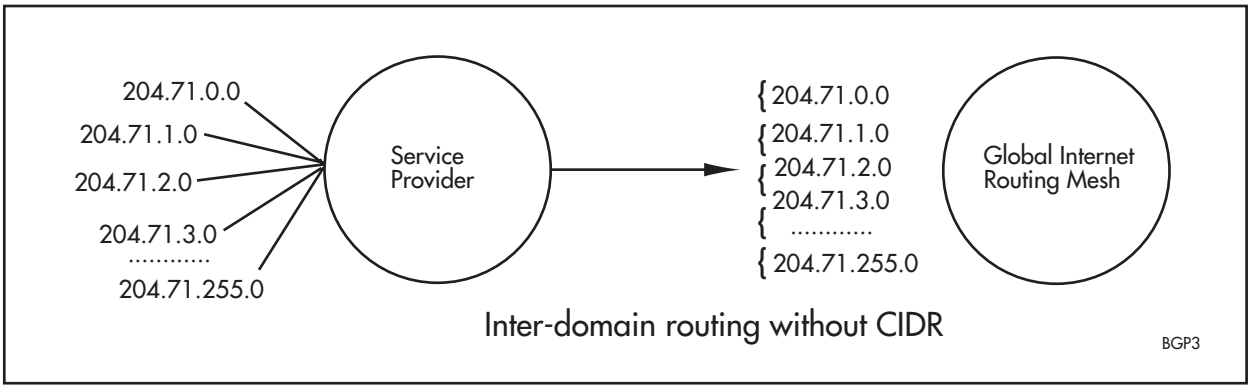
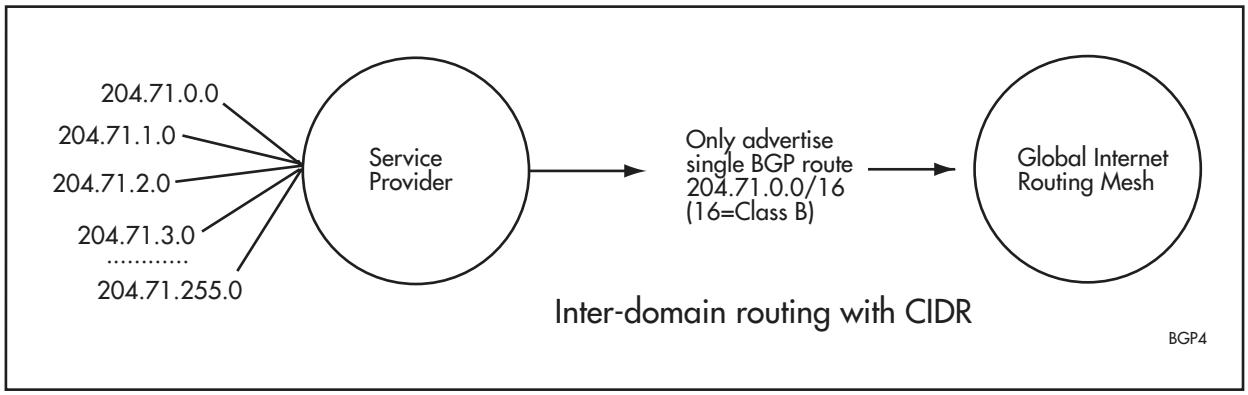


Figure 5-4 on page 5-10 shows an example of inter-domain routing with CIDR. By routing with CIDR, the service provider can aggregate the classful networks used by its customers into single classless advertisements. This provides routing for hundreds of customers by announcing only one advertisement into the global Internet routing mesh.

Figure 5-4: Inter-domain routing with CIDR



CIDR also copes with overlapping routes/prefixes because the route with the longest match (greatest number of bits/more specific netmask) is always chosen first. As the prefix traverses Autonomous Systems, each AS adds its AS number to the AS path bits, in both the BGP attribute and also the source IP and next-hop address.

BGP Multi-Homing

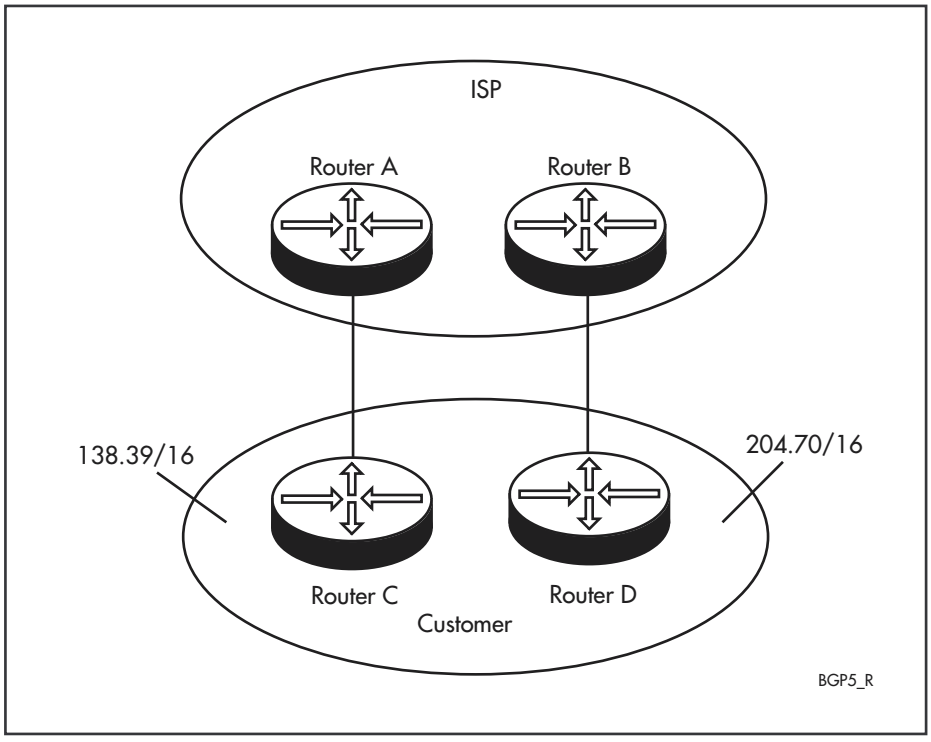
An AS may have multiple EBGP speakers connected to different ASs. This is known as BGP *multi-homing*. Multi-homing is used to improve reliability of the AS connection to the Internet, and to improve network performance owing to the fact that the network's bandwidth is the sum of all the circuits' bandwidth. Network performance increases when more than one connection is used at a time, otherwise, maximum performance is the bandwidth being used by one connection at a given time. An even split of traffic across multiple connections is called *load balancing*.

Sites can be multi-homed in the following ways:

- to a single Internet Service Provider (ISP) or Network Service Provider (NSP)
- to more than one ISP or NSP

The following figure illustrates the most reliable multi-homing topology to a single ISP involving different switches in the ISP and different switches in the customer network.

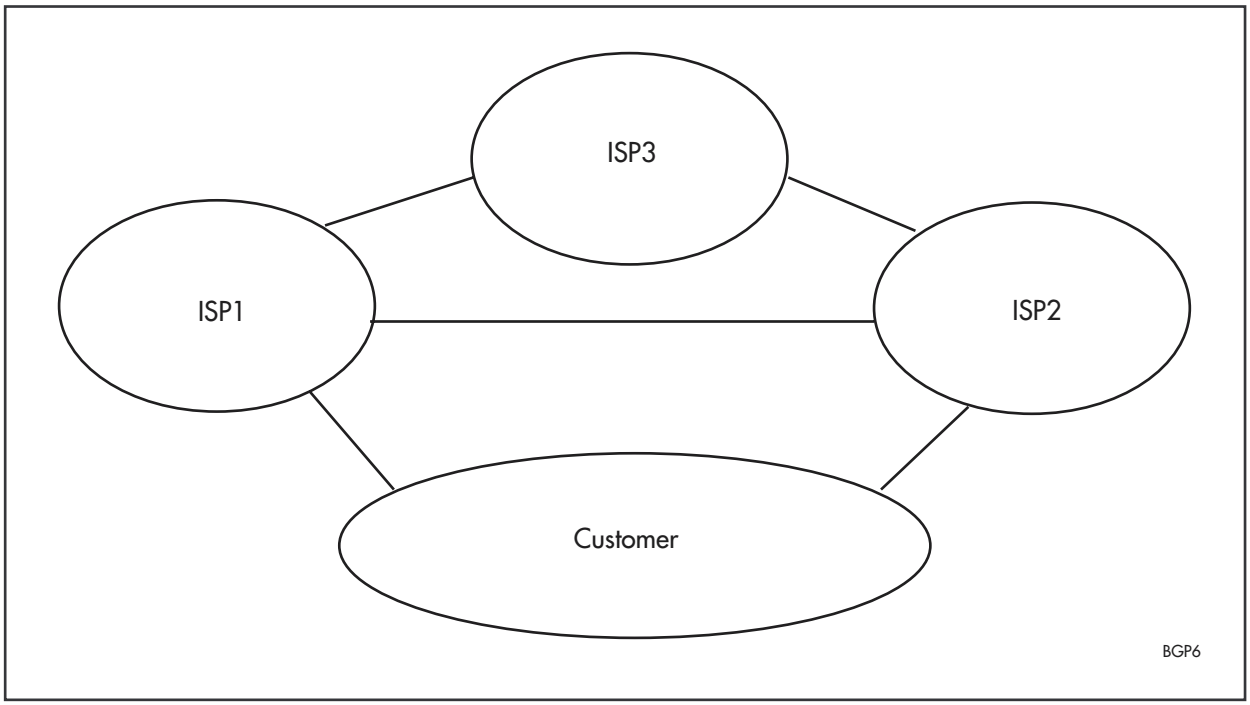
Figure 5-5: Multi-homing to a single ISP



This example is the most reliable because no equipment is shared between the two links. If the traffic between the two networks is equal, the approach to load balancing would be to use the link between Switch A and Switch C for traffic going to 138.39/16 and use the link between Switch B and Switch D for traffic going to 204.70/16.

Multi-homing to more than one provider is shown in Figure 5-6. The customer is multi-homed to ISP1 and ISP2; ISP1, ISP2, and ISP3 connect to each other. The customer has to decide how to use address space, as this is critical for load balancing from the ISPs to the customer, whether it delegates it by ISP1, ISP2, both, or independently.

Figure 5-6: Multi-homing to more than one provider



When the customer uses the address space delegated to it by ISP1, the customer uses a more specific prefix out of ISP1's aggregate and ISP1 can announce only the aggregate to ISP2. When the customer gets as much traffic from ISP1 as it gets from both ISP2 and ISP3, load balancing can be good. When ISP2 and ISP3 together send substantially more traffic than ISP1, load balancing can be poor. When the customer uses the address space delegated to it by ISP2, it does the same although ISP1 is the ISP to announce the more-specific route and attract traffic to it. Load balancing may be quite good if ISP1's address space is used, but not very good if ISP2's space is used.

When the customer uses the address space delegated to it by both ISP1 and ISP2, the degree of load balancing from ISP1 and ISP2 to the customer depends on the amount of traffic destined for the two ISPs. If the amount of traffic for the two is about the same, load balancing towards the customer can be quite good, if not, load balancing can be poor.

The option of the customer getting its own address space from a registry rather than from either ISP1 or ISP2 provides the most control but there is no aggregation. *Aggregation* is the combining of several different routes so that a single route can be advertised. This minimises the size of the routing table. The customer needs address space that makes it through the *route filters* (see "BGP Route Filtering"); otherwise, the customer may end up having no connectivity. If the customer gets address space that makes it through the route filters, there must be control over which provider uses which path to reach the customer.

When ISP1 is the largest, it may want to reach the customer via the ISP1-customer link, but have ISP2 and ISP3 go through the ISP2-customer link. When the path learned from ISP2 is shorter than the path learned from ISP1, ISP3 uses ISP2 to reach the customer.

BGP Route Filtering

BGP route filtering enables network providers to control their routing tables and meet the terms of business relationships they have with the networks they are connected to.

As a provider, you can filter the routing information that your switches receive from the networks they connect to, and that they advertise to those networks. This gives you control over the path of any traffic originating from or traversing your network.

Usually, one or more of your BGP switches form peer relationships with BGP switches at other ISPs with which you have entered into data transporting agreements. The process of BGP filtering is, in effect, the process of specifying the routes that your switches send or receive from each of their peers.

There are three filter types that you can apply to the BGP update messages that your switch exchanges with a particular BGP peer:

- AS path filters

  Path filters look at the AS_path attribute in update messages. If the attribute in the update message matches the filter criteria then the whole update message is filtered out (or accepted, depending on what action the filter entry has been configured to carry out).

- prefix filters

  Prefix filters look at the individual prefixes within an update message, and compare them against an IP routing filter. If a prefix within the update message matches the filter criteria then that individual prefix is filtered out (or accepted).

- route maps

  Route maps have a complex combination of match criteria and actions. When applied to a BGP peer, they can:

  - accept or reject update messages on the basis of origin, community, AS path or MED
  - accept or reject particular routes, by comparing the update message's routes with a prefix list
  - alter the attribute values in matching update messages.

You can use filters in both the incoming and outgoing directions. In the incoming direction, they filter the update packets that the switch receives from the peer. In the outgoing direction, they filter the update packets that the switch sends to the peer.

If you create more than one type of filter, the switch first applies the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update or prefix, so an update or prefix is only included if all the applied filters result in it being included.

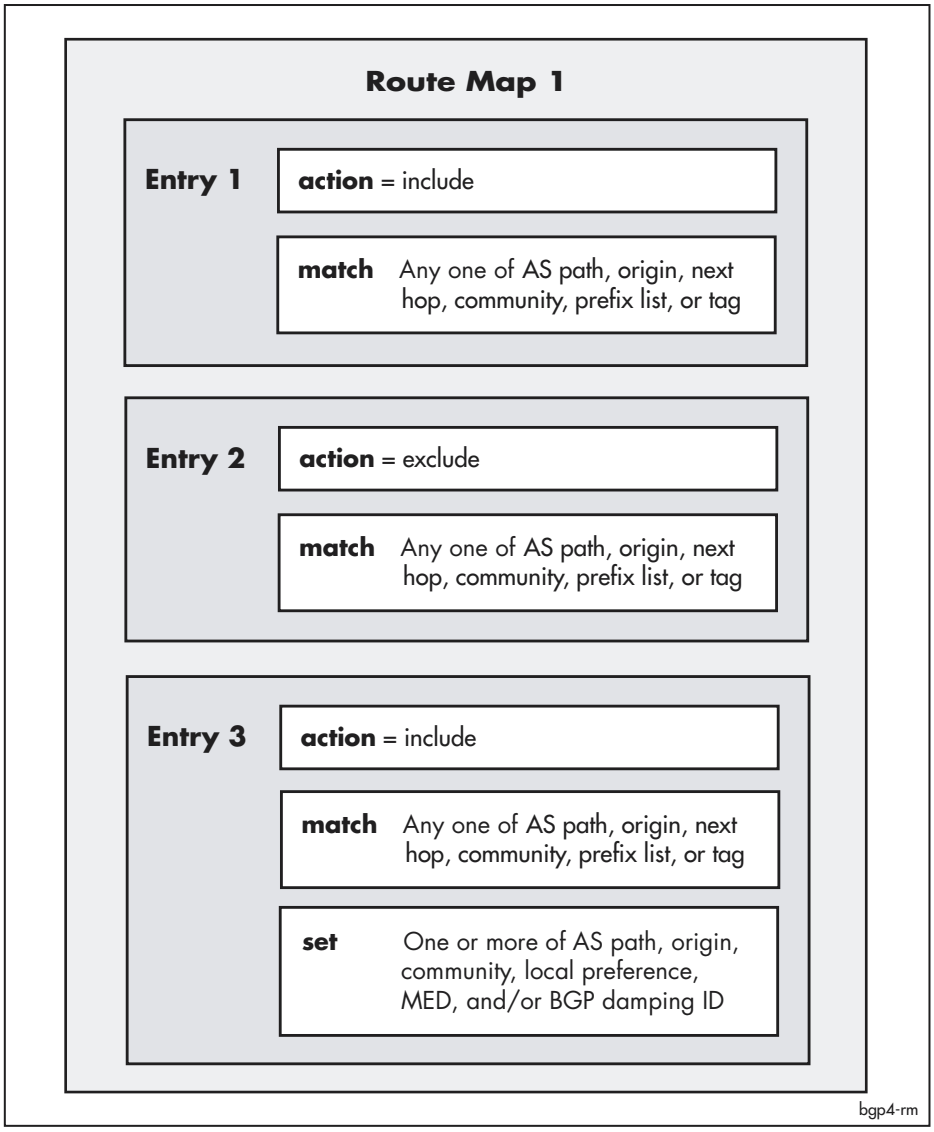
Route Maps

Route maps allow you to configure complex flexible filters. They achieve this by having several levels of structure:

- each route map consists of multiple entries
- each entry consists of an *action* (include or exclude) and at least one clause:
  - zero or one *match* clause, which determines which update messages or prefixes match the entry. If you do not specify a match clause, every update message or prefix matches.
  - zero or more *set* clauses, which change the BGP attributes of matching update messages or prefixes.

The following figure shows valid combinations of action and clause.

Figure 5-7: The structure of a route map



Through their set clauses, route maps can:

- add up to 10 AS numbers to the beginning of the AS_path attribute
- replace the community attribute with another of up to 10 entries or add up to 10 communities to the community attribute

- change the local_preference attribute to another value
- change the MED attribute to another value, or remove it
- change the origin attribute to another value
- set the BGP damping ID

When a BGP process uses a route map:

1. It checks the entries in order, starting with the lowest numbered entry, until it finds a match.
2. Then it takes the action specified by that entry's action parameter. If the action is **exclude**, it filters that update or prefix out. If the action is **include**, it filters that update or prefix in.
3. Then if the action is **include**, it modifies attributes as specified by the entry's set clauses, if there are any.
4. Then it stops processing the route map; it does not check the remaining entries.

Every route map ends with an implicit entry that matches all routes with an action of **include**. This ensures that if no entries in a route map generate a match, the update message or route is included without modification.

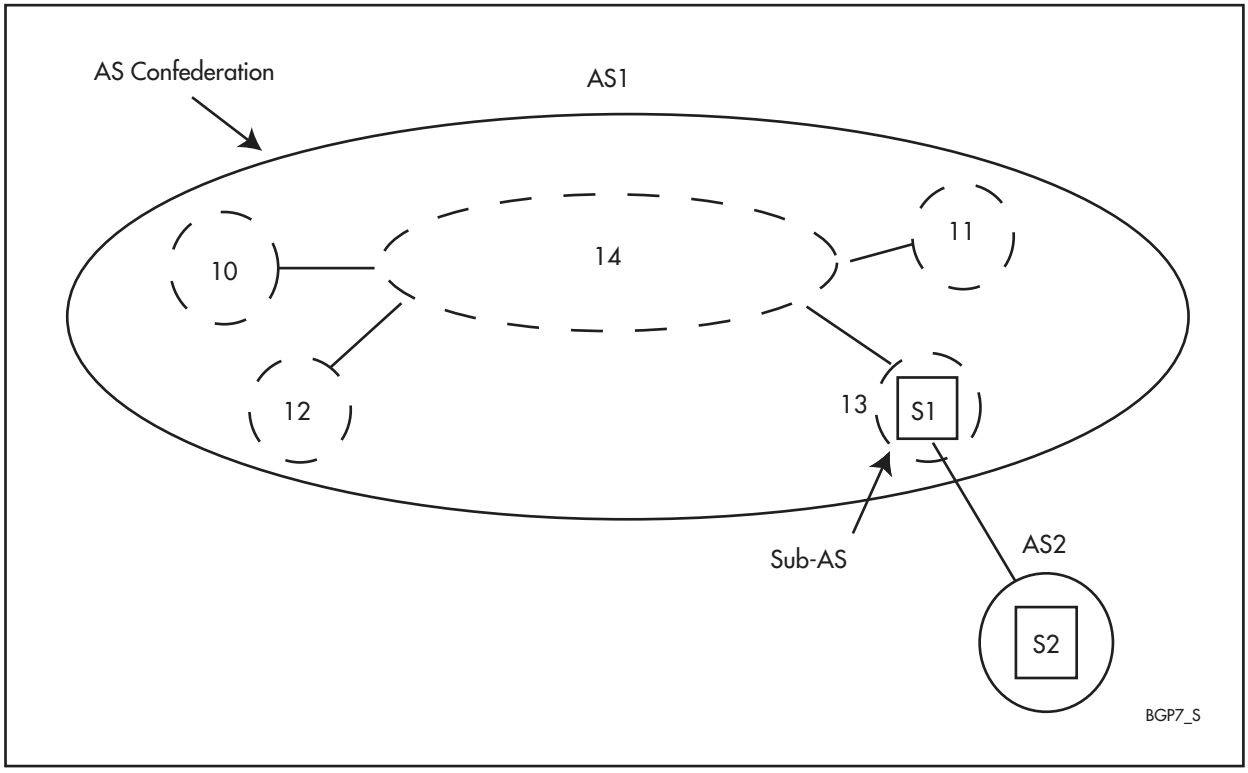
For information on creating and applying filters, see "How to Filter Routes for BGP" on page 5-27.

AS Confederations

An AS confederation is a collection of autonomous systems that are advertised as a single AS number to BGP speakers that are not members of the confederation. The autonomous systems in a confederation communicate between themselves using Confederation BGP (CBGP). AS confederations are used to subdivide autonomous systems with a very large number of BGP speakers to control routing policy.

Figure 5-8 is an example of an AS confederation. AS1 has been split into several sub-ASs (AS10-AS14). The original AS does not look any different to switch 2, which is outside the confederation. Switch 2 still sees AS1 rather than AS10-AS14. This implies that switches within the confederation are configured with an AS number for the confederation (for example, AS1), and an *AS member number*, an AS number visible only to those members within the confederation (for example, AS10).

Figure 5-8: Example of an AS confederation



Splitting a large AS into several smaller ones significantly reduces the number of intra-domain BGP connections. Unfortunately, splitting an AS may increase the complexity of routing policy based on AS path information for all members on the Internet. It also increases the maintenance overhead of coordinating external peering when the internal topology of this collection of ASs is modified.

Dividing an AS may unnecessarily increase the length of the sequence portions of the AS path attribute, and may adversely affect optimal routing of packets through the Internet.

Triggers

The Trigger Facility can be used to automatically run specified command scripts when particular triggers are activated. When a trigger is activated by an event, parameters specific to the event are passed to the script that is run. For a full description of the Trigger Facility, see the Trigger chapter.

Triggers can be created for two BGP events:

- when the switch runs low on memory
- when a peer changes state.

Module MODULE=BGP

Event MEMORY
Description The switch has run low enough on memory that BGP has had to start dropping routes.
Parameters There are no command parameters for this event.
Script Arguments There are no arguments to pass to the script.

Event PEERSTATE
Description The PEERSTATE trigger causes a trigger whenever a state change occurs that matches the peer, state and direction conditions. If the state is ANY and the direction is BOTH, two triggers are generated, one for leaving the old state and one for entering the new state.
Parameters The following command parameters can be specified in the **create** and **set trigger** commands:

Parameter	Description
peer=any|<*ipaddress*>	The IP address of the peer for which the state changes are interested in. This parameter is required in the **create trigger** command for BGP triggers, but is optional in the **set trigger** command unless the trigger event is changing from **memory** to **peerstate**.
bgpstate=idle|connect|active|opensent|openconfirm|established|any	The BGP state for which the trigger is required. This parameter is required in the **create trigger** command for BGP triggers, but is optional in the **set trigger** command, unless the trigger event is changing from **memory** to **peerstate**.
direction=enter|leave|both	Whether a match is made for the state the peer is leaving, entering, or both. This parameter is required in the **create trigger** command for BGP triggers, but is optional in the **set trigger** command unless the trigger event is changing from **memory** to **peerstate**.

Script Arguments The trigger passes the following arguments to the script:

Argument	Description
%1	The peer ID of the peer that has just undergone the state change.
%2	The state just left or entered.
%3	Whether the state was left or entered.

Example To create trigger 1, which activates whenever the switch becomes low on memory, initiating the script MEMLOW.SCP, use the command:

```
create trigger=1 module=bgp event=memory script=memlow.scp
   repeat=yes
```

To create trigger 2, which activates whenever peer=172.30.1.2 leaves the ESTABLISHED state, initiating the script PEERDOWN.SCP, use the command:

```
create trigger=2 module=bgp event=peerstate peer=172.30.1.2
   bgpstate=established direction=leave script=peerdown.scp
   repeat=yes
```

To modify trigger 2, which activates when any peer leaves the ESTABLISHED state, use the command:

```
set trigger=2 peer=any
```

How to Configure BGP Peers

How to Create a Basic BGP AS

This section describes how to configure your network as an Autonomous System by using EBGP to send and receive routing information from an external peer (for example, an ISP), and by using IBGP to communicate routes within the AS.

For this basic BGP setup, you need to configure:

- the external BGP speaker. This is the switch connected to the remote peer. In Figure 5-9, switch B is the external speaker and switch D is the remote peer. For the configuration procedure, see Table 5-2 on page 5-20. For checking and debugging, see Table 5-4 on page 5-22.
- the internal BGP speakers. These are all the other switches in the AS, connected to the external BGP speaker. In Figure 5-9, switches A and C are internal speakers. For the configuration procedure, see Table 5-3 on page 5-21. For checking and debugging, see Table 5-4 on page 5-22.

Figure 5-9: Example of the use of IBGP and EBGP

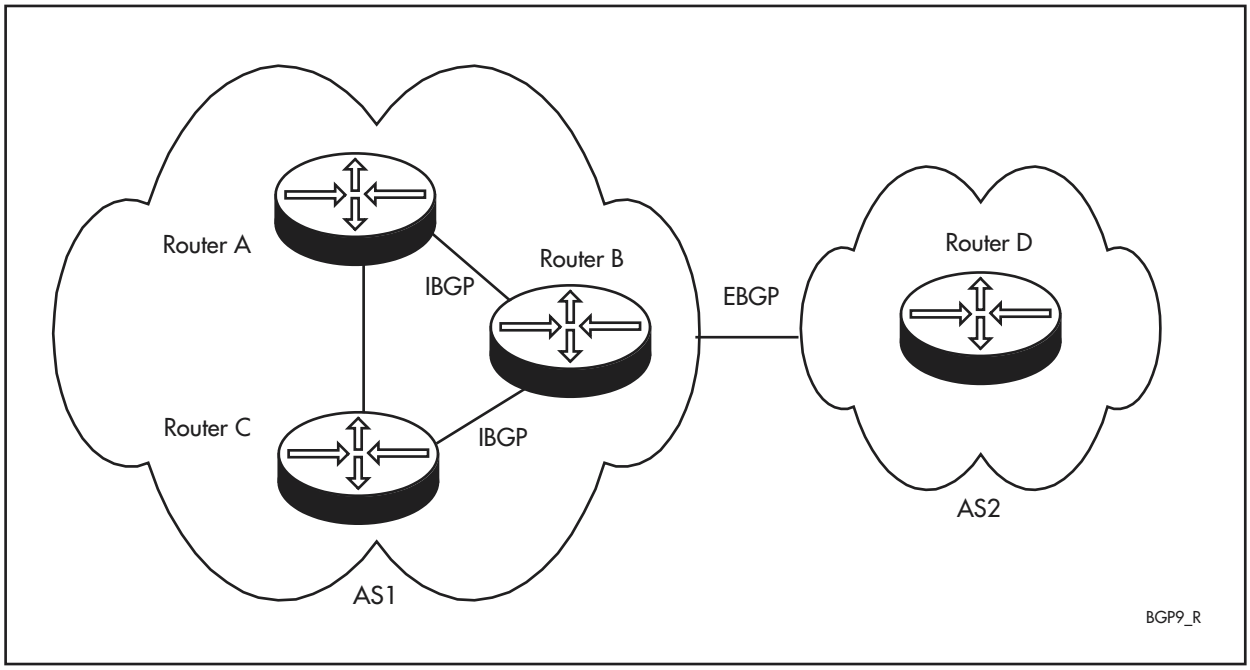


Table 5-2: Procedure for configuring the external BGP speaker

Step	Command	Action
1		Configure the lower-layer protocols that link the switch to the remote peer, for example frame relay, PPP.
2	**set ip autonomous**=1..65534	Assign your AS number to the switch.
3	add ip interface=*interface* ipaddress=*ipadd* [*other-options*] add ip route=0.0.0.0 interface=*interface* nexthop=*remote-peer-ipadd* enable ip	Configure IP on the interface that links the switch to the remote peer: • assign an IP address • create a default route, if the IP addresses assigned to the interfaces connecting switches B and D are on different subnets • enable IP
4	add ip interface=*interface* ipaddress=*ipadd* [*other-options*]	Configure IP on the interfaces that link the switch to each internal speaker.
5	**set bgp** routerid=*ipadd* [*other-options*]	Configure an interface on the switch to be the source of IP packets generated by BGP. This step is not required but is good practice.
6	**add bgp peer**=*remote-peer-ipadd* remoteas=*remote-peer-asn* [authentication={md5|none}] [client={yes|no}] [connectretry={default|0..4294967295}] [description=*description*] [ehops={default|1..255}] [fastfallover={no|yes}] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=routemap] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [password=*password*] [privateasfilter={no|yes}] [sendcommunity={no|yes}]	Add the remote peer to the switch. For the **peer** parameter, enter the IP address of the interface, on the remote peer, that the external speaker connects to. For the **remoteas** parameter, enter the remote peer's ASN.
7	**enable bgp peer**=*remote-peer-ipadd*	Enable the peer. The switch establishes a BGP connection with the remote peer and exchanges routing information.

Table 5-3: Procedure for configuring an internal BGP speaker

Step	Command	Action
1		Connect the switch directly to the external speaker and configure lower-layer protocols as required, for example VLANs.
2	**set ip autonomous**=1..65534	Assign your AS number to the switch.
3	add ip interface=*interface* ipaddress=*ipadd* [*other-options*] add ip route=*ext-speaker-ipadd* interface=*interface* nexthop=*ipadd* enable ip	Configure IP on the interface that links the switch to the external speaker: • assign an IP address • create a route to the external speaker if necessary • enable IP
4	**set bgp** routerid=*ipadd* [*other-options*]	Configure an interface on the switch to be the source of IP packets generated by BGP. This step is not required but is good practice.
5	**add bgp peer**=*external-speaker-ipadd* remoteas=*external-speaker-asn* [authentication={md5|none}] [client={yes|no}] [connectretry={default|0..4294967295}] [description=*description*] [ehops={default|1..255}] [fastfallover={no|yes}] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=routemap] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [password=*password*] [privateasfilter={no|yes}] [sendcommunity={no|yes}]	Add the external speaker to the switch as a BGP peer. For the **peer** parameter, enter the IP address of the external speaker's interface that this internal speaker connects to. For the **remoteas** parameter, enter the external speaker's ASN (which is the same as the internal speaker's ASN).
6	**enable bgp peer**=*ext-speaker-ipadd*	Enable the peer. The switch establishes a BGP connection with the external speaker and exchanges routes.

Table 5-4: Procedure for checking and debugging BGP peers

Step	Command	Action
1	**show bgp peer** **show bgp peer**=*external-speaker-ipadd*	Check that the connections are established and that the internal and external speakers are exchanging messages. For example output and definitions, see Figure 5-32 on page 5-150 and Figure 5-33 on page 5-151.
2	**show bgp**	Check the number of routes learned, and other information. For example output and definitions, see Figure 5-16 on page 5-132.
3	**show bgp route**[=*prefix*] [regexp=*aspathregexp*] community={internet|noadvertise| noexport|noexportsubconfed|*aa*:*xx*}[,...]} [add={no|yes}]	List information about all or a subset of the learned routes. Note that BGP may learn many thousands of routes. For example output and definitions, see Figure 5-35 on page 5-158.
4	**enable bgp** **debug**={damping|msg|state|update|all} [,...] [peer=*ipadd*]	Enable BGP debugging. Note that debugging may produce very large amounts of data.

How to Create BGP Peers Using Peer Templates

Peer templates make it easier to create BGP peers when many peers have identical inbound and outbound filtering policies, or timer values. They enable you to define a template set of these values, which you can subsequently apply to many different peers. You can assign a template to a BGP peer either when you create the peer, or afterwards.

Table 5-5: Procedure for using a template to create a BGP peer

Step	Command	Action
1	**add bgp peertemplate**=1..30 [client={yes|no}] [connectretry={default|0..4294967295}] [description=*description*] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=*routemap*] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [privateasfilter={no|yes}] [sendcommunity={no|yes}]	Create the template. You can specify most of the peer settings in the template.
2	**show bgp peertemplate**[=1..30]	Check the template settings.
3	**add bgp peer**=*ipadd* remoteas=*asn* policytemplate=1..30 [authentication={md5|none}] [password=*password*] [description=*description*] [ehops={default|1..255}] [fastfallover={no|yes}]	Create the peer entry and attach the template to it. You can also specify peer settings that are not available in the template.

How to Modify BGP Peers (Without Templates)

To modify a peer, unless the peer is using a template, use the command:

```
set bgp peer
```

You do not need to disable the peer first.

For information on changing peers that use templates, see "How to Modify BGP Peers that Use a Template" .

Once you have modified the peer, the switch needs to update that peer. The switch supports the following RFCs for updating modified peers:

- RFC 2918 Route Refresh Capability for BGP-4.
- RFC 2842 Capabilities Advertisement with BGP-4.

Automatic updates

You can configure the switch to make updates automatically by using the command:

```
enable bgp autosoftupdate
```

This is disabled by default. Note that you must enable automatic updating before you modify the peer.

Manually-triggered updates

Alternatively, you can manually trigger the BGP peer to reset by using the command:

```
reset bgp peer={all|ipadd} soft={in|out|all}
```

The **soft** parameter determines the direction to update. There are two types of updates:

- Inbound updates, which reset routes that the switch receives from the peer. To trigger these, the switch sends a Route Refresh message to the peers it receives routes from. The Route Refresh message triggers the peers to resend a BGP Update message.
- Outbound updates, which reset routes the switch sends. To reset these, the switch simply sends a BGP Update message to the affected BGP peers.

If you do not manually or automatically trigger an immediate update, changes to the peer take effect when the switch next receives an update message from that peer or sends an update message to it.

To see if automatic updating is enabled, use one of the commands:

```
show bgp
```

```
show bgp peer
```

In the command **show bgp peer**, you can see that the switch and its peer have negotiated automatic updating when the "Capabilities" entry contains "Route Refresh". This command also displays the number of route refresh messages received from and sent to the peer.

How to Use a Template to Modify BGP Peers

You can apply a template to an existing peer, which overrides its current settings with the template settings.

Table 5-6: Procedure for using a template to modify a BGP peer

Step	Command	Action
1	**add bgp peertemplate**=1..30 [client={yes|no}] [connectretry={default|0..4294967295}] [description=*description*] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=*routemap*] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [privateasfilter={no|yes}] [sendcommunity={no|yes}]	Create the template. You can specify most of the peer settings in the template.
2	**show bgp peertemplate**[=1..30]	Check the template settings.
3	**set bgp peer**=*ipadd* remoteas=*asn* policytemplate=1..30 [authentication={md5|none}] [password=*password*] [description=*description*] [ehops={default|1..255}] [fastfallover={no|yes}]	Attach the template to the peer. You can also specify peer settings that are not available in the template.
4	**reset bgp peer soft** reset bgp peer=*ipadd* soft={in|out|all}	If automatic updating is not enabled, trigger the peer to update.

How to Modify BGP Peers that Use a Template

Once you have assigned a template to a peer, the method you use to modify the peer depends on the type and scope of the modification.

- To change a parameter on **all** peers that use the template, when the parameter **is** available in the template, change the template. Use the command:

```
set bgp peertemplate=1..30 [client={yes|no}]
    [connectretry={default|0..4294967295}]
    [description=description]
    [holdtime={default|0|3..65535}]
    [infilter={none|300..399}] [inpathfilter={none|1..99}]
    [inroutemap=routemap] [keepalive={default|1..21845}]
    [local={none|1..15}] [maxprefix={off|1..4294967295}]
    [maxprefixaction={terminate|warning}]
    [minasoriginated={default|0..3600}]
    [minrouteadvert={default|0..3600}]
    [nexthopself={no|yes}] [outfilter={none|300..399}]
    [outpathfilter={none|1..99}] [outroutemap=routemap]
    [privateasfilter={no|yes}] [sendcommunity={no|yes}]
```

- To change an **individual** peer when the parameter **is not** available in the template, specify the peer and parameter by using the command:

```
set bgp peer=ipadd remoteas=asn
    [authentication={md5|none}] [password=password]
    [description=description] [ehops={default|1..255}]
    [fastfallover={no|yes}]
```

- To change an **individual** peer when the parameter **is** available in the template, first remove the template from the peer by using the command:

```
set bgp peer=ipadd policytemplate=
```

  Specifying **policytemplate=** like this with no number removes the template. The peer retains the template's settings. Then change the settings you need to for that peer by using the command:

```
set bgp peer=ipadd [other-options]
```

For information on resetting peers after modification, see "How to Modify BGP Peers (Without Templates)" .

How to Delete BGP Peers

Table 5-7: Procedure for deleting a BGP peer

Step	Command	Action
1	**disable bgp peer**={*ipadd*|all}	Disable the peer. Enabled peers cannot be deleted.
2	**delete bgp peer**={*ipadd*|all}	Delete the peer.

As soon as the peer session goes down, the switch removes any routes it learned from that peer. Once the route selection timer expires, the switch withdraws the routes from any other peers it had advertised them to.

How to Filter Routes for BGP

Filters allow you to control the routes that BGP learns and advertises. There are three different types of filter: AS path filters, prefix filters and route maps. For a description of the filter types, see "BGP Route Filtering" on page 5-13.

In very general terms, configuring any of these filters involves two steps:

1. Create the filter.
2. Apply it.

   You can apply AS path filters, prefix filters and route maps to update messages received from a BGP peer, or sent to the BGP peer.

   You can also apply a route map to a particular prefix, independent of the peer. See "How to apply route maps to imported routes" on page 5-35.

How to Configure AS Path Filters

The AS path attribute lists the AS numbers of every Autonomous System that the routing information in an update message has passed through. It shows the path the update message has taken, and how "close" the routes are to the switch. You can filter to accept or reject update messages on the basis of all or part of their AS path.

Creating AS path lists

The first step is to create an *AS path list* and add entries to it by using one of the commands:

```
add ip aspathlist=1..99 [entry=1..4294967295]
   include=aspath-reg-exp

add ip aspathlist=1..99 [entry=1..4294967295]
   exclude=aspath-reg-exp
```

Each entry uses a regular expression, *aspath-reg-exp*, to both specify the AS numbers that the entry matches, and to establish whether matching AS numbers are included or excluded. Table 5-8 shows regular expression syntax and examples.

Table 5-8: Syntax for AS path regular expressions

Token	Description	Examples	Meaning of example
<AS number>	Matches that identical AS number.	123	Matches any AS path attribute that contains AS 123 (but not 1234, 12345, or 5123).
^	Matches the start of the AS path attribute.	^123	Matches AS path attributes that have AS 123 as the first AS.
$	Matches the end of the AS path attribute.	^$	Matches an empty AS path attribute.
		^123$	Matches an AS path attribute with a single AS number, 123.
<space>	Separates AS numbers in a regular expression.	"123 456"	Matches AS path attributes that contain ASs 123 and 456, in that order, with no other AS numbers between them.
" "	Surrounds regular expressions that contain spaces.		
.	Matches any AS number.	.*	Matches all AS path attributes.
*	Matches zero or more repetitions of the preceding token in the AS path list being filtered	"123 .* 456"	Matches AS path attributes that contain ASs 123 and 456, in that order, with any number of other AS numbers between them.
+	Matches one or more repetitions of the preceding token in the AS path list being filtered.	"123 .+ 456"	Matches AS path attributes that contain ASs 123 and 456, in that order, with at least one other AS number between them.

You can apply AS path lists directly to BGP peers, or use them in route maps (see "How to Configure Route Maps" on page 5-30).

Applying path lists to peers

To apply the AS path list directly as a filter on a BGP peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn [inpathfilter=1..99]
    [outpathfilter=1..99] [other-options]

set bgp peer=ipadd [inpathfilter=1..99] [outpathfilter=1..99]
    [other-options]
```

The **inpathfilter** parameter applies the AS path list as a filter on update messages that the switch receives from the peer. The switch only accepts update messages if they match an AS path list entry that has the action **include**. If an update message matches an entry with the action **exclude**, the switch rejects the update. If an update message does not match any entry in the AS path list, the switch rejects the update. This is because each non-empty AS path list ends with an implicit entry that matches any AS path list and has the action **exclude**.

The **outpathfilter** parameter applies the AS path list as a filter on update messages that the switch sends to the peer. The switch only sends update messages if the update's AS path attribute matches an entry that has the action **include**. If a route matches an entry with the action **exclude**, the switch does not advertise it to that peer. If an update message does not match any entry in the AS path list, the switch does not advertise it to that peer.

An empty AS path list is equivalent to a path list that matches all AS numbers and has the action **include**.

How to Configure Prefix Filters

Prefix filters use IP routing filters. A routing filter is an IP filter with a number in the range 300-399. It matches on the source and mask of the prefix, and specifies whether matching prefixes are included or excluded.

You can use a prefix filter to reject some of the routes from an update message, without rejecting the whole update. This enables you to configure the switch to accept only routes for particular networks from a particular peer, and to send only routes for particular networks to a particular peer. Therefore, the switch can send or receive subsets of routes that have originated from or traversed a particular AS (or list of ASs), which is not possible with AS path filtering.

Creating IP routing filters

To create a routing filter, use the command:

```
add ip filter=300..399 action={include|exclude} source=ipadd
    [smask=ipadd] [entry=1..255]
```

The **source** parameter is the network IP address of the subnet to be filtered.

The **smask** parameter determines how many bits of the prefix are significant. When the switch checks prefixes in an update message against the filter, it only checks the significant bits.

By default, new entries are added at the end of the filter. If you want the entry to be checked before some of the other entries, give it a lower entry number. This pushes existing entries with the same or higher number further down the list.

Applying routing filters to peers

To apply the routing filter as a prefix filter on a BGP peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn [infilter=300..399]
    [outfilter=300..399] [other-options]

set bgp peer=ipadd [infilter=300..399] [outfilter=300..399]
    [other-options]
```

The switch checks every route in the update message against every entry in the filter, starting with the entry with the lowest entry number, until it finds a match or gets to the end of the filter.

The **infilter** parameter applies the filter to update messages that the switch receives from the peer. If the switch finds a match and that match has action **exclude**, the switch rejects that route. If the match has action **include**, or there is no match, the switch accepts the route.

The **outfilter** parameter applies the filter to update messages that the switch sends to the peer. If the switch finds a match and that match has action **exclude**, the switch removes that route from the update message. If the match has action **include**, or there is no match, the switch leaves the route in the update message and therefore advertises it to the peer.

How to Configure Route Maps

A route map consists of multiple entries, which are in effect individual filters. Each entry specifies both what it matches on, in a *match* clause, and what is done to matching traffic, in the entry's *action* and any *set* clauses it has.

Most set clauses modify the BGP attributes of matching update messages. If you want to change the attributes of all candidate routes, configure an entry with no match clause. Such an entry matches all update messages.

This section describes how to:

- create a route map
- configure match clauses
- configure set clauses
- apply the route map to a BGP peer
- apply the route map to a BGP prefix, independent of the peer

How to create a route map

You do not have to create a route map as a separate step—adding the first entry automatically creates it.

How to configure an entry with a match clause

The match clause for a route map entry determines which update messages or prefixes match the entry. Each entry can only match on one characteristic. Available characteristics are:

- AS path list
- community list
- origin attribute
- next_hop attribute
- prefix list
- tag

Matching on AS path list

The first step is to create an *AS path list* and add entries to it by using one of the commands:

```
add ip aspathlist=1..99 [entry=1..4294967295]
    include=aspath-reg-exp

add ip aspathlist=1..99 [entry=1..4294967295]
    exclude=aspath-reg-exp
```

Table 5-8 on page 5-28 shows the valid syntax for the regular expression *aspath-reg-exp* and gives syntax examples.

Then use the AS path list in the match clause of a route map by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match aspath=1..99
```

When the switch uses this route map to examine an update message, the switch goes through the entries in the AS path list. The update matches if an entry in

the AS path list matches the AS_path in the update message, **and** that AS path list entry is an **include** entry.

If the update message matches, the switch carries out the action of the route map. This is one of:

- exclude the update message
- include the update message without modification
- include the update message and modify its attributes

Note that the action (include/exclude) of the AS path list and the action of the route map entry are separate. Table 5-9 shows the effect of each combination.

Table 5-9: The effect of actions in AS path list and route map entries

AS path list entry	Route map entry	Action when route map applied
include	include	An update message with that AS_path matches, and is processed
include	exclude	An update message with that AS_path matches, and is discarded
exclude	include	An update message with that AS_path does not match. The switch continues checking to see if the update message matches other entries in the route map.
exclude	exclude	An update message with that AS_path does not match. The switch continues checking to see if the update message matches other entries in the route map.

In this context, the parameters **include** and **exclude** in the AS path list do not indicate whether the matching update message is allowed or dropped; they simply indicate whether the update matches or does not match the path list. This is different to the behaviour when you use the AS path list itself as a filter, as described in "How to Configure AS Path Filters" on page 5-27.

Example comparing AS path filter and route map

Compare this configuration, which uses an AS path filter:

```
add ip aspathlist=2 entry=1 exclude="^$"
add ip aspathlist=2 entry=2 include="15557"
set bgp peer=192.168.200.201 outpathfilter=2
```

with this configuration, which uses a route map and matches on AS path list:

```
add ip aspathlist=2 entry=1 include="^$"
add ip aspathlist=2 entry=2 exclude="15557"
add ip routemap=outdef3 entry=1 action=exclude match
   aspathlist=2
set bgp peer=192.168.200.201 outroutemap=outdef3
```

With both these configurations, the switch drops update messages with empty AS paths, and advertises update messages with an AS path containing 15557. For the route map to achieve this (the second configuration):

- The AS path list has to **include** empty paths, so that the empty path matches the path list, and therefore is included into the route map's action of dropping packets that match the path list.
- The AS path list has to **exclude** updates whose AS path includes 15557. This excludes those updates from the route map's action of dropping packets that match the path list, so they are not dropped.

Matching on community list

The first step is to create a *community list* and add entries to it by using one of the commands:

```
add ip communitylist=1..99 [entry=1..4294967295]
    include={internet|noexport|noadvertise|
    noexportsubconfed|AA:XX}[,...]

add ip communitylist=1..99 [entry=1..4294967295]
    exclude={internet|noexport|noadvertise|
    noexportsubconfed|AA:XX}[,...]
```

Then use the community list in the match clause of a route map by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match community=1..99
    [exact={no|yes}]
```

Note that the action (include/exclude) of the community list and of the route map entry are separate. This leads to the same behaviour as the distinction between the AS path list include/exclude parameters and the route map entry action. For a discussion of the distinction between these two include/exclude actions, see "Matching on AS path list" on page 5-30 and Table 5-9 on page 5-31.

If you specify **exact=yes**, an update message only matches the route map entry if its community attribute contains all the communities specified in the community list, and no other communities. If you specify **exact=no**, which is the default, then the set of communities in the attribute list of the update message must contain all the communities in the specified community list, but can also contain other communities.

Matching on next hop

Create a route map entry that specifies the IP address of the next hop by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match nexthop=ipadd
```

This lets you select or discard routes that traverse a particular node.

Matching on origin

Create a route map entry that specifies the value of the origin attribute by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match
    origin={egp|igp|incomplete}
```

This lets you select or discard routes depending on how BGP learned them: internally, externally, or from another means (such as statically-configured routes).

Matching on prefix list

A prefix list consists of a list of entries, each of which specifies:

- an IPv4 prefix, and a mask length or range of mask lengths. Together these specify the prefixes that the entry applies to.
- whether those prefixes explicitly match or explicitly do not match the prefix list.

All the match options described previously—AS path, community, next hop and origin—match on the **attributes** in an update message. Prefix list does not; it matches prefixes.

To use a prefix list, first create the prefix list and add entries to it by using the command:

```
add ip prefixlist=name entry=1..65535
    [action={match|nomatch}] [masklength=range] [prefix=ipadd]
```

The **masklength** parameter specifies the range of prefix mask lengths matched by this entry in the prefix list. The *range* is either a single CIDR mask from 0 to 32, or two masks separated by a hyphen. These options are valid for setting the mask length:

- As a mask length range (**masklength=*a*-*b*)**.

  For a route to match against this entry, its prefix mask length must be between *a* and *b* inclusive. *a* must be less than *b*.

- As a single mask length (**masklength=*a***).

  For a route to match against this entry, its prefix mask length must be exactly *a*.

- As an implicit mask length, by not specifying **masklength** (for example, **prefix=192.168.0.0**).

  For a route to match against this entry, its prefix mask length must correspond exactly to the mask for the class of the given address—in this example, 24.

Once you have created the prefix list, use it in the match clause of a route map by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match prefixlist=name
```

Note that the action of the prefix list and of the route map entry are separate. Table 5-10 shows the effect of each combination.

Table 5-10: The effect of actions in prefix list and route map entries

Prefix list entry	Route map entry	Action when route map applied
match	include	An update message that contains the prefix matches the route map entry. The prefix is processed.
match	exclude	An update message that contains the prefix matches the route map entry. The prefix is removed from the update message. Other prefixes in the update are not removed.
nomatch	include	An update message that contains the prefix does not match the route map entry. The switch continues checking to see if the update message matches other entries in the route map.
nomatch	exclude	An update message that contains the prefix does not match the route map entry. The switch continues checking to see if the update message matches other entries in the route map.

In this context, the parameters **match** and **nomatch** in the prefix list do not indicate whether the prefix is allowed or dropped; they simply indicate whether the prefix matches or does not match the prefix list.

Matching on tag

See "How to Control Import of Static Routes" on page 5-49 for instructions on tagging routes and using the tags.

How to configure an entry with a set clause

Once you have determined what update messages or prefixes a route map entry matches, you can configure set clauses to change the attributes of matching items.

To create a set clause for an entry, use one of the commands shown in Table 5-11.

Table 5-11: The available set clauses for route maps

Command	Result
add ip routemap=*routemap* entry=1..4294967295 set aspath={1..65534[,...]}	Adds up to 10 AS numbers at the beginning of the AS path attribute.
add ip routemap=*routemap* entry=1..4294967295 set community={noexport|noadvertise| noexportsubconfed|*AA:XX*}[,...]} [add={no|yes}]	Either: • replaces the community attribute with a list of up to 10 community values, if **add=no** (the default), or • adds up to 10 community values to the community attribute, if **add=yes**
add ip routemap=*routemap* entry=1..4294967295 set localpref=0..4294967295	Replaces the existing local_preference attribute, or sets it if it was not already set.
add ip routemap=*routemap* entry=1..4294967295 set med={0..4294967295|remove}	Replaces the existing MED attribute, or sets it if it was not already set, or if you specify **med=remove**, deletes the MED attribute.
add ip routemap=*routemap* entry=1..4294967295 set origin={igp|egp|incomplete}	Replaces the existing origin attribute, or sets it if it was not already set.
add ip routemap=*routemap* entry=1..4294967295 set bgpdampid=1..100	Sets the BGP route flap damping ID that is given to matching routes (see "Damping routes on specific peers" on page 5-38).

A prefix list can match a subset of prefixes in an update message. You can use this to change the attributes of some of the prefixes in an outgoing update, without having to change the attributes of all the prefixes. However, an update message contains just one set of attributes, which must apply to all the prefixes in the update. Therefore, the switch splits the original update into two updates:

- one that contains the original attribute values and the prefixes that were not included by the route map entry, and
- one that contains the new attribute values and the prefixes that were included by the route map entry

How to apply route maps to BGP peers

To use the route map to filter or modify update messages that it receives from a peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn inroutemap=routemap
    [other-options]

set bgp peer=ipadd inroutemap=routemap [other-options]
```

To use the route map to filter or modify update messages that it sends to a peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn outroutemap=routemap
    [other-options]

set bgp peer=ipadd outroutemap=routemap [other-options]
```

The switch checks every route in the update message against every entry in the filter, starting with the entry with the lowest entry number, until it finds a match or gets to the end of the filter.

If your route map is intended to modify the community attribute of outgoing update messages, you also need to enable the switch to set the community attribute in messages to that peer. Use one of the commands:

```
add bgp peer=ipadd remoteas=asn outroutemap=routemap
    sendcommunity=yes [other-options]

set bgp peer=ipadd outroutemap=routemap sendcommunity=yes
    [other-options]
```

How to apply route maps to imported routes

The switch is able to import routes into BGP that it learnt by non-BGP means—static routes, or routes learnt by OSPF or RIP.

You can apply a route map to this importation process so that the imported routes are given certain attributes, or so that certain routes are blocked from being imported. You can apply the route map by using either of the commands:

```
add bgp import={interface|ospf|rip|static} routemap=routemap

add bgp network=prefix[/0..32] [mask=mask] routemap=routemap
```

The switch uses the route map to:

- filter routes or set attributes when it imports the routes into BGP
- set attributes on any update message in which it advertises the routes

Note that the entries in route maps that are applied to BGP importing cannot have match clauses that match on AS path or community. These attributes are not relevant to non-BGP routes. The route map entries can match on origin, next hop or tag.

How to Optimise BGP

How to Minimise the Impact of Unstable EBGP Routes

The problem

BGP-managed networks are more efficient and stable when routing update messages are kept to a minimum. Under some network conditions, BGP generates an excessive rate of update messages due to "route flapping", in which some routes frequently oscillate between being reachable and unreachable. Route flapping causes a ripple effect through the BGP network as the changes are propagated. In extreme cases, the network does not reach a stable, converged state for a substantial period of time.

The solution: route flap damping

BGP route flap damping, as defined in RFC 2439, limits the impact and visibility of route flapping to a switch's BGP peers. When a local BGP peer learns a route, it immediately adds it to its Routing Information Base (RIB) but may not immediately select or advertise it. It can only select or advertise the route once its internal BGP suppression engine considers that the route is sufficiently stable. A new route has no history of instability, so is immediately made available as normal. A route that has been previously learned but withdrawn, however, may be suppressed for a period of time, based on the severity of its previous instability. Therefore, BGP route flap damping suppresses routes that are considered too unstable to be used locally or advertised to any BGP peers, until the route has remained stable for a sufficient period of time. Persistently unstable routes may be excluded from selection indefinitely. By taking into account the prior behaviour of that route, the switch can estimate the future stability of the route accurately enough to reduce switch processing load without significantly impacting the convergence time for more stable routes.

Figure of Merit (FoM)

A route's history of instability is recorded via the maintenance of a statistic defined as a Figure of Merit (FoM) by RFC 2439. The FoM for a particular BGP route quantifies that route's history of stability, or lack of stability. When the switch learns a new route, it grants the route an initial FoM of zero, indicating no history of instability. At this point, the BGP suppression engine is not interested in the route.

If an EBGP peer ever withdraws a route, the switch increments the FoM for that route by 1000. As soon as a route earns a non-zero instability metric, the BGP suppression engine begins to monitor it, but in most configurations takes no other action at this stage. If the route exhibits further instability, its FoM increases. Once the FoM exceeds a configurable suppression threshold, the BGP suppression engine begins to suppress it. At this stage, BGP no longer selects or advertises the route. Each route's FoM is reduced over time at a configurable rate, so if a suppressed route remains stable for a sufficient period of time, its status is eventually downgraded to monitored. If a monitored route's FoM reaches zero, or a value very close to zero, monitoring stops until further instability is observed. Figure 5-10 on page 5-37 shows the states and progress between them. Figure 5-11 on page 5-37 shows how a route's FoM is maintained over time.

Figure 5-10: The process applied to routes by route flap damping

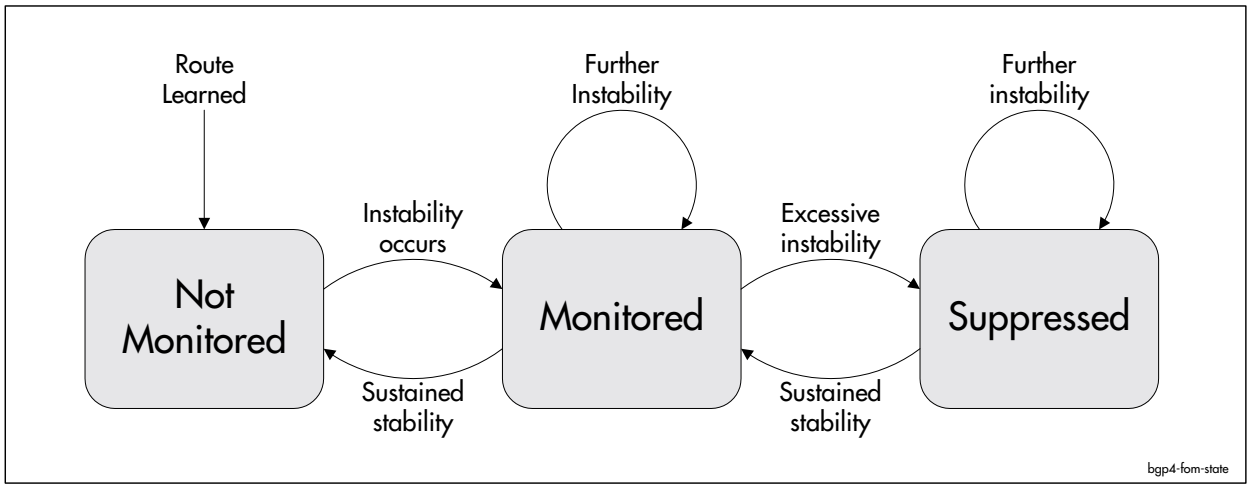
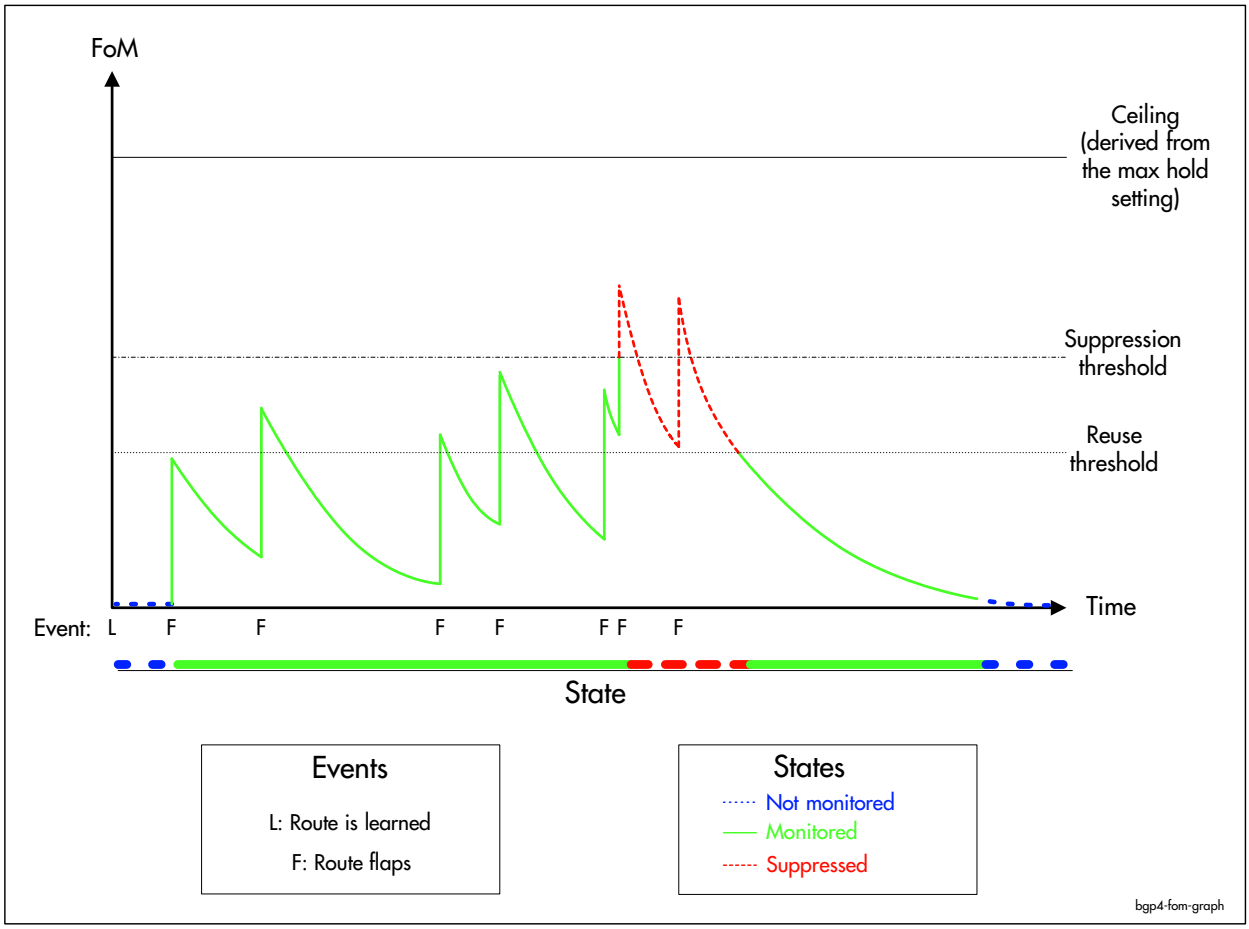


Figure 5-11: Change in FoM over time as a route flaps, showing when the route is suppressed



How to configure route flap damping

BGP route flap damping is disabled by default. You can enable it on all routes, or limit it to routes received by particular peers. You can use the default threshold settings, or specify different settings. The settings are captured by damping **parameter sets**, which are collections of four configuration parameters that determine the nature of the treatment received by relevant routes from the BGP suppression engine. Table 5-12 shows the parameters and the effect of increasing or decreasing each value.

Table 5-12: The effect of modifying route flap damping parameters

Parameter	Meaning of parameter	Change	Effect of change
suppression	**Suppression** is an FoM value. When a route's FoM exceeds this threshold, the route is suppressed.	Raised	Increases the number of times the route can become unreachable before it is suppressed.
		Lowered	Decreases the number of times the route can become unreachable before it is suppressed.
reuse	**Reuse** is an FoM value. Once a route is suppressed, it remains suppressed until its FoM falls below this threshold.	Raised	Decreases the minimum time that the route is suppressed for.
		Lowered	Increases the minimum time that the route is suppressed for.
halflife	**Halflife** is the time interval within which the route's FoM will halve, if the route remains stable. For example, if the halflife is 15, the FoM of a stable route reduces by 50% over a 15 minute period, 75% over a 30 minute period, and so on.	Lengthened	Lowers the FoM more slowly, so increases the time the route is suppressed for.
		Shortened	Lowers the FoM more quickly, so decreases the time the route is suppressed for.
maxhold	**Maxhold** multiplied by **halflife** is the maximum period of time that a route must remain stable in order to become unsuppressed. For example, if **halflife** is 15 and **maxhold** is 4, the route is unsuppressed after 60 min of stability even if its FoM still exceeds **reuse**.	Increased	Increases the time that a severely unstable route must be stable for, before it is unsuppressed.
		Reduced	Decreases the time that a severely unstable route must be stable for, before it is unsuppressed.

Damping all routes

To apply route flap damping to all incoming routes by using the default parameter settings, use one of the commands:

```
enable bgp damping

enable bgp damping parameterset=0
```

If you do not want to use the default parameter settings, change the settings for the default parameter set 0 by using the command:

```
set bgp damping parameterset=0 [description=description]
    [suppression={default|1..20000}]
    [reuse={default|1..20000}] [halflife={default|1..45}]
    [maxhold={default|1..8}]
```

For more information about the default parameter set, see "The default parameter set" on page 5-39.

When you enable route flap damping globally, the switch examines the instability history of every route it receives from every remote peer, and suppresses the route if appropriate.

Damping routes on specific peers

To limit route flap damping to some or all routes received by a particular peer, follow this procedure:

Step	Command	Action
1	**create bgp damping parameterset**=1..100 [description=*description*] [suppression={default|1..20000}] [reuse={default|1..20000}] [halflife={default|1..45}] [maxhold={default|1..8}]	Create a parameter set.
2	**enable bgp damping** parameterset=1..100	Enable BGP damping for the parameter set. You must enable only the desired parameter set, or route flap damping applies to all routes.
3	**show bgp damping**	Check the state and settings of the parameter set.
4	**add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match aspath=1..99 **add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match community=1..99 [exact={no|yes}] **add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match nexthop=*ipadd* **add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match origin={egp|igp|incomplete} **add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match prefixlist=*name*	Create a route map to match the routes you want to apply route flap damping to. If you are using an AS path list or community list to match routes, also configure the list. See "How to Configure Route Maps" on page 5-30 for more information.
5	**add ip routemap**=*routemap* entry=1..4294967295 set bgpdampid=1..100	Associate the damping parameter set with routes that match the route map. The **bgpdampid** parameter is the number of the route flap damping parameter set.
6	set bgp peer=*ipadd* inroutemap=*routemap*	Configure the BGP peer to use the route map on update messages it receives.

The default parameter set

If you enable route flap damping, all routes that you do not specifically apply a parameter set to are processed by the default parameter set. This set is numbered 0, and by default has the following settings:

- suppression=2000
- reuse=750
- halflife=15
- maxhold=4

The purpose of the default parameter set is to suppress routes that are not processed by any other parameter set, as shown in Figure 5-12 on page 5-40. Therefore you cannot limit the default parameter set to routes received by particular peers.

You can disable the default parameter set without disabling the whole of BGP damping by using the command:

```
disable bgp damping parameterset=0
```

You cannot destroy the default parameter set, but you can modify its settings by using the command:

```
set bgp damping parameterset=0 [description=description]
    [suppression={default|1..20000}]
    [reuse={default|1..20000}] [halflife={default|1..45}]
    [maxhold={default|1..8}]
```

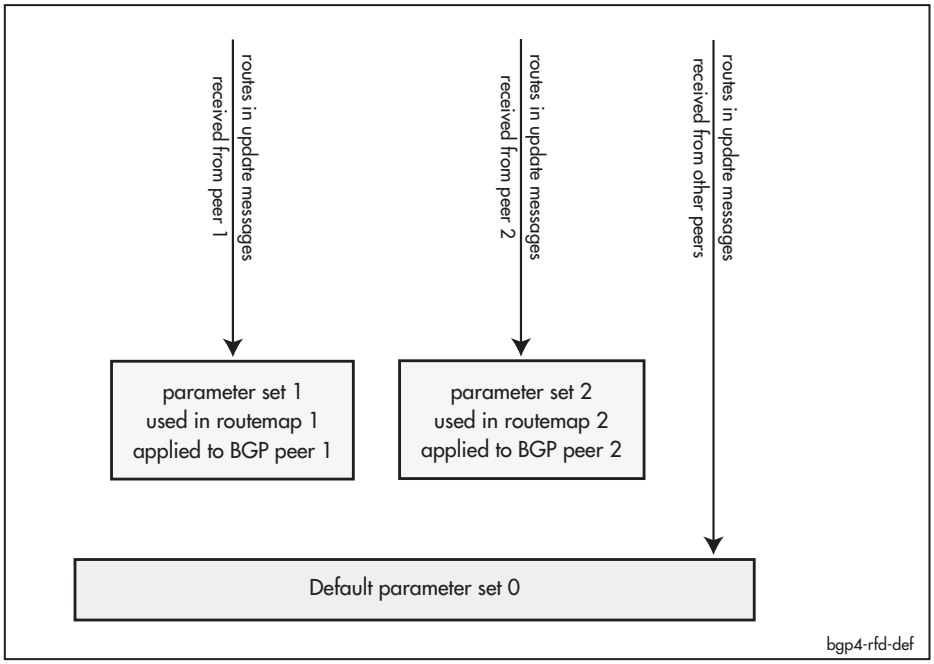
You can return the modified default parameter set to its original values by using the command:

```
set bgp damping parameterset=0 suppression=2000 reuse=750
   halflife=15 maxhold=4
```

Alternatively, you can reset the default parameter set and at the same time destroy all other parameter sets and clear all instability history. To do this, use the command:

```
purge bgp damping
```

Figure 5-12: Use of certain parameter sets on some peers and the default set on all others



Displaying route flap damping information

To display the current state of route flap damping and each parameter set, and the parameter values for each parameter set, use the command:

```
show bgp damping
```

To display a list of all monitored and suppressed routes, including their current FoM and the period of time which they must remain stable in order to progress

from suppressed to monitored, or from monitored to not monitored, use the command:

```
show bgp damping routes
```

How to Withdraw Routes As Soon As they Fail

The problem

By default, when the interface that supports an EBGP peer session goes down, the corresponding peer session is not reset until that session's hold timer expires.

The solution: fast fallover

Fast fallover is an option that you can enable for individual peers, that resets the session as soon as the switch's interface to the peer goes down. It provides fast failover in case of link failures, because the switch withdraws paths as soon as the link goes down, rather than waiting for up to three minutes to propagate the change. As a result, fast fallover greatly improves the rate of convergence to new network topology.

How to configure fast fallover

Fast fallover is disabled by default. To enable fast fallover on the switch's link to a peer, use one of the commands:

```
add bgp peer=ipadd remoteas=1..65534 fastfallover=yes
    [other-options]
```

```
set bgp peer=ipadd fastfallover=yes [other-options]
```

To disable fast fallover on the switch's link to a peer, use the command:

```
set bgp peer=ipadd fastfallover=no [other-options]
```

To see if fast fallover is enabled on the switch's link to a peer, use the command:

```
show bgp peer=ipadd
```

What about unstable links?

Certain types of links can be particularly unreliable. If you enable fast fallover on such a link, the BGP sessions supported by the link will flap. This causes frequent route changes and excessive update messaging within the network.

If a link to a peer is susceptible to brief outages, we do not recommend enabling fast fallover on it. If it is susceptible to longer outages, fast fallover may be desirable because it lets the switch rapidly change to an alternative path.

Using it with VLANs

BGP peer fast fallover recognises that a link has gone down when the relevant layer 2 interface notifies BGP that its link status has changed from up to down. A VLAN only changes its link status to down when all switch ports in the VLAN are down. If the switch connects to a peer through a port in a VLAN, and the VLAN also contains other ports, the link to the peer may go down without changing the VLAN status, so BGP cannot tell that the link is down. This happens when the port connected to the peer goes down but unrelated ports in the VLAN are still up. Therefore, we recommend you only apply fast fallover to a peer reached through a VLAN when all ports in the VLAN connect to the peer.

How to Improve IBGP Scalability

The problem

If a BGP peer learns a route from an EBGP peer, and selects it as the best available route to the given destination network, it sends an update message advertising that route to all its IBGP and EBGP peers. However, if a peer learns a route from an IBGP peer, it does not send an advertisement to its other IBGP peers. This policy requires that all BGP speakers within an autonomous system be fully meshed—each internal speaker must be connected to every other internal speaker. As a result, the scalability of a BGP autonomous system is in the order of n^2 (n speakers require $n(n-1)/2$ peer sessions).

The solution: route reflection

BGP Route Reflection improves the scalability of the AS, by giving specific IBGP peers the authority to advertise IBGP-learned routes to a predefined subset of their IBGP peers. Route Reflection is defined in RFC 2796 "BGP Route Reflection—An Alternative to Full Mesh IBGP". As shown in Figure 5-13, an AS using route reflection consists of at least one switch that advertises IBGP-learned routes, called a *Route Reflector* (RR), and that switch's IBGP peers. Each peer is one of the following types:

- *Client Peer* (CP)

  Client peers maintain IBGP peer sessions only with one or more of the RRs of their AS. CPs rely on an RR to advertise routes that they originate to the other members of the AS. When an RR receives a route from a client peer, the RR reflects the route to all its peers, both client and non-client.

- *Non-Client Peer* (NCP)

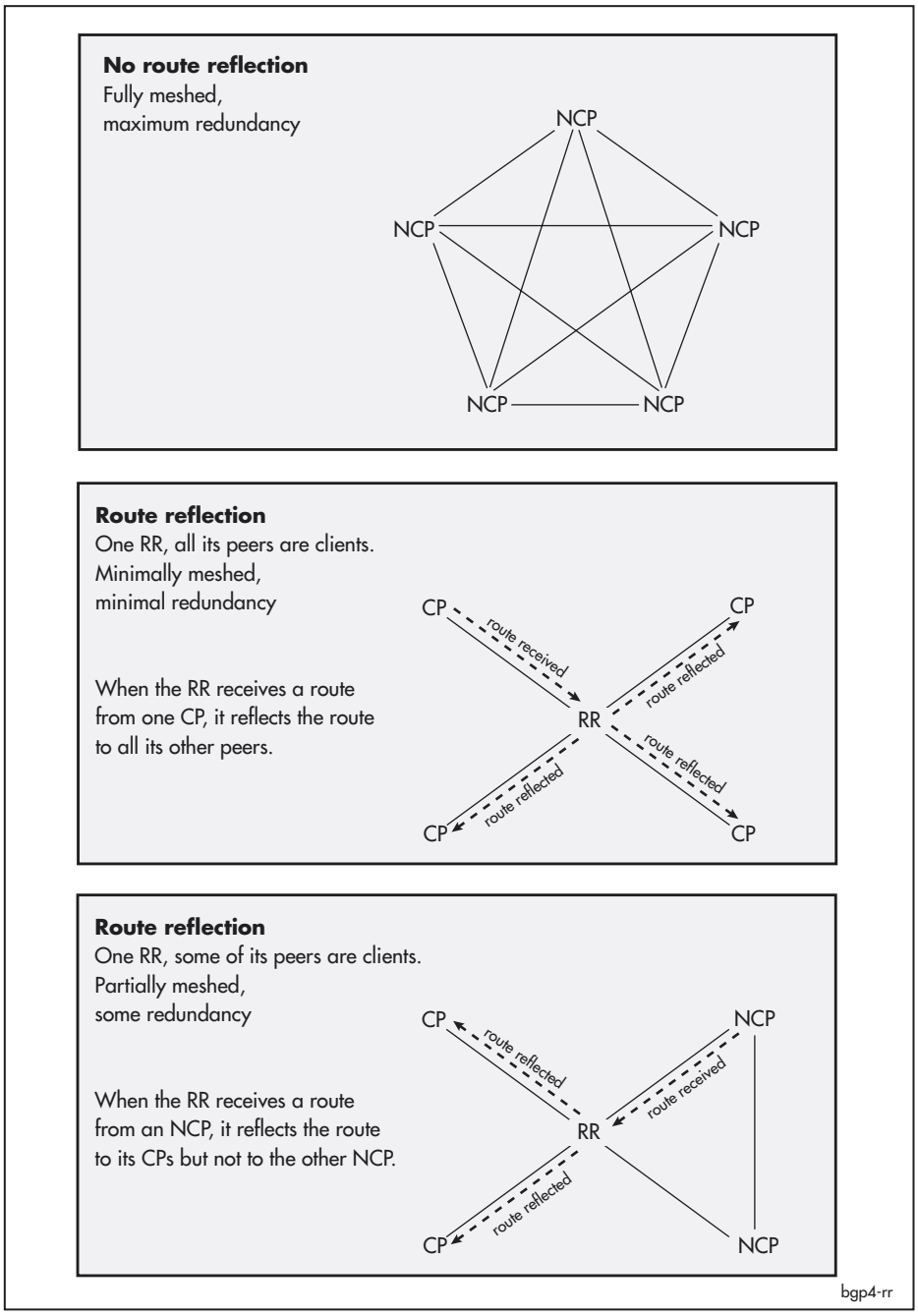
  Non-client peers maintain peer sessions with both the RRs of their AS, and all other non-client peers in the AS. NCPs only rely on an RR to advertise routes to the RR's client peers. When an RR receives a route from a non-client peer, the RR reflects the route only to its client peers, not to any other non-client peers it has.

  This means that client peers do not have to be connected to each other, but non-client peers do.

Note that route reflection does not affect the route selection process; it simply determines which peers the selected routes are advertised to.

Tip You can also use BGP confederations instead of full-meshing or route reflection. However, route reflection has the advantage that only the BGP hosts that perform the reflection need to understand route reflection. All hosts in a confederation need to be confederation-aware.

Figure 5-13: An IBGP AS of 5 switches, with and without route reflection



How to configure route reflection

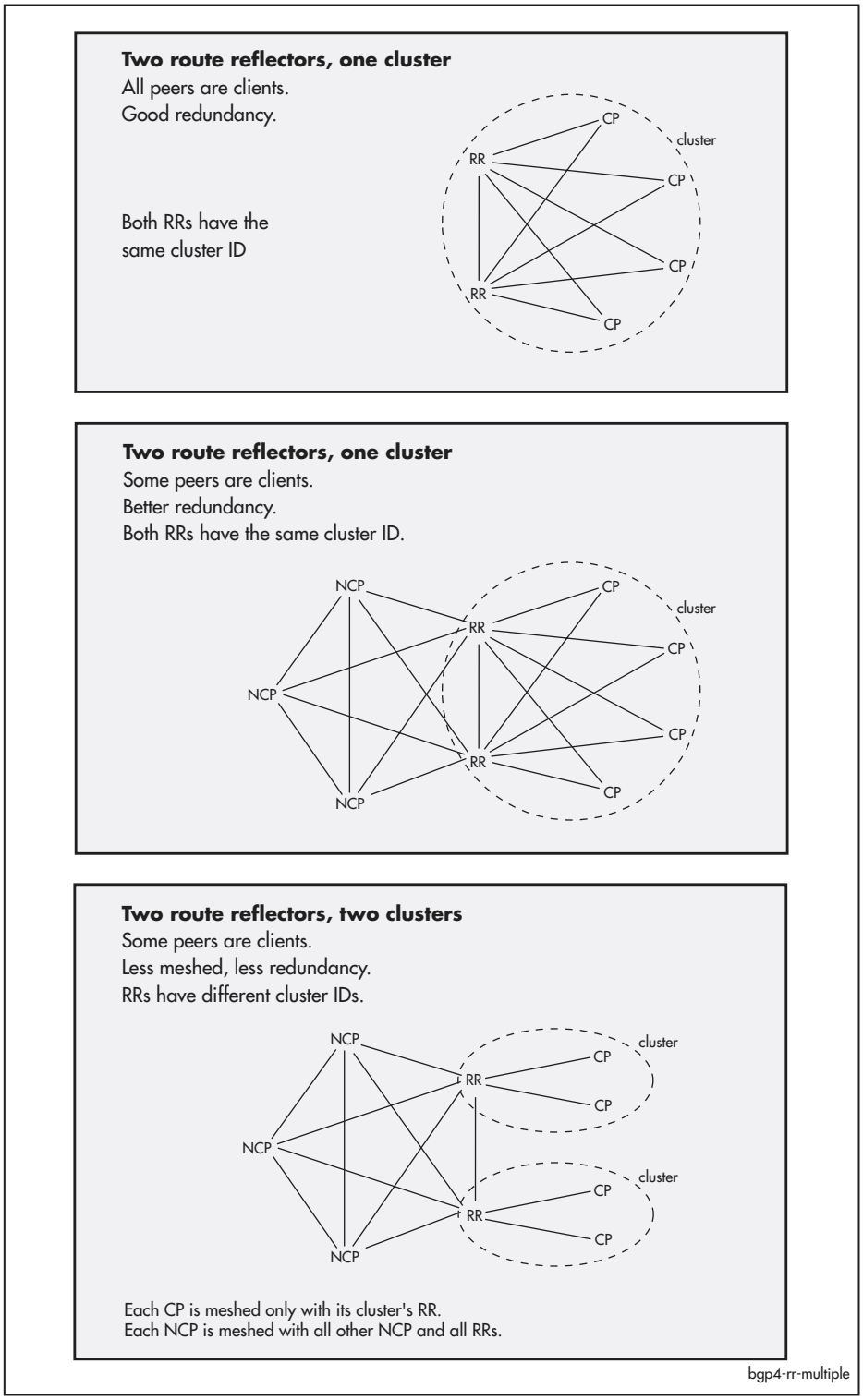
To configure route reflection, follow this procedure:

Step	Command	Action
1	—	Determine which switch in your AS will be the RR and which switches will be client peers of the RR.
2	—	Connect the switches in your IBGP AS together so that each peer is directly connected to the RR. Non-client peers must also be connected to each other—any non-client part of the AS must be fully meshed, because NCPs do not re-advertise routes amongst themselves.
3	**add bgp peer**=*ipadd* remoteas=*asn* client={no|yes} [*other-options*] or **set bgp peer**=*ipadd* client={no|yes} [*other-options*]	On the switch that is the RR, configure peer relationships to each of the other switches, specifying whether each peer is a client or non-client. The default for the **client** parameter is **no**, making the peer a non-client peer. The switch is an RR if it has at least one client peer.
4	**add bgp peer**=*ipadd* remoteas=*asn* [client=no] [*other-options*] or **set bgp peer**=*ipadd* [client=no] [*other-options*]	On the other switches, configure peer relationships to the route reflector, and to any other peers. Note that the RR is a non-client peer of its clients.

RR redundancy

Although a full-mesh AS suffers from poor scalability, it has the advantage of being extremely robust. In contrast, an AS that employs a single RR to serve a large number of clients is relatively vulnerable to congestion or loss of connectivity, because the RR plays a critical role in the operation of the AS. As shown in Figure 5-13 on page 5-43, you can reduce the vulnerability by making parts of the AS fully meshed. Alternatively, you can have multiple RRs within an AS by configuring one or more *clusters* (Figure 5-14 on page 5-45).

Figure 5-14: Multiple RRs in an AS



When there are multiple RRs in an AS, routing information loops become possible. Route reflection uses two BGP attributes to detect and prevent loops:

Attribute	Length	Type code	Definition and use	User-set
Originator_ID	4-byte	9	An attribute that identifies the first local AS member to learn the route. When the switch is an RR and receives a new route from an IBGP peer, it adds the originator_ID attribute if one was not already present. The switch uses the router ID of the IBGP peer that received the given route from an EBGP peer. If a CP or NCP receives an update that contains its own router ID as the originator ID, it ignores the update.	No
Cluster_list	variable	10	A list of 4-byte cluster_ID values that together represent the reflection path of the given route through the local AS. When an RR reflects a route, it adds its cluster_ID to the cluster_list. If a CP or NCP receives an update that contains its router ID as the originator, it ignores the update instead of reflecting it. This prevents routing information loops.	Cluster_ID is configurable.

CPs and NCPs forward these attributes unchanged within their AS. The switch removes the attributes from updates that are destined for EBGP peers.

How to configure multiple RRs

To configure multiple RRs in an AS, follow the procedure:

Step	Command	Action
1	—	Partition the AS into clusters. Each cluster has at least one RR and at least one CP.
2	**set bgp** cluster=*ipadd* [*other-options*]	Give all RRs in each cluster the same CLUSTER_ID so they can avoid routing information loops. Use the router ID of one of the RRs as the CLUSTER_ID.
3	**add bgp peer**=*ipadd* remoteas=*asn* client={no|yes} [*other-options*] or **set bgp peer**=*ipadd* client={no|yes} [*other-options*]	Configure the required peer relationships on all switches in the AS (see "How to configure route reflection" on page 5-44). Configure the RRs as peers of each other so each RR can reflect the other RRs' routes.

How to Handle Spikes in Memory Use

The problem

While BGP is running, other software modules may cause a spike, or surge, of system memory utilisation for brief periods of time.

The solution: BGP backoff

BGP backoff enables BGP to elegantly handle low system memory situations. When memory is heavily utilised, BGP backs off and delays its processing until system memory is more abundant.

The backoff utility allows other processes access to the memory resources they need, without actually shutting BGP down unless it determines that BGP has backed off for a prolonged period of time. By default, BGP delays its processing for 10 seconds if system memory utilisation reaches 95%.

How to configure BGP backoff

To change BGP system memory backoff settings, use the command:

```
set bgp backoff[=0..100] [consecutive=1..20]
    [multiplier=0..1000] [basetime=0..100]
    [totallimit=0..1000] [step=0..1000]
```

This command provides the following configuration options:

- percentage limit of total system memory utilisation that causes BGP to back off—the **backoff** parameter
- time that BGP backs off for—a combination of the **step**, **basetime** and **multiplier** parameters (see "How long BGP backs off" below)
- total number of backoffs before all BGP peers are disabled—the **totallimit** parameter
- total consecutive number of backoffs before all BGP peers are disabled—the **consecutive** parameter.

How long BGP backs off

The backoff time is recalculated after a given number of backoffs. This is termed a *step*. The first backoff time is calculated as:

```
base time x multiplier/100
```

The backoff time is recalculated after each step based on the current backoff time:

```
current backoff time x multiplier/100
```

The value is rounded down to the nearest second (unless it is less than 1 second, in which case it is set to 1 second).

For example, a base time of 60 seconds with a multiplier of 110 increases the timeout by 10 percent every time the backoff time is recalculated. Thus, a step value of 2 and multiplier of 110 results in the following numbers:

Backoff Iteration	Time to Backoff (secs)
0	60
1	60
2	66
3	66
4	72
5	72
6	79
7	79

A multiplier of less than 100 percent gives the effect of a decay mechanism, and a multiplier of greater than 100 percent gives the effect of an accumulative mechanism.

Consecutive backoffs

If BGP gets to the end of the backoff period and system memory is still heavily utilised, BGP will immediately back off again without performing any processing. Such backoffs are called *consecutive backoffs*. By default, the number of consecutive backoffs is limited to 20. After BGP reaches this limit, the switch considers that BGP is irrecoverable and disables all peers. You can change the limit.

The switch counts the number of consecutive backoffs. It resets the count to zero whenever BGP is able to perform some processing after a backoff.

How to Stop BGP from Overloading System Memory

BGP memory accounting limits BGP's use of total system memory, to 95% of the total memory by default. The switch disables BGP if it uses more than this percentage of system memory. The switch shuts down BGP peers, and therefore drops all routes learnt from those peers.

To change the memory limit, use the command:

```
set bgp memlimit[=0..100]
```

To see the current limit and usage, use the command:

```
show bgp memlimit
```

To see detailed technical information about memory usage, use the command:

```
show bgp memlimit scan
```

How to Avoid Leaking Private AS Numbers into Global BGP Tables

AS numbers are two bytes in length, so range from 1 to 65535. Of this value range, the AS numbers 1 to 64511 are globally unique and are assigned by InterNIC. The remaining value range from 64512 to 65535 is reserved for AS numbers that are private. These numbers are unique only within the scope of a given administrative domain.

Because private AS numbers are not globally unique, they should not be leaked to global BGP routing tables, in which context they become ambiguous. To prevent private AS numbers from crossing administrative boundaries, the switch supports the stripping of private AS numbers from the AS Path attributes of outgoing update messages. You can configure this on a per-peer or per-template basis. It is disabled by default. To configure a peer, use one of the commands:

```
add bgp peer=ipadd  remoteas=asn privateasfilter={yes|no}
    [other-options]
```

```
set bgp peer=ipadd  privateasfilter={yes|no} [other-options]
```

To configure a peer template, use one of the commands:

```
add bgp peertemplate=1..30 privateasfilter={yes|no}
    [other-options]

set bgp peertemplate=1..30 privateasfilter={yes|no}
    [other-options]
```

How to Control Import of Static Routes

You can control which static routes you import into BGP by *tagging* routes with an identification number.

Table 5-13: Procedure for importing particular static routes

Step	Command	Action
1	**add ip route**=*ipadd* interface=*interface* nexthop=*ipadd* tag=1..65535 [*other-options*] or **set ip route**=*ipadd* interface=*interface* mask=*mask* nexthop=*ipadd* tag=1..65535 [*other-options*]	Specify a number to tag each static route that you want to import. You can also tag routes you specifically want to exclude.
2	**show ip route**	Check the number that the route is tagged with.
3	**add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match tag=1..65535	Create a route map with entries that match the tagged routes.
4	**add bgp import**=static routemap=*routemap* **add bgp network**=*prefix*[/0..32] [mask=*mask*] routemap=*routemap*	Use the route map when importing routes into BGP.

Availability You can only use a route map that matches on **tag** when you use the **add bgp network** and **add bgp import** commands to import static routes from IP to BGP. Tagging does not filter routes that are sent to BGP peers, and does not match update messages that are received from BGP peers.

How to Set the IP Address By Which the Switch Identifies Itself

When the switch is acting as a BGP speaker, it uses an IP address to identify itself to its peers in these situations:

- when establishing the TCP session and sending TCP messages
- in the *open* message it sends at the beginning of the session
- when it considers itself to be the next hop for a route that it is advertising to its peers.

Address selection rules

The address the switch uses in each of these situations depends on the situation and whether you have configured a router ID or a local interface address. The rules for each situation are:

1. TCP session source address

   If a local IP address has been set for the peer, use it. Otherwise allow TCP to select a source IP address, which it will do based on the outgoing interface.

2. BGP Identifier in *open* message

   If the router ID has been set, use it. Otherwise, if a local IP address has been set for the peer, use that. If neither has been set, use the highest IP address configured on any of the switch's interfaces.

3. Next hop address

   If the switch learned the route from an IBGP peer, use the learned next hop address—the next hop that the IBGP peer supplied for the route.

   If the switch learned the route from an EBGP peer and the learned next hop is in the same subnet as the switch, use the learned next hop.

   If the switch learned the route from an EBGP peer and the learned next hop is in a different subnet to the switch, then:

   - if a local IP address has been set for the peer to which the switch is sending the update, use it
   - otherwise, if the switch has an IP route to that network, use the IP address of the interface via which the route reaches that network
   - otherwise, use the IP address of the interface via which the switch reaches the peer to which it is sending the update

How to configure router ID

To configure a router ID, use the command:

```
set bgp routerid=ipadd [other-options...]
```

How to configure local interface

To configure a local interface, first create the local interface and give it an IP address by using the command:

```
add ip local=1..15 ipaddress=ipadd [other-options...]
```

Then apply the local interface to the BGP peer by using one of the commands:

```
add bgp peer=ipadd remoteas=1..65534 local=1..15
   [other-options...]
```

```
set bgp peer=ipadd local=1..15 [other-options...]
```

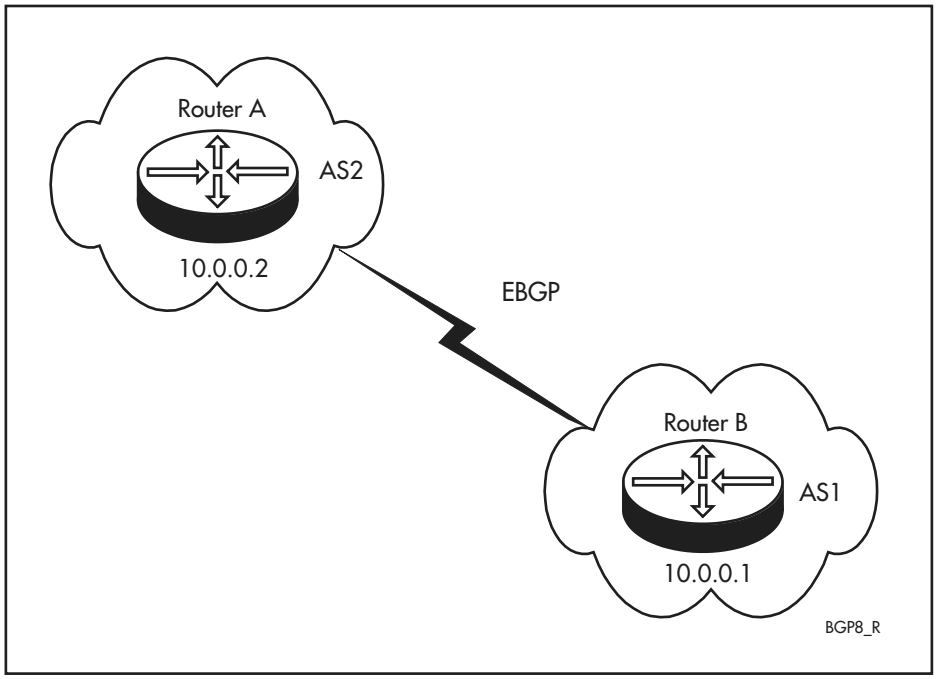
Configuration Examples

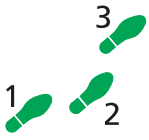
The following examples illustrate the steps required to configure BGP-4 on the switch. The first example shows a simple configuration without filtering, the other examples illustrate configurations using filtering parameters.

Example One

Switch A has been configured with IP address 10.0.0.2, AS number 2, and is to establish a BGP session with a peer, (Switch B), configured with IP address 10.0.0.1, AS number 1. This example assumes that the IP address has already been configured and the two peers can ping each other. Figure 5-15 on page 5-51 illustrates a simple BGP-4 configuration.

Figure 5-15: Example of a simple BGP-4 configuration





To configure Switch A and Switch B as peers.

1. **Set the AS number**

   To set the autonomous system number for Switch A, use the command:

   ```
   set ip autonomous=2
   ```

2. **Set the local IP address**

   To set the configuration of Switch A's local IP interface, use the command:

   ```
   set ip local ip=10.0.0.2
   ```

   The local IP interface is a virtual interface that represents the IP routing module itself. The interface can be assigned an IP address, which can then be used as the source address of IP packets generated internally by IP protocols such as RIP, OSPF, PING, and NTP.

3. **Add Switch B as a peer.**

   To add Switch B as a peer to Switch A, use the command:

   ```
   add bgp peer=10.0.0.1 remoteas=1
   ```

There is a limit of 64 configurable peers.

The peer is now configured, but not yet able to establish a connection.

4. Establish a connection between Switch A and Switch B.

To enable Switch B so that a connection can be established, use the command:

```
enable bgp peer=10.0.0.1
```

Switch A can now attempt to establish a connection with Switch B.

5. Check that the connection has been successfully established.

To verify that the connection is successfully established, use the command:

```
show bgp peer
```

This produces the following output.

```
BGP peer entries

Peer               State     AS      InMsg         OutMsg
-------------------------------------------------------------
-
10.0.0.1           Estab     1       23456         3245
```

6. Show the detailed output for a particular switch.

To see detailed information about the status of Switch B, use the command:

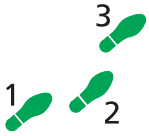
```
show bgp peer=10.0.0.1
```

This produces the following output.

```
Peer ................ 10.0.0.1
Description ......... -
State ............... Established
Remote AS ........... 1
Connect Retry ....... 120s
Hold time ........... 90s (actual 0s)
Keep alive .......... 30s (actual 0s - no KEEPALIVES)
Min AS originated ... 15
Min route advert .... 30
Filtering
  In filter ......... -
  In path filter .... -
  In route map ...... -
  Out filter ........ -
  Out path filter ... -
  Out route map ..... -
Max prefix .......... OFF
External hops ....... 1 (EBGP multihop disabled
Next hop self ....... No
Send community ...... No
Messages In/Out ..... 23456/3245
Debugging ........... -
Device .............. -
```

Example Two

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **inpathfilter** filtering parameter. The **inpathfilter** filters received BGP update messages based upon their AS path attributes. The filter is defined on a per peer basis.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that has all received update messages that have originated from AS 300 filtered out (that is, the originating AS is at the end of the AS list)

1. **Add an AS path list entry.**

```
add ip aspathlist=1 entry=1 exclude="300$"
```

2. **Set up the BGP peer.**

```
add bgp peer=10.0.0.1 remoteas=1 inpathfilter=1
```

All update messages sent to this peer are included by default.

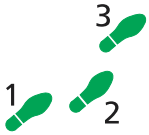
3. **An extra step to exclude routes that did not originate from AS 300 but went through AS 200 would be to use the command:**

```
add ip aspathlist=1 entry=2 exclude="200".
```

If an update message has AS 200 in it, and it originated from AS 300, then the first entry in the **aspathlist** is matched and the second entry match is not attempted.

Example Three

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **outpathfilter** filtering parameter. Using **outpathfilter** filters out BGP update messages being transmitted based upon their AS path attributes. The filter is defined on a per peer basis.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that does not have any routes that have originated from AS550 advertised to it

1. **Add an AS path list entry.**

```
add ip aspathlist=2 entry=1 exclude="550$"
```

2. **Set up the BGP peer.**

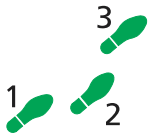
```
add bgp peer=10.0.0.1 remoteas=1 outpathfilter=1
```

All other update messages received from this peer are included by default.

Example Four

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **outroutemap** filtering parameter. The **outroutemap** filter is applied after all other filters have acted on an update message. The **outroutemap** filter is applied to an update message being transmitted, and filters it based on its AS path attribute if its match clause specifies an **aspathlist**.

The advantage of routemap filters over path filters is that they can be used to modify the attributes of a received BGP update message, whereas the path filters only includes or excludes messages.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that does not have any routes that have passed through AS550 advertised to it

1. **Add an AS path list entry.**

```
add ip aspathlist=2 entry=1 include="550"
```

2. **Exclude routes that originated from AS550.**

```
add ip routemap=as550 entry=1 match aspathlist=2
   action=exclude
```

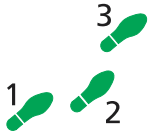
The **aspathfilter** has an **include** so that a match is made for the routemap's **entry**, and the routemap's **entry** action is taken. If the **aspathlist entry** was an **exclude**, then an AS Path that matched this line from the AS Path filter would stop processing the AS Path filter, would not match a Routemap Entry, and would move to the next routemap entry without any of the current routemap entry's **action/set** having been applied.

3. **Set up the BGP peer.**

```
add bgp peer=10.0.0.1 remoteas=1 outroutemap=as550
```

Example Five

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **inroutemap** filtering parameter. The **inroutemap** filter is applied after all other filters have acted on an update message. The **inroutemap** filter is applied to a received update message, and filters it based on its AS path attribute if its match clause specifies an **aspathlist**.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that have all received update messages that have originated from AS 300 filtered out (that is, the originating AS is at the end of the AS list)

1. **Add an AS path list entry.**

```
add ip aspathlist=1 entry=1 include="300$"
```

2. **Filter out messages from AS300**

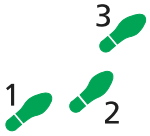
```
add ip routemap=as300 entry=1 match aspathlist=1
   action=exclude
```

3. **Set up the BGP peer.**

```
add bgp peer=10.0.0.1 remoteas=1 inroutemap=as300
```

Example Six

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **infilter** filtering parameter as Switch B advertises a route to network 101.0.0.0, but there is a preferred route (because of an arrangement with the network) to use.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that accepts all routes learned, except routes to network 101.0.0.0/8

1. **Add a filter entry to exclude a network.**

```
add ip filt=300 entry=1 action=exclude source=101.0.0.0
   smask=255.0.0.0
```

2. **Add a filter entry to include all routes.**

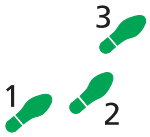
```
add ip filt=300 entry=2 action=include source=0.0.0.0
   smask=0.0.0.0
```

3. **Set up the INFILTER parameter.**

```
add bgp peer=10.0.0.1 remoteas=1 infilter=300
```

Example Seven

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration where the peer is one that does not want to advertise a route to network 102.0.0.0 to use by using the **outfilter** filtering parameter.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that sends all routes used, except routes to network 102.0.0.0/8

1. **Add a filter entry to exclude a network.**

```
add ip filt=301 entry=1 action=exclude source=102.0.0.0
   smask=255.0.0.0
```

2. **Add a filter entry to include all routes.**

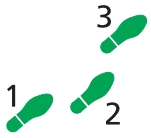
```
add ip filt=301 entry=2 action=include source=0.0.0.0
   smask=0.0.0.0
```

3. **Set up the INFILTER parameter.**

```
add bgp peer=10.0.0.1 remoteas=1 outfilter=301
```

Example Eight

Switch A has been configured with IP address 10.0.0.2, AS number 2, Switch B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration filtering on the basis of the community attribute, filtering on inbound and outbound routes.



To set the community attributes.

1. **Create routemaps that set the community attribute.**

   Calculate your community number using the following equation: (AS number x 65536)+1 = Community 1. For example, the AS number=2, so Community 1 is (2 x 65536)+1 = 131073.

```
add ip routemap=map0 entry=1 set community=131073
add ip routemap=map1 entry=1 set community=131074
add ip routemap=map2 entry=1 set community=131075
add ip routemap=map3 entry=1 set community=131076
add ip routemap=map4 entry=1 set community=131077
add ip routemap=map5 entry=1 set community=131078
add ip routemap=map6 entry=1 set community=131079
```

2. **Associate the routemaps with subnets, which can then be applied to the routes as they are added to the BGP routing tables.**

```
add bgp net=192.168.0.0/24 routemap=map0
add bgp net=192.168.1.0/24 routemap=map1
add bgp net=192.168.2.0/24 routemap=map2
add bgp net=192.168.3.0/24 routemap=map3
add bgp net=192.168.4.0/24 routemap=map4
add bgp net=192.168.5.0/24 routemap=map5
add bgp net=192.168.6.0/24 routemap=map6
add bgp net=192.168.7.0/24 routemap=map6
add bgp net=192.168.8.0/24 routemap=map6
add bgp net=192.168.9.0/24 routemap=map6
add bgp net=192.168.10.0/24 routemap=map6
```

   Note that the community attribute of the last five routes are set to the same value (16). Any routemaps not included may be used for other BGP peers.

3. **Filter the inbound routes on Switch B.**

```
add ip community=1 entry=1 include=16
add ip community=1 entry=2 exclude=internet
```

   This builds a community list consisting of those routes with the community attribute value set to 16 and exclude all other routes.

4. **Associate the community filter with a routemap.**

```
add ip routemap=mapin entry=1 match communitylist=1
add ip routemap=mapin entry=2 action=exclude
```

5. **Apply the routemap to the BGP peer.**

```
set bgp peer=10.0.0.2 sendcommunity=yes inr=mapin
```

Command Reference

This section describes the commands available on the switch to enable, configure, control and monitor BGP, including the commands available to configure IP to use BGP.

The shortest valid command is denoted by capital letters in the Syntax section.

add bgp aggregate

Syntax

```
ADD BGP AGGregate=prefix[/0..32] [MASK=mask]
    [SUMmary={NO|YES}] [ROUTEMap=routemap]
```

Description

This command adds an aggregate entry to BGP. When a peer advertises a route that is a subset of the entry's prefix, the switch adds the aggregate entry to its database, as well as the entry for the more specific route. This can increase the efficiency of BGP, by allowing the switch to process and advertise a single route, instead of a large number of more specific subnets.

Note that the switch does not use the aggregate route for IP routing. The switch only uses the aggregate to determine which routes to advertise.

The switch does not add the aggregate entry to its database until it receives an advertisement of a more specific subnet.

The switch advertises the aggregate route as coming from the switch's autonomous system, and sets the aggregate's atomic_aggregate attribute.

Parameter	Description
AGGregate	The network prefix to be used for this aggregate entry. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the switch uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the aggregate entry. This parameter is provided for compatibility with other switch commands that specify an IP address and mask; we recommend that you instead specify the mask in the **aggregate** parameter. If you specify a mask in this parameter and the **aggregate** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network

Parameter	Description	
SUMmary	Whether the switch advertises only the aggregate route, or also the more specific routes that make up the aggregate. Default: **no**	
	no	The switch advertises the more specific routes that make up the aggregate.
	yes	The switch only advertises the aggregate route. Note that unadvertised routes are still displayed in the output of the **show bgp route** command, but are marked with an "s".
ROUTEMap	The route map used to filter the more specific routes that make up the aggregate, or to set attributes for the aggregate route. The *routemap* is the name of the appropriate pre-existing map. Default: no route map (routes are not filtered and attributes are not set)	
Tip The shortest string you can enter is shown in capital letters.		

Examples

To add an aggregate entry for the network 198.168.0.0, with a mask of 255.255.0.0, use the command:

```
add bgp agg=192.168.0.0/16
```

As soon as the switch learns a more specific route, such as 192.168.1.0/24, BGP also adds an entry for 192.168.0.0/16 to the routing table.

To add an aggregate entry for the network 192.168.8.0/21 and use route map *agg_map*, use the command:

```
add bgp agg=192.168.8.0/21 routem=agg_map
```

Related Commands

delete bgp aggregate
set bgp aggregate
show bgp aggregate
show bgp route

add bgp confederationpeer

Syntax

```
ADD BGP CONFEDerationpeer=1..65534
```

Description

This command adds an Autonomous System to the AS confederation to which this switch belongs. An AS confederation is a group of Autonomous Systems which communicate between themselves using confederation BGP, and communicate to Autonomous Systems outside the confederation as if they were a single Autonomous System. For more information about AS confederations, see "AS Confederations" .

The **confederationpeer** parameter specifies the number of an Autonomous System that is to be treated as one of the Autonomous Systems in the confederation. This number cannot be the same as this switch's AS number, or this switch's confederation ID. The specified AS number should not already have been added with this command.

A switch need not be configured with all of the AS numbers in the AS confederation, but only those with which it is to have peer relationships. Similarly, the confederation ID for the switch only has to be configured on switches that are to have peer relationships with BGP switches outside the confederation.

When you create the peer relationship by using the **add bgp peer** command on page 5-62, specify the peer's confederation ID in the **remoteas** parameter.

Examples

To set up a confederation with AS numbers 65502, 65503 and 65504, whose external AS number is 1234, and with this switch in AS 65501, use the commands:

```
set ip au=1234
set bgp conf=65501
add bgp confed=65502
add bgp confed=65503
add bgp confed=65504
```

Related Commands

delete bgp confederationpeer
set bgp
set ip autonomous
show bgp confederation

add bgp import

Syntax

```
ADD BGP IMPort={INTerface|OSPF|RIP|STAtic}
    [ROUTEMap=routemap]
```

Description

This command adds an import entry to BGP. This instructs BGP to import routes from a given route source into the BGP route table. Optionally, you can specify a route map to allow filtering of routes and setting of BGP attributes.

When BGP imports routes from a protocol, it only imports routes which are the best routes to their destination networks. BGP determines that a route is the best route if:

- the route goes over an active interface
- that interface does not have an infinite or unreachable route
- the route has a higher routing preference metric than other candidate routes.

Parameter	Description	
IMPort	The source of routing information for the routes that are to be imported into BGP. Default: no default	
	INTerface	Imports interface routes.
	OSPF	Imports OSPF routes.
	RIP	Imports RIP or RIP2 routes.
	STAtic	Imports statically configured routes.
ROUTEMap	The route map used to filter the routes imported into BGP and to set attributes for the routes as advertised by BGP. The *routemap* is the name of the appropriate pre-existing map. The route map can **match** on origin, next hop, prefix list or tag, and can use any of the **set** parameters. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To import OSPF routes into BGP and use the route map *ospf_bgp_map* to filter and set attributes, use the command:

```
add bgp imp=ospf routem=ospf_bgp_map
```

Related Commands

add ip routemap
delete bgp import
set bgp import
show bgp import

add bgp network

Syntax

```
ADD BGP NETwork=prefix[/0..32] [MASK=mask]
    [ROUTEMap=routemap]
```

Description

This command adds a network to the list of networks that the switch can advertise to remote BGP peers. You can also optionally specify a route map to filter the networks and/or to set attributes on the routes sent.

Statically defining a BGP network with this command does not cause the switch to advertise the network immediately—the switch does not know the next hop for the network. Defining a BGP network informs BGP that if the switch learns a route to the network by a non-BGP means, for example statically or from OSPF, then BGP should advertise the network.

Parameter	Description
NETwork	The network to add to the list of networks that can be advertised. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the switch uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the network. This parameter is provided for compatibility with other switch commands that specify an IP address and mask; we recommend that you instead specify the mask in the **network** parameter. If you specify a mask in this parameter and the **network** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network
ROUTEMap	The route map used to filter this network and to set attributes on routes that are sent in the BGP update messages that advertise this route. The *routemap* is the name of the appropriate pre-existing map. The route map can **match** on origin, next hop, prefix list or tag, and can use any of the **set** parameters. Default: no route map (routes are not filtered and attributes are not set)

Tip The shortest string you can enter is shown in capital letters.

Examples

To add the network 192.169.2.0 to the list of networks advertised by BGP and to use the route map "normal", use the command:

```
add bgp net=192.169.2.0/24 routem=normal
```

Related Commands

delete bgp network
show bgp network

add bgp peer

Syntax

```
ADD BGP PEer=ipadd REMoteas=1..65534
    [AUthentication={MD5|NONE}] [CLIEnt={NO|YES}]
    [CONnectretry={DEFault|0..4294967295}]
    [DESCription=description] [EHOps={DEFault|1..255}]
    [FASTFallover={NO|YES}] [HOLdtime={DEFault|0|3..65535}]
    [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
    [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
    [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
    [MAXPREFIXAction={Terminate|Warning}]
    [MINAsoriginated={DEFault|0..3600}]
    [MINRouteadvert={DEFault|0..3600}]
    [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
    [OUTPathfilter={NONE|1..99}] [OUTRoutemap=routemap]
    [PASSword=password] [PRIVateasfilter={NO|YES}]
    [SENdcommunity={NO|YES}]

ADD BGP PEer=ipadd POLICYTemplate=1..30 REMoteas=1..65534
    [AUthentication={MD5|NONE}] [DESCription=description]
    [EHOps={DEFault|1..255}] [FASTFallover={NO|YES}]
    [PASSword=password]
```

Description

This command adds a BGP peer to the switch. This command adds the peer in the disabled state; the switch does not attempt to communicate with the peer until the **enable bgp peer** command on page 5-100 command is entered. This allows you to fully configure the peer entry before starting to communicate with it.

Parameter	Description	
PEer	The IP address of the new peer, in dotted decimal notation. This address should be the address that this switch uses when communicating with the peer; that is, the address of the interface on the peer that is closest to this switch. Default: no default	
REMoteas	The remote Autonomous System to which this peer belongs. If the remote AS number is the same as this switch's AS number, the peer is an internal BGP (IBGP) peer. If the remote AS number is different from this switch's AS number, the peer is an external BGP (EBGP) peer. If the remote AS numbers are different but the switches have the same confederation peer, the peer is a confederation BGP peer. The AS number is assigned by the IANA. Default: no default	
AUthentication	Whether to use MD5 authentication for the BGP peer. If you specify **md5**, you must also specify **password**. Default: **none**	
	MD5	An MD5 digest is added to every BGP packet sent over the TCP connection and is authenticated at the other end. If any part of the digest cannot be verified, the packet is dropped with no response sent.
	NONE	The BGP session is not authenticated.

Parameter	Description	
CLIEnt	Whether the peer is a client of the switch when the switch is a route reflector (RR). RRs selectively advertise routes they learn from their IBGP peers to their other IBGP peers. The switch is a route reflector if it has at least one client peer, so if **client=yes** for at least one of its peers. For more information about route reflection, and client and non-client peers, see "How to Improve IBGP Scalability" on page 5-42. Route reflection is valid for IBGP peers, so **client=yes** is valid only when the local ASN and the remote ASN are the same. Default: **no**	
	NO	The peer is a non-client peer of the RR. When the RR receives a route from a non-client peer, the RR reflects the route only to its client peers, not to any other non-client peers it has.
	YES	The peer is a client peer of the RR. When the RR receives a route from a client peer, the RR reflects the route to all its peers, both client and non-client.
CONnectretry	The time interval between attempts to establish a BGP connection to the peer, in seconds. Default: **120**	
	0	The switch does not repeat an attempt to establish a BGP connection.
	1..4294967295	The switch waits the specified number of seconds between attempts.
	DEFault	The switch waits 120 seconds between attempts.
DESCription	A description of the peer, which has no effect on its operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
EHOps	The number of hops put in the *TTL* (Time To Live) field of BGP messages for external BGP. Normally, EBGP requires that BGP peers be connected to a common network, which means they are separated by a single hop. Setting **ehops** to a value greater than 1 indicates that multihop EBGP is allowed. Default: **1**	
	1..255	The specified number of hops is put into the TTL field.
	DEFault	The number of hops put in the TTL field is 1.
FASTfallover	Whether fast fallover is enabled on the link to the peer. If fast fallover is enabled, the peer session is reset as soon as the interface that supports the session goes down. If fast fallover is disabled, the session is reset only when its keepalive timer expires. Default: **no** (fast fallover is disabled)	

Parameter	Description	
HOLdtime	The value in seconds that this switch proposes for the time interval between reception of keepalive and/or update messages from this peer. The actual hold time used on a peer connection is negotiated when the connection is opened, as the lower of the hold times proposed. Default: **90**	
	0	This switch proposes not to have a hold time on this BGP connection.
	3..65535	This switch proposes the specified number of seconds as hold time.
	DEFault	This switch proposes a hold time of 90 seconds.
INFilter	The IP routing filter that acts as a prefix filter to filter any prefixes advertised via incoming BGP update messages from this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command and create a filter with a number from 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
INPathfilter	The AS path list that filters the BGP update messages from this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
INRoutemap	The route map that filters and/or modifies prefixes from this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
KEEpalive	The time in seconds that this switch would prefer to leave between keepalive messages to this peer. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**	
	1..21845	This switch prefers the specified number of seconds as keepalive interval.
	DEFault	This switch prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the switch uses this address as the source for BGP packets it generates and sends to this BGP peer. For a description of when the switch uses the local interface, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50. Default: **none**	
MAXPREFIX	The maximum number of network prefixes that the switch expects to receive from this peer. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the switch expects to receive from this peer. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The switch logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The switch resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the switch to this peer, of routes that originate in the switch's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.

Parameter	Description	
MINRouteadvert	The minimum time in seconds between advertisements, from the switch to this peer, of routes that originate outside the switch's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this switch advertises to this peer that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the switch sends to this peer specify this switch as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter to filter the prefixes sent in BGP update messages to this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
OUTPathfilter	The AS path list that filters the BGP update messages sent to this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
OUTRoutemap	The route map that filters and/or modifies prefixes sent to this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
PASSword	The key used by the authentication algorithm. Two BGP peers can only communicate with each other if they have the same key. *password* is a character string from 1 to 80 characters long. All printable characters are valid except the question mark and double quotes. If *password* contains spaces, it must be in double quotes. Only valid if **authentication=md5** Default: no default	
POLICYTemplate	The ID number of the peer policy template that applies to this peer. The specified policy template must already exist. To create a template, use the **add bgp peertemplate** command on page 5-68. You can only specify **remoteas**, **description**, **authentication**, **password**, **fastfallover,** and **ehops** at the same time as **policytemplate**. The template provides all other configuration values.	
PRIVateasfilter	Whether private AS numbers (from 64512 to 65535) are stripped from the AS PATH attribute on update messages the switch sends to the peer. Default: **yes**, if **remoteas** is a public AS number; **no** if **remoteas** is a private AS number.	
SENdcommunity	Whether the switch includes the community attribute in update messages that it sends to this peer. Default: no	
	YES	The community attribute is set in update messages to this peer. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to this peer. To create a route map use the **add ip routemap** command on page 5-79.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peer.

Tip The shortest string you can enter is shown in capital letters.

Examples

To add a BGP peer whose IP address is 192.168.1.1 and whose AS number is 54321, use the command:

```
add bgp pe=192.168.1.1 rem=54321 desc="test remote bgp peer"
```

Related Commands

add ip aspathlist
add ip routemap
delete bgp peer
disable bgp peer
enable bgp peer
reset bgp peer
set bgp peer
show bgp peer

add bgp peertemplate

Syntax

```
ADD BGP PEERTemplate=1..30 [CLIEnt={NO|YES}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description]
   [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=routemap]
   [PRIVateasfilter={NO|YES}] [SENdcommunity={NO|YES}]
```

Description This command creates a template for use on BGP peers.

Parameter	Description	
PEERTemplate	The ID number of the template. Default: no default	
CLIEnt	Whether peers that use the template are clients of the switch if the switch is a route reflector (RR). RRs selectively advertise routes they learn from their IBGP peers to their other IBGP peers. The switch is a route reflector if it has at least one client peer, so if **client=yes** for at least one of its peers. For more information about route reflection, and client and non-client peers, see "How to Improve IBGP Scalability" on page 5-42. Route reflection is only valid for IBGP peers, so **client=yes** is only valid if the local ASN and the remote ASN are the same. Default: **no**	
	NO	Peers that use the template are non-client peers of the RR. When the RR receives a route from a non-client peer, the RR reflects the route only to its client peers, not to any other non-client peers it has.
	YES	Peers that use the template are client peers of the RR. When the RR receives a route from a client peer, the RR reflects the route to all its peers, both client and non-client.
CONnectretry	The time interval between attempts to establish a BGP connection to peers that use the template, in seconds. Default: **120**	
	0	The switch does not repeat an attempt to establish a BGP connection.
	1..4294967295	The switch waits the specified number of seconds between attempts.
	DEFault	The switch waits 120 seconds between attempts.

Parameter	Description	
DESCription	A description for the peers that use the template, which has no effect on their operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
HOLdtime	The value in seconds that this switch proposes for the time interval between reception of keepalive and/or update messages from peers that use the template. The actual hold time used on a peer connection is negotiated when the connection is opened, and is the lower of the hold times proposed. Default: **90**	
	0	This switch proposes not to have a hold time on this BGP connection.
	3..65535	This switch proposes the specified number of seconds as hold time.
	DEFault	This switch proposes a hold time of 90 seconds.
INFilter	The routing filter that acts as a prefix filter and filters any prefixes advertised via incoming BGP update messages from peers that use the template. You can use a prefix filter to exclude routes to particular networks from update messages. The filter must already exist. To create a filter use the **add ip filter** command. The filter number must be in the range 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
INPathfilter	The AS path list that filters the BGP update messages from peers that use the template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
INRoutemap	The route map that filters and/or modifies prefixes from peers that use the template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
KEEpalive	The time in seconds that this switch would prefer to leave between keepalive messages to peers that use the template. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**	
	1..21845	This switch prefers the specified number of seconds as keepalive interval.
	DEFault	This switch prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the switch uses this address as the source for BGP packets it generates and sends to peers that use this template. For a description of when the switch uses the local interface, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50. Default: **none**	
MAXPREFIX	The maximum number of network prefixes that the switch expects to receive from peers that use the template. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the switch expects to receive from peers that use the template. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The switch logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The switch resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the switch to peers that use the template, of routes that originate in the switch's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.

Parameter	Description	
MINRouteadvert	The minimum time in seconds between advertisements, from the switch to peers that use the template, of routes that originate outside the switch's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this switch advertises to peers that use the template that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the switch sends to peers that use this template specify this switch as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter and filters the prefixes sent in BGP update messages to peers that use this template. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
OUTPathfilter	The AS path list that filters the BGP update messages sent to peers that use this template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
OUTRoutemap	The route map that filters and/or modifies prefixes sent to peers that use this template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
PRIVateasfilter	Whether private AS numbers (from 64512 to 65535) are stripped from the AS PATH attribute on update messages the switch sends to peers that use this template. Default: **no**	
SENdcommunity	Whether the switch includes the community attribute in update messages that it sends to peers that use this template. Default: no	
	YES	The community attribute is set in update messages to peers that use this template. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to peers that use this template. To create a route map use the **add ip routemap** command on page 5-79.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peers that use this template.

Tip The shortest string you can enter is shown in capital letters.

Examples

To create a new peer policy template with a hold time of 30 seconds, and assign it to a peer, use the commands:

```
add bgp peert=1 hol=30
add bgp pe=192.168.1.0/24 policyt=1
```

Related Commands

add bgp peer
set bgp peer
set bgp peertemplate
show bgp peer

add ip aspathlist

Syntax

```
ADD IP ASPATHlist=1..99 [ENTry=1..4294967295]
   INCLude=aspath-reg-exp

ADD IP ASPATHlist=1..99 [ENTry=1..4294967295]
   EXCLude=aspath-reg-exp
```

Description

This command adds an entry to an AS path list, and creates the list if it does not already exist. You must specify the index number of the AS path list, and may also specify the position of the entry in the list.

When the switch searches through an AS path list, the first entry that causes a match stops the search, returning the result **include** or **exclude** depending on the type of entry. A totally empty AS path list is identical to an AS path list that matches all AS paths and is of type **include**. Any non-empty AS path list has an implicit entry at the end that matches all AS paths and is of type **exclude**. For more information about using AS path lists, see "How to Configure AS Path Filters" on page 5-27.

Parameter	Description
ASPATHlist	The ID number of the AS path list. You can create up to 99 AS path lists. Default: no default
ENTry	The desired position of the new entry in the AS path list once the entry has been added. Entries are numbered from 1 to the number of entries in the list. Default: The entry is added to the end of the list
INCLude	An AS path regular expression, which specifies the AS path values that this entry includes in this AS path list. When you use the AS path list in a route map or filter, the map or filter carries out its specified action on update messages with a matching AS path attribute value. Regular expressions are a list of one or more AS numbers, separated by spaces. To match from the first number in the list, start the expression with the ^ character. To match the last number, end with the $ character. If the expression contains spaces, surround it with double quotes. For more information about valid syntax, see Table 5-8 on page 5-28. For example: • **include="23334 45634 88988"** includes any path containing these numbers • **include="^23334 45634 88988$"** includes only that exact path • **include=^23334** includes any path that begins with 23334 Default: no default

Parameter	Description
EXCLude	An AS path regular expression, which specifies the AS path values that this entry excludes from this AS path list. When you use the AS path list in a route map or filter, the map or filter does not carry out its specified action on update messages with a matching AS path attribute value. Regular expressions are a list of one or more AS numbers, separated by spaces. To match from the first number in the list, start the expression with the ^ character. To match the last number, end with the $ character. If the expression contains spaces, surround it with double quotes. For more information about valid syntax, see Table 5-8 on page 5-28. For example: • **exclude="23334 45634 88988"** excludes any path containing these numbers • **exclude="^23334 45634 88988$"** excludes only that exact path • **exclude=23334$** excludes any path that ends with 23334 Default: no default

Tip The shortest string you can enter is shown in capital letters.

Examples

To add an entry to AS path list 1 that matches all AS paths and excludes them, use the command:

```
add ip aspath=1 excl=.*
```

To add an entry to AS path list 2 that matches an empty AS path and includes it, use the command:

```
add ip aspath=2 incl=^$
```

Related Commands

add ip routemap
delete ip aspathlist
show ip aspathlist

add ip communitylist

Syntax

```
ADD IP COMmunitylist=1..99 [ENTry=1..4294967295]
   INCLude={INTernet|NOExport|NOAdvertise|
   NOEXPORTSubconfed|AA:XX}[,...]

ADD IP COMmunitylist=1..99 [ENTry=1..4294967295]
   EXCLude={INTernet|NOExport|NOAdvertise|
   NOEXPORTSubconfed|AA:XX}[,...]
```

Description This command adds an entry to a community list, and creates the list if it does not already exist. You must specify the index number of the community list, and may also specify the position of the entry in the list.

Parameter	Description	
COMmunitylist	The ID number of the community list. You can create up to 99 lists. Default: no default	
ENTry	The desired position of the new entry in the community list once the entry has been added. Entries are numbered from 1 to the number of entries in the list. Default: The entry is added to the end of the list	
INCLude	A community name, community number, or comma-separated list of names and numbers, which specifies the communities that this entry includes in this community list. When you use the community list in a route map or filter, the map or filter carries out its specified action on update messages with a matching community attribute value. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	AA:*XX*	The number of a community. *AA* and *XX* are both integers in the range 0 to 65534. *AA* is the AS number. *XX* is a value chosen by the ASN administrator.

Parameter	Description	
EXCLude	A community name, community number, or comma-separated list of names and numbers, which specifies the communities that this entry excludes from this community list. When you use the community list in a route map or filter, the map or filter does not carry out its specified action on update messages with a matching community attribute value. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	AA:*XX*	The number of a community. *AA* and *XX* are both integers in the range 0 to 65534. *AA* is the AS number. *XX* is a value chosen by the ASN administrator.
Tip The shortest string you can enter is shown in capital letters.		

Examples To add an entry to community list 1 that matches communities attributes that contain the communities NOEXPORT and 70000 and excludes them, use the command:

```
add ip com=1 excl=noe 70000
```

Related Commands

add ip routemap
delete ip communitylist
show ip communitylist

add ip prefixlist

Syntax

```
ADD IP PREFIXList=name ENTry=1..65535
   [ACTion={MATch|NOMatch}] [MASklength=range]
   [PREfix=ipadd]
```

Description

This command adds a numbered *entry* to a prefix list. If the prefix list does not already exist, this command first creates it. You can create up to 400 prefix lists, with up to 1000 entries in each list.

Parameter	Description
PREFIXList	A name to identify the prefix list. A string 1 to 15 characters long. Valid characters are uppercase letters (A-Z), lowercase letters (a-z), digits (0-9) and the underscore character ("_"). If *name* contains spaces, it must be in double quotes. Default: no default
ENTry	An integer to specify the position of the new entry in the prefix list. When a BGP peer uses a prefix list, it checks the entries in order, starting with the lowest, until it finds a match. Therefore, give more specific entries lower numbers than general entries. If you leave gaps between entry numbers, you can add future entries between existing entries. Each prefix list has an implicit final entry that matches all addresses, with an action of **nomatch**. Default: no default
ACTion	Whether matching prefixes are included or excluded by the process that is using the prefix list. You can use multiple entries in a prefix list with actions of **match** and **nomatch** to build up a list of prefixes. Prefixes with **action=match** are included in the list. Then to use this list of prefixes, create a route map that matches it and apply the route map to a peer. The route map also has an **action** parameter, which determines whether the peer includes or excludes the prefixes in the list. Default: **match** MATch — The prefix list includes the prefix. NOMatch — The prefix list excludes the prefix.

Parameter	Description
MASklength	The range of prefix mask lengths matched by this entry in the prefix list. The *range* is either a single CIDR mask from 0 to 32, or two masks separated by a hyphen. These options are valid for setting the mask length: • as a mask length range (**masklength=*a*-*b*)**. For a route to match against this entry, its prefix mask length must be between *a* and *b* inclusive. *a* must be less than *b*. • as a single mask length (**masklength=*a***). For a route to match against this entry, its prefix mask length must be exactly *a*. • as an implicit mask length, by not specifying **masklength** (for example, **prefix=192.168.0.0**). For a route to match against this entry, its prefix mask length must correspond exactly to the mask for the class of the given address; in this example, 24. Default: The natural mask for the prefix, based on whether it is a class A, B, or C network
PREfix	The network address matched by this entry in the prefix list, specified in dotted decimal notation. If you do not specify a prefix, the switch sets it to 0.0.0.0. This is correct if you are matching all routes or the default route. Default: 0.0.0.0
Tip The shortest string you can enter is shown in capital letters.	

Examples

To match only routes from the 192.168.0.0/16 network, use the command:

```
add ip prefixlist=sample1 entry=1 action=match
   prefix=192.168.0.0 masklength=16
```

To match all routes in all 192.168.0.0 networks, except those in the 192.168.7.0 network, use the commands:

```
add ip prefixlist=sample2 entry=1 action=nomatch
   prefix=192.168.7.0 masklength=24-32
```

```
add ip prefixlist=sample2 entry=2 action=match
   prefix=192.168.0.0 masklength=16-32
```

To exclude the default route, use the command:

```
add ip prefixlist=sample3 entry=1 action=nomatch masklength=0
```

To include all routes, use the command:

```
add ip prefixlist=sample4 entry=1 action=match
   masklength=0-32
```

Related Commands

add ip routemap
delete ip prefixlist
set ip routemap
show ip prefixlist

add ip routemap

Syntax for an empty entry

```
ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}]
```

Syntax for a *match* clause

```
ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch ASPath=1..99

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch COMmunity=1..99
   [EXAct={NO|YES}]

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch NEXThop=ipadd

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch
   ORIGin={EGP|IGP|INCOmplete}

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch PREFIXList=name

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch TAG=1..65535
```

Syntax for a *set* clause

```
ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ASPath={1..65534[,...]}

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET
   COMmunity={NOExport|NOAdvertise|NOEXPORTSubconfed|
   AA:XX}[,...]} [ADD={NO|YES}]

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET BGPDampid=1..100

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET LOCalpref=0..4294967295

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET MED={0..4294967295|REMOVE}

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ORIGin={IGP|EGP|INCOmplete}
```

Description

This command adds a numbered *entry* to a route map, or adds a *clause* to an existing entry in a route map. If the route map does not already exist, this command first creates it.

Route maps are made up of a list of entries. Each entry contains:

- zero or one **match** clause, to determine which update messages the entry applies to. If an entry does not have a match clause, the effect is that it matches everything.
- one **action**, to determine whether matching update messages are included or excluded by the process that is using the route map (by default matching items are included).

- zero, one, or more **set** clauses. Most **set** clauses change the attributes of matching update messages. Each entry can have at most one **set** clause of a given type. For example, you can only set the MED once in an entry.

Parameters for both *match* and *set* clauses

Parameter	Description
ROUTEMap	The name of the route map to add the entry or clause to. The *routemap* is a character string 0 to 15 characters long. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character. Default: no default
ENTry	An integer to specify the position of the new entry in the route map. When a BGP peer uses a route map, it checks the entries in order, starting with the lowest, until it finds a match. If you leave gaps between entry numbers, you can add future entries between existing entries. Be careful when specifying the entry number. If you make an error in the number (for example, enter entry=11 instead of entry=1), the switch adds a new entry to the route map. Default: no default
ACtion	Whether matching prefixes or update messages are included or excluded by the process that is using the route map. The **action** parameter applies to the entire entry, but you can change it at the same time as you add a clause. The most recently entered value of this parameter applies to the entire entry. It is not meaningful to have **action=exclude** in an entry with a set clause. Default: the current setting. If there is no current setting, **include**
Tip The shortest string you can enter is shown in capital letters.	

Parameters for *match* clauses

Parameter	Description
MAtch	Adds a **match** clause to the entry, to determine which update messages the entry applies to. A route map entry can have zero or one **match** clauses. A entry without a **match** clause matches all update messages.
ASPath	The ID number of an AS path list. An update message matches the route map entry if its AS path attribute matches the AS path list. To configure an AS path list use the **add ip aspathlist** command on page 5-73. Default: no default
COMmunity	The ID number of a community list. An update message matches the route map entry if its community attribute matches the community list.To configure a community list use the **add ip communitylist** command on page 5-75. Default: no default

Parameter	Description	
EXAct	Whether the community attribute in an update message must precisely match the route map's community list. Only valid when you specify both **match** and **community**. Default: **no**	
	YES	An update message only matches the route map entry if its community attribute contains all the communities specified in the community list and only those communities.
	NO	An update message still matches the route map entry if its community attribute contains all the communities specified in the community list plus extra communities.
NEXThop	The IP address of the next node in the path to the route's destination, specified in dotted decimal notation. An update message matches the route map entry if its next_hop attribute matches this address. Default: no default	
ORIGin	An origin attribute value, which indicates BGP's source for the routes at their originating AS. An update message matches the route map entry if its origin attribute matches this value. Default: no default	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.
PREFIXList	The name of a prefix list. A route matches the route map entry if the prefix list contains that route. To create a list use the **add ip prefixlist** command on page 5-77. Default: no default	
TAG	A tag that identifies a particular static route. A route matches this route map entry if it has been tagged with this value by using the **tag** parameter of the **add ip route** command on page 13-102 of Chapter 13, Internet Protocol (IP). You can only use a route map that matches on **tag** when you use the **add bgp network** and **add bgp import** commands to import static routes from IP to BGP. You cannot use **tag** to filter routes that are sent to BGP peers or to match update messages that are received from BGP peers. Default: no default	
Tip The shortest string you can enter is shown in capital letters.		

Parameters for *set* clauses

Parameter	Description
SET	Adds a **set** clause to the entry, to modify an attribute in update messages that match the entry. A route map entry can have zero, one or more **set** clauses, but can only modify each attribute once. A entry without a **set** clause does not modify any attributes.
ASPath	A comma-separated list of 1 to 10 AS numbers. These numbers are added to the beginning of the update message's AS path attribute. Default: no default
COMmunity	A comma-separated list of 1 to 10 communities, identified by name or number. If the **add** parameter is **yes**, these communities are added to the update message's community attribute. If the **add** parameter is **no** (its default), these communities replace the update message's community attribute. Note that you must also set the peer's **sendcommunity** parameter to **yes** if you want the peer to include the community attribute in the update messages it sends. By default, peers do not include the community attribute in outgoing updates. Default: no default
INTernet	The community of routes that can be advertised to all BGP peers.
NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
NOAdvertise	The community of routes that must not be advertised to other BGP peers.
NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
AA:*XX*	The number of a community. *AA* and *XX* are both integers in the range 0 to 65534. *AA* is the AS number. *XX* is a value chosen by the ASN administrator.
ADD	Whether the list of communities specified by the **community** parameter is added to the community attribute, or replaces the community attribute. Only valid when you specify both **set** and **community**. Default: **no**
YES	The communities are added to the update message's community attribute.
NO	The communities replace the update message's community attribute.
BGPDampid	The BGP route flap damping ID that is given to matching routes. This is the same as the ID number of the parameter set that maintains that route's FoM upon it exhibiting instability. If the parameter set does not exist, the default parameter set is applied to matching routes. For more information about using route maps when configuring route flap damping, see "Damping routes on specific peers" . Default: no default

Parameter	Description	
LOCalpref	The metric to write into the update message's local_preference attribute. IBGP uses the local preference to determine which path it should use inside the AS to reach the advertised prefix. A lower metric indicates a preferred path. EBGP does not use this attribute. Default: no default	
MED	The metric to write into the update message's Multi_Exit_Discriminator attribute. EBGP uses the MED to determine the optimal path for reaching the advertised prefixes. A lower metric indicates a preferred path. IBGP does not use this attribute. Default: no default	
	0..4294967295	This value is written into the MED attribute of the matched update message.
	REMOVE	The MED attribute is removed from the matched update message.
ORIGin	The value to write into the update message's origin attribute. The origin indicates BGP's source for the routes at their originating AS. Default: no default	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.

Tip The shortest string you can enter is shown in capital letters.

Examples

To add a route map entry that sets the community attribute to 489816064 for all BGP routes, use the command:

```
add ip routem=set_comm ent=10 set com=489816064
```

This command creates the route map, adds an entry to it, and adds a set clause to the entry. No match clause is required because we wish to match all routes. To use this route map for routes being sent to BGP peer 192.168.1.1, use the command:

```
set bgp peer=192.168.1.1 outr=set_comm
```

Related Commands

delete ip routemap
set ip routemap
show ip routemap

create bgp damping parameterset

Syntax

```
CREate BGP DAMping PARameterset=1..100
   [DESCription=description]
   [SUPpression={DEFault|1..20000}]
   [REUse={DEFault|1..20000}] [HALflife={DEFault|1..45}]
   [MAXhold={DEFault|1..8}]
```

Description This command creates a parameter set for route flap damping.

If route flap damping is currently enabled as a whole, the new parameter set is enabled. If route flap damping is currently disabled, or only particular parameter sets are currently enabled, the new parameter set is disabled.

Parameter	Description	
PARameterset	A unique ID number to identify the parameter set. Default: no default	
DESCription	A description of the parameter set, which has no effect on its operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
SUPpression	A Figure of Merit (FoM) value, which indicates route stability. When a route's FoM exceeds this threshold, the route is suppressed. **Suppression** must be greater than or equal to **reuse**. If **suppression** is less than 1000, a route is suppressed when it becomes unreachable for the first time. Default: **2000**	
	1..20000	The route is suppressed once its FoM exceeds this value.
	DEFault	The route is suppressed once its FoM exceeds 2000.
REUse	A Figure of Merit (FoM) value, which indicates route stability. Once a route is suppressed, it remains suppressed until its FoM falls below this threshold. **Reuse** must not exceed **suppression**. Default: **750**	
	1..20000	The route becomes available again once its FoM drops to this value.
	DEFault	The route becomes available again once its FoM drops to 750.
HALflife	The time interval, in minutes, within which the route's FoM will halve, if the route remains stable. For example, if **halflife** is 15, the FoM of a stable route reduces by 50% over a 15 minute period, 75% over a 30 minute period, and so on. Default: **15**	
	1..45	The FoM of a stable route halves in this number of minutes.
	DEFault	The FoM of a stable route halves in 15 minutes.

Parameter	Description	
MAXhold	When multiplied by **halflife**, gives the maximum time in minutes for which a suppressed route must remain stable in order to become unsuppressed. The lowest **maxhold** value of 1 gives a maximum suppression time of 1 x **halflife**, and the highest **maxhold** value of 8 gives a maximum suppression time of 8 x **halflife**. For example, if **halflife** is 15 and **maxhold** is 4, the route is unsuppressed after 60 min of stability even if its FoM still exceeds **reuse**. Default: **4**	
	1..8	The **halflife** is multiplied by this value to give the maximum suppression time.
	DEFault	The **halflife** is multiplied by 4 to give the maximum suppression time.
Tip The shortest string you can enter is shown in capital letters.		

Examples

To create BGP route flap damping parameter set 3 with a halflife of 5 minutes and a suppression threshold of 3000, use the command:

```
create bgp damping parameterset=3 halflife=5 suppression=3
```

This set is more tolerant of route instability than the default.

Related Commands

add bgp peer
disable bgp damping
enable bgp damping
show bgp damping
show bgp damping routes

delete bgp aggregate

Syntax `DELete BGP AGGregate=`*`prefix`* `[MASK=`*`ipadd`*`]`

Description This command deletes an aggregate entry from BGP. BGP no longer advertises the aggregate entry. If the aggregate entry is currently in the BGP route table, BGP also sends an update message to all peers to withdraw the route. Associated aggregate suppressed routes are then reconsidered for advertisement.

Parameter	Description
AGGregate	The network prefix of the aggregate entry to delete. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the switch uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the aggregate entry. This parameter is provided for compatibility with other switch commands that specify an IP address and mask; we recommend that you instead specify the mask in the **aggregate** parameter. If you specify a mask in this parameter and the **aggregate** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network.

Tip The shortest string you can enter is shown in capital letters.

Examples To delete the aggregate entry for the network 192.168.8.0/21, use the command:

```
del bgp agg=192.168.8.0/21
```

Related Commands

add bgp aggregate
set bgp aggregate
show bgp aggregate

delete bgp confederationpeer

Syntax `DELete BGP CONFEDerationpeer=1..65534`

Description This command deletes an Autonomous System from the AS confederation to which this switch belongs. An AS confederation is a group of Autonomous Systems that communicate between themselves using confederation BGP, and communicate to Autonomous Systems outside the confederation as if they were a single Autonomous System. For more information about AS confederations, see "AS Confederations" .

The **confederationpeer** parameter specifies the number of an Autonomous System that is no longer to be treated as one of the Autonomous Systems in the confederation. The specified AS number must be an already existing AS confederation peer.

Examples To remove AS 60003 from the AS confederation to which this switch belongs, use the command:

```
del bgp confed=60003
```

Related Commands
add bgp confederationpeer
set bgp
show bgp confederation

delete bgp import

Syntax `DELete BGP IMPort={INTerface|OSPF|RIP|STAtic}`

Description This command deletes an import entry from BGP. Routes from the source specified are no longer imported into the BGP route table. Routes already in the BGP route table are removed.

Parameter	Description	
IMPort	The source of routing information for the routes that are no longer to be imported into BGP. Default: no default	
	INTerface	Stops the import of interface routes.
	OSPF	Stops the import of OSPF routes.
	RIP	Stops the import of RIP or RIP2 routes.
	STAtic	Stops the import of statically configured routes.
Tip The shortest string you can enter is shown in capital letters.		

Examples To stop importing OSPF routes into BGP, use the command:

```
del bgp imp=ospf
```

Related Commands

add bgp import
set bgp import
show bgp import

delete bgp network

Syntax `DELete BGP NETwork=`*`prefix`* `[MASK=`*`ipadd`*`]`

Description This command deletes a network from the list of networks that the switch can advertise to remote BGP peers. If the switch had previously advertised the route, BGP sends an update message to all peers to withdraw the network.

Parameter	Description
NETwork	The network to remove from the list of networks that can be advertised. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the switch uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the network. This parameter is provided for compatibility with other switch commands that specify an IP address and mask; we recommend that you instead specify the mask in the **network** parameter. If you specify a mask in this parameter and the **network** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network
Tip The shortest string you can enter is shown in capital letters.	

Examples To delete the network 192.169.2.0 from the list of networks advertised by BGP, use the command:

```
del bgp net=192.169.2.0/24
```

Related Commands

add bgp network
show bgp network

delete bgp peer

Syntax `DELete BGP PEer=`*`ipadd`*

Description This command deletes a BGP peer from the switch. The BGP peer must be in a disabled state: either never enabled, or previously enabled and subsequently disabled with the **disable bgp peer** command on page 5-97.

The **peer** parameter specifies the IP address of the peer to be deleted, in dotted decimal notation. The peer must be an existing BGP peer on this switch.

Examples To delete a BGP peer whose IP address is 192.168.1.1, use the command:

```
del bgp pe=192.168.1.1
```

Related Commands **disable bgp peer**
show bgp peer

delete bgp peertemplate

Syntax `DELete BGP PEERTemplate=1..30`

Description This command deletes an existing BGP peer policy template from the switch. All peers that have been assigned the specified peer template receive their own copies of the current peer template settings. You can subsequently modify these peers.

The **peertemplate** parameter specifies the ID number of the template to be deleted.

Examples To delete BGP peer template 1, use the command:

```
del bgp peert=1
```

Related Commands **add bgp peertemplate**
show bgp peer

delete ip aspathlist

Syntax `DELete IP ASPATHlist=1..99 [ENTry=1..4294967295]`

Description This command deletes an entry from an AS path list or deletes an entire AS path list. You cannot delete an AS path list if a route map is using it, or if a peer is using it as a filter. First use the **match** parameter of the **delete ip routemap** command on page 5-93 to delete the route map entry, or the **set bgp peer** command on page 5-112 to remove the filter association.

Parameter	Description
ASPATHlist	The ID number of the AS path list to delete, or to remove an entry from. Default: no default
ENTry	The number of the entry to delete. If you do not specify an entry, the whole AS path list is deleted. Default: no default

Tip The shortest string you can enter is shown in capital letters.

Examples To delete the third entry in AS path list 1, use the command:

```
del ip aspath=1 ent=3
```

To delete AS path list 1 and all its entries, use the command:

```
del ip aspath=1
```

Related Commands
add ip aspathlist
show ip aspathlist

delete ip communitylist

Syntax `DELete IP COMmunitylist=1..99 [ENTry=1..4294967295]`

Description This command deletes an entry from a community list or the entire list. You cannot delete a community list if a route map is using it. First use the **match** parameter of the **delete ip routemap** command on page 5-93 to delete the route map entry.

Parameter	Description
COMmunitylist	The ID number of the community list to delete, or to remove an entry from. Default: no default
ENTry	The number of the entry to delete. If you do not specify an entry, the whole community list is deleted. Default: no default

Tip The shortest string you can enter is shown in capital letters.

Examples To delete the entire community list 1, use the command:

```
del ip com=1
```

Related Commands
add ip communitylist
show ip communitylist

delete ip prefixlist

Syntax `DELete IP PREFIXList[=name] [ENTry=1..65535]`

Description This command deletes:

- an entry from a particular prefix list if you specify a name in the **prefixlist** parameter and an **entry** number
- a prefix list if you specify a name in the **prefixlist** parameter but do not specify an **entry** number
- all prefix lists if you do not specify a name in the **prefixlist** parameter or an **entry** number

You cannot delete a prefix list if a route map is using it. Delete the route map entry first.

Examples To delete entry 2 from the prefix list “office”, use the command:

```
del ip prefixl=office entry=2
```

Related Commands
add ip prefixlist
delete ip routemap
set ip routemap
show ip prefixlist

delete ip routemap

Syntax

```
DELete IP ROUTEMap=routemap

DELete IP ROUTEMap=routemap ENTry=1..4294967295

DELete IP ROUTEMap=routemap ENTry=1..4294967295
   MAtch={ASPath|COMmunity|NEXThop|ORIGin|PREFIXList|TAG}

DELete IP ROUTEMap=routemap ENTry=1..4294967295
   SET={ASPath|COMmunity|LOCalpref|MED|ORIGin|TAG}
```

Description

This command deletes one of:

- an entire route map
- a single entry in a route map, or
- a match or set clause in an entry in a route map

For information on route maps, see "Route Maps" on page 5-14 and "How to Configure Route Maps" on page 5-30.

You cannot delete a whole route map if a BGP peer is using it, or if an aggregate, network or import process is using it.

Parameter	Description
ROUTEMap	The name of the route map to be deleted or the name of the route map from which an entry, match clause, or set clause is to be deleted. Default: no default
ENTry	The number of the entry in the route map to be deleted, or the number of the entry from which a match clause or set clause is to be deleted. The entry must already exist in the route map. If you do not specify an entry, the whole route map is deleted. Default: no default
MAtch	The type of match clause to be deleted from the route map entry. Since only one match clause is allowed in a route map entry, this uniquely identifies the clause. Default: no default
SET	The type of set clause to be deleted from the route map entry. Since only one set clause of each type is allowed in a route map entry, this uniquely identifies the clause. Default: no default
Tip The shortest string you can enter is shown in capital letters.	

Examples

To delete the localpref set clause from entry 10 in route map "set_loc_pref", use the command:

```
del ip routem=set_loc_pref ent=10 set=loc
```

Related Commands

add ip routemap
set ip routemap
show ip routemap

destroy bgp damping parameterset

Syntax `DESTroy BGP DAMping PARameterset={ALL|1..100}`

Description This command removes the specified BGP route flap damping parameter set from the group of available parameter sets. You can destroy a parameter set only if BGP damping is disabled for that parameter set or as a whole.

The **parameterset** parameter specifies the parameter set to destroy. If you specify **all**, all user-defined parameter sets are destroyed.

The default parameter set, numbered 0, cannot be destroyed, but you can modify the settings of the default parameter set by using the command **set bgp damping parameterset=0**.

Example To destroy parameter set 3, use the command:

```
destroy bgp damping parameterset=3
```

Related Commands
add bgp peer
disable bgp damping
enable bgp damping
show bgp damping
show bgp damping routes

disable bgp autosoftupdate

Syntax `DISable BGP AUTOsoftupdate`

Description This command disables automatic updating of modified BGP peers. Changes to a peer take effect only when the peer next receives or sends an update message, unless you manually trigger the update by using the **reset bgp peer soft** command on page 5-102. Automatic updating is disabled by default.

Examples To disable automatic updating, use the command:

```
dis bgp auto
```

Related Commands
reset bgp peer
set bgp peer
show bgp peer

disable bgp damping

Syntax `DISable BGP DAMping [PARameterset={ALL|0..100}]`

Description This command disables monitoring and suppression of flapping BGP routes by disabling BGP route flap damping for one or all parameter sets. This command clears route stability history information and may be used to turn off route flap damping temporarily for configuration changes.

The **parameterset** parameter specifies the parameter set to disable. If you do not specify **parameterset**, all route flap damping is disabled. If you specify **parameterset**, only that parameter set is disabled, not all route flap damping, unless you specify the last enabled parameter set. In that case the feature is disabled, which is equivalent to not specifying a parameter set.

Examples To disable BGP route flap damping for all enabled parameter sets and the feature, use the command:

```
disable bgp damping
```

To disable BGP route flap damping for parameter set 3 only, use the command:

```
disable bgp damping parameterset=3
```

If parameter set 3 is the last enabled parameter set, the feature is disabled.

Related Commands
add bgp peer
enable bgp damping
show bgp damping
show bgp damping routes

disable bgp debug

Syntax

```
DISable BGP DEBug[={ALL|DAMping|MSG|STAte|UPdate}[,...]]
   [PEer=ipadd]
```

Description

This command disables one or more forms of BGP debugging, optionally for a given BGP peer.

You can direct BGP debugging to only one manager device at a time. This means that if someone is debugging BGP on another terminal device, you cannot enable debugging on the current terminal device. However, you can use this command to disable debugging for the other device, and then enable debugging for the current device.

Parameter	Description	
DEBug	The debugging options to disable; a single option or a comma-separated list of options. Default: **all**	
	ALL	All debugging options.
	DAMping	Messages to reflect BGP route flap damping state changes and events. Events include routes getting penalised for flapping, route state transition to suppressed and to reuse, routes becoming reachable and routes becoming unreachable.
	MSG	Message reception and transmission. There are four message types: open, update, keepalive, and notify. The output of the debug messages consists of the timestamp, the direction of the message, incoming or outgoing, the IP address of the peer, the message type and the details. For open, keepalive, and notify messages, the details consist of the decoded contents of the packet. For update messages, the details consist of the lengths of the withdrawn routes and NRLI fields, and a complete decode of the path attributes.
	STAte	State machine events and transitions. The output of the debug messages consists of the timestamp, the IP address of the peer, the event that causes the state change, and the old and new states.
PEer	The IP address of the peer for which debugging is no longer required, in dotted decimal notation. The peer must be an existing BGP peer on this switch. Default: no default (debugging is turned off for all BGP peers)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To disable all debugging for all BGP peers, use the command:

```
dis bgp deb
```

Related Commands

enable bgp debug

disable bgp peer

Syntax `DISable BGP PEer={ALL|`*ipadd*`}`

Description This command disables a given BGP peer, or all BGP peers. The switch destroys its TCP connection to the peer, and the associated BGP session. The switch and the peer withdraw any routes they learned during that session.

The **peer** parameter specifies the IP address of the peer to disable, in dotted decimal notation. If you specify **all**, all BGP peers are disabled.

Examples To disable the BGP peer 192.168.1.1, use the command:

```
dis bgp pe=192.168.1.1
```

Related Commands **enable bgp peer**
show bgp peer

enable bgp autosoftupdate

Syntax `ENAble BGP AUTOsoftupdate`

Description This command enables the switch to automatically update BGP peers after you modify them. Automatic updating is disabled by default.

Examples To enable automatic updating, use the command:

```
ena bgp auto
```

Related Commands **reset bgp peer**
set bgp peer
show bgp peer

enable bgp damping

Syntax `ENAble BGP DAMping [PARameterset={ALL|0..100}]`

Description This command enables monitoring and suppression of flapping BGP routes through route flap damping. Route flap damping is disabled by default. Use this command to enable it on:

- all routes by using the command:

  ```
  enable bgp damping
  ```

  This enables damping itself, and all parameter sets. If you associate particular parameter sets with particular peers, the switch applies those parameter sets to update messages from those peers. For all other update messages, the switch uses the default parameter set (see Figure 5-12 on page 5-40).

- some or all routes from a specific BGP peer by using the command:

  ```
  enable bgp damping parameterset=1..100
  ```

  You also need to associate the parameter set with the peer by specifying the parameter set in a route map and applying that route map to the peer. See "How to configure route flap damping" for more information.

- all routes when route flap damping has been previously enabled on only some of the parameter sets by using the command:

  ```
  enable bgp damping parameterset=all
  ```

- all routes that are not associated with a user-defined parameter set by using the command:

  ```
  enable bgp damping parameterset=0
  ```

  The default parameter set is used on these routes.

The **parameterset** parameter specifies the ID number of the parameter set to enable.

Examples To enable BGP route flap damping for all existing parameter sets, use the command:

```
enable bgp damping
```

To enable BGP route flap damping for parameter set 3, use the command:

```
enable bgp damping parameterset=3
```

If some of the parameter sets are currently enabled, enable BGP route flap damping for all other parameter sets by using the command:

```
enable bgp damping parameterset=all
```

Related Commands

add bgp peer
disable bgp damping
show bgp damping
show bgp damping routes

enable bgp debug

Syntax

```
ENAble BGP DEBug={ALL|DAMping|MSG|STAte|UPdate}[,...]
   [PEer=ipadd]
```

Description

This command enables one or more forms of BGP debugging, optionally for a given BGP peer.

You can direct BGP debugging to only one manager device at a time. This means that if someone is debugging BGP on another terminal device, you cannot enable debugging on the current terminal device. However, you can use the **disable bgp debug** command on page 5-96 to disable debugging for the other device, and then enable debugging for the current device.

Parameter	Description	
DEBug	The debugging options to enable; a single option or a comma-separated list of options. Default: **all**	
	ALL	All debugging options.
	DAMping	Messages to reflect BGP route flap damping state changes and events. Events include routes getting penalised for flapping, route state transition to suppressed and to reuse, routes becoming reachable and routes becoming unreachable.
	MSG	Message reception and transmission. There are four message types: open, update, keepalive, and notify. The output of the debug messages consists of the timestamp, the direction of the message, incoming or outgoing, the IP address of the peer, the message type and the details. For open, keepalive, and notify messages, the details consist of the decoded contents of the packet. For update messages, the details consist of the lengths of the withdrawn routes and NRLI fields, and a complete decode of the path attributes.
	STAte	State machine events and transitions. The output of the debug messages consists of the timestamp, the IP address of the peer, the event that causes the state change, and the old and new states.
PEer	The IP address of the peer for which debugging is required, in dotted decimal notation. Default: no default (debugging is turned on for all BGP peers)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To enable message and state debugging for BGP peer 192.168.1.1, use the command:

```
ena bgp deb=msg,state peer=192.168.1.1
```

Related Commands

disable bgp debug

enable bgp peer

Syntax `ENAble BGP PEer={ALL|`*ipadd*`}`

Description This command enables a specific BGP peer or all BGP peers. The switch initialises BGP resources, initiates a TCP connection to the peer, and listens for connections initiated by the peer.

The **peer** parameter specifies the IP address of the peer to enable, in dotted decimal notation. The peer must already exist. If you specify **all**, all BGP peers are enabled.

Examples To enable all BGP peers, use the commands:

```
ena bgp pe=all
```

Related Commands
disable bgp peer
reset bgp peer
show bgp peer

purge bgp damping

Syntax `PURge BGP DAMping`

Description This command purges all configuration information relating to BGP route flap damping. All user-defined parameter sets are destroyed. Accumulated route stability history is cleared, and the route flap damping is disabled.

Caution All current configuration and stability history information will be lost. Use with extreme caution!

Example To purge BGP route flap damping, use the command:

```
purge bgp damping
```

Related Commands
create bgp damping parameterset
disable bgp damping
enable bgp damping
reset bgp damping
show bgp damping

reset bgp damping

Syntax `RESET BGP DAMping [PARameterset={ALL|0..100}]`

Description This command clears the BGP route flap damping stability history for all BGP routes, or the routes attached to the specified **parameterset**.

The **parameterset** parameter specifies a parameter set. The stability history of all routes attached to this parameter set is reset. If you do not specify **parameterset**, or if you specify **parameterset=all**, all bgp damping routes are reset.

Example To clear BGP route flap damping stability history for all routes attached to parameter set 3, use the command:

```
reset bgp damping parameterset=3
```

Related Commands
create bgp damping parameterset
disable bgp damping
enable bgp damping
purge bgp damping
show bgp peer

reset bgp peer

Syntax `RESET BGP PEer={ALL|ipadd}`

Description This command resets a specific BGP peer or all BGP peers. This is the equivalent of disabling the peer, then immediately enabling it. The switch destroys its TCP connection to the peer and the associated BGP session. The switch and the peer withdraw routes they learned from that session. Then the switch initialises BGP resources, initiates a TCP connection to the peer, and listens for connections initiated by the peer.

The **peer** parameter specifies the IP address of the peer to reset, in dotted decimal notation. If you specify **all**, all BGP peers are reset.

Examples To reset BGP peer 192.168.1.1, use the command:

```
reset bgp pe=192.168.1.1
```

Related Commands
enable bgp peer
disable bgp peer
show bgp peer

reset bgp peer soft

Syntax `RESET BGP PEer={ALL|`*ipadd*`} SOft={IN|OUt|ALl}`

Description This command updates all BGP peers or a specific one after you have modified the peer. You do not need to use this command if automatic updating is enabled (see the **enable bgp autosoftupdate** command on page 5-97).

Parameter	Description	
PEer	The peer or peers to update. Default: no default	
	ALL	All peers are updated.
	ipadd	Only the specified peer is updated. The peer is identified by its IP address in dotted decimal notation.
SOft	The direction to update. Default: no default	
	IN	Updates routes that the peer receives. To trigger this update, the peer sends a Route Refresh message to the remote peers it receives routes from. The Route Refresh message triggers the remote peers to resend a BGP Update message.
	OUt	Updates routes that the peer sends. To reset these, the peer simply sends a BGP Update message to the affected BGP peers.
	ALl	Updates routes the peer sends and receives.

Tip The shortest string you can enter is shown in capital letters.

Examples To update BGP peer 192.168.1.1 after you have changed the filter it uses on incoming routes, use the command:

```
reset bgp pe=192.168.1.1 soft=in
```

Related Commands

enable bgp peer
disable bgp peer
show bgp peer

set bgp

Syntax

```
SET BGP [CLUSter=ipadd] [CONfederationid={NONE|1..65534}]
   [LOCalpref={DEFault|0..4294967295}]
   [MED={NONE|0..4294967294}] [PREFExt={DEFault|1..255}]
   [PREFInt={DEFault|1..255}] [ROuterid=ipadd]
   [SELEction_timer=3..60] [TABlemap[=routemap]
```

Description This command sets global BGP parameters in the switch.

Parameter	Description
CLUSter	The cluster ID of the cluster for which the switch is a route reflector (RR). **cluster** is only used if the switch is performing BGP route reflection (see "How to Improve IBGP Scalability" on page 5-42). By default, the cluster ID is the local BGP identifier. This is sufficient when the switch is the only RR in its cluster. However, if the cluster contains multiple RRs, you must give all of the RRs the same cluster ID, which should be the IP address of one of the RRs. Failure to do so may result in routing information loops. Default: the local BGP identifier
CONfederationid	The AS number of the AS confederation to which this switch belongs. This AS number is used as the AS number for this switch when communicating with BGP peers outside the confederation. A peer is outside the confederation if its AS number is different from this switch's AS number, and it is not one of this switch's confederation peers. For more information about AS confederations, see "AS Confederations" on page 5-16. You need to specify this parameter if the switch has peer relationships with any BGP switches outside the confederation. When you create the peer relationship using the **add bgp peer** command on page 5-62, specify the peer's confederation ID in the **remoteas** parameter. Default: **none**
LOCalpref	The local preference value used in all update messages sent to internal peers when this is not overridden by changes due to the actions in a route map. The local preference is used in internal BGP to decide which BGP route to put into the main routing table. The route with the highest value of local preference is used. Default: **100**
DEFault	The local preference is 100.
0..4294967295	The local preference is the specified number.
MED	The Multi-Exit Discriminator (MED) value that is placed in update messages to external BGP peers from this switch when not overridden by the action of a route map. The MED attribute is used by switches in one AS to distinguish between different exit points in the same neighbouring AS. A route with a lower value of MED is used, all other things being equal. For more information about the use of MED in route selection, see "BGP Route Selection" on page 5-9. Default: **none** (no MED attribute is put in update messages)

Parameter	Description	
PREFExt	The route preference BGP gives to routes that it learns from external peers. The switch uses routes with a lower preference value before routes with a higher preference. This parameter has been superseded by the **set ip route preference** command, but is still accepted for backwards compatibility. The **set ip route preference** command allows a wider range of preference values. When you save your configuration using the **create config** command, the router converts the **prefext** parameter to a **set ip route preference** command. Default: **170**	
	DEFault	The route preference is 170.
	1..255	The route preference is the specified number.
PREFInt	The route preference BGP gives to routes that it learns from internal peers. The switch uses routes with a lower preference value before routes with a higher preference. This parameter has been superseded by the **set ip route preference** command, but is still accepted for backwards compatibility. The **set ip route preference** command allows a wider range of preference values. When you save your configuration using the **create config** command, the router converts the **prefint** parameter to a **set ip route preference** command. Default: **170**	
	DEFault	The route preference is 170.
	1..255	The route preference is the specified number.
ROuterid	A 4-byte number that uniquely identifies the switch in a network system in certain circumstances, specified as an IP address in dotted decimal notation. For a description of when the switch uses the router ID, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50. Default: The default local interface's IP address, if it is configured. Otherwise, the highest interface IP address on the switch.	
SELEction_timer	The time in seconds that the router waits, after changes to its BGP routing information database, before it determines whether the changes need to be propagated to its BGP peers. By deferring the operation, multiple changes may be able to be aggregated into a single update. Therefore accepting slightly longer convergence times may reduce the BGP messaging overhead. Default: **15** seconds.	
TABlemap	A route map for BGP routes to be passed through before installing the route into the routing table. If you specify **tablemap** without specifying a routemap, you clear the **tablemap** setting and the switch does not pass routes through a route map before writing them into the routing table. Default: no default (the switch does not pass routes through a route map before writing them into the routing table)	
Tip The shortest string you can enter is shown in capital letters.		

Examples To set the preference of routes learned by internal BGP to be better than the preference of routes learned by OSPF (which is 10), use the command:

```
set bgp prefi=9
```

Related Commands

add bgp confederationpeer
delete bgp confederationpeer
show bgp
show bgp confederation

set bgp aggregate

Syntax

```
SET BGP AGGregate=prefix [MASK=ipadd] [SUMmary={NO|YES}]
   [ROUTEMap[=routemap]]
```

Description

This command modifies the parameters of an existing aggregate entry in the BGP route table.

Parameter	Description	
AGGregate	The aggregate entry to modify, which is identified by its network prefix. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the switch uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default	
MASK	The network mask for the aggregate entry. This parameter is provided for compatibility with other switch commands that specify an IP address and mask; we recommend that you instead specify the mask in the **aggregate** parameter. If you specify a mask in this parameter and the **aggregate** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network	
SUMmary	Whether the switch advertises only the aggregate route, or also the more specific routes that make up the aggregate. Default: **no**	
	no	The switch advertises the more specific routes that make up the aggregate.
	yes	The switch only advertises the aggregate route. Note that unadvertised routes are still displayed in the output of the **show bgp route** command, but are marked with an "s".
ROUTEMap	The route map used to filter the more specific routes that make up the aggregate, or to set attributes for the aggregate route. The *routemap* is the name of the appropriate pre-existing map. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To set the route map for the aggregate entry for the network 192.168.8.0/21 to be route map *agg_map1*, use the command:

```
set bgp agg=192.168.8.0/21 routem=agg_map1
```

Related Commands

add bgp aggregate
delete bgp aggregate
show bgp aggregate

set bgp backoff

Syntax

```
SET BGP BACkoff[=0..100] [BASEtime=0..100]
   [CONSecutive=1..20] [MULtiplier=0..1000] [STep=0..1000]
   [TOTallimit=0..1000]
```

Description

This command configures BGP backoff, which stops BGP processing when system memory is heavily used by another process.

Parameter	Description
BACkoff	The percentage utilisation of system memory that causes BGP to back off, from 0 to 100. Default: 95
BASEtime	The time value in seconds used to calculate the total backoff time for the first backoff iteration, from 0 to 100. The first backoff time is calculated as: `basetime x multiplier/100` Default: 10
CONSecutive	The number of consecutive backoffs that causes BGP to disable all peers, from 1 to 20. Default: 20
MULtiplier	A multiplier for increasing or decreasing the backoff time at each backoff iteration, from 0 to 1000. The change in backoff time at each step is calculated as: `current backoff time x multiplier/100` Default: 100
STep	The number of backoff iterations after which the backoff time is recalculated. Default: 1
TOTallimit	The total number of backoffs that may occur until all peers are disabled. Default: 0 (no limit)

Tip The shortest string you can enter is shown in capital letters.

Examples

To back BGP processing off when the system memory is 90% utilised, use the command:

```
set bgp bac=90
```

Related Commands

set bgp memlimit
show bgp backoff
show bgp memlimit

set bgp damping parameterset

Syntax

```
SET BGP DAMping PARameterset=1..100
   [DESCription=description]
   [SUPpression={DEFault|1..20000}]
   [REUse={DEFault|1..20000}] [HALflife={DEFault|1..45}]
   [MAXhold={DEFault|1..8}]
```

Description This command modifies the settings of a parameter set for route flap damping.

Parameter	Description	
PARameterset	A unique ID number that identifies the parameter set. Default: no default	
DESCription	A description of the parameter set, which has no effect on its operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
SUPpression	A Figure of Merit (FoM) value, which indicates route stability. When a route's FoM exceeds this threshold, the route is suppressed. **Suppression** must be greater than or equal to **reuse**. If **suppression** is less than 1000, a route is suppressed when it becomes unreachable for the first time. Default: **2000**	
	1..20000	The route is suppressed once its FoM exceeds this value.
	DEFault	The route is suppressed once its FoM exceeds 2000.
REUse	A Figure of Merit (FoM) value, which indicates route stability. Once a route is suppressed, it remains suppressed until its FoM falls below this threshold. **Reuse** must not exceed **suppression**. Default: **750**	
	1..20000	The route becomes available again once its FoM drops to this value.
	DEFault	The route becomes available again once its FoM drops to 750.
HALflife	The time interval, in minutes, within which the route's FoM will halve, if the route remains stable. For example, if **halflife** is 15, the FoM of a stable route reduces by 50% over a 15 minute period, 75% over a 30 minute period, and so on. Default: **15**	
	1..45	The FoM of a stable route halves in this number of minutes.
	DEFault	The FoM of a stable route halves in 15 minutes.

Parameter	Description	
MAXhold	When multiplied by **halflife**, gives the maximum time in minutes for which a suppressed route must remain stable in order to become unsuppressed. The lowest **maxhold** value of 1 gives a maximum suppression time of 1 x **halflife**, and the highest **maxhold** value of 8 gives a maximum suppression time of 8 x **halflife**. For example, if **halflife** is 15 and **maxhold** is 4, the route is unsuppressed after 60 min of stability even if its FoM still exceeds **reuse**. Default: **4**	
	1..8	The **halflife** is multiplied by this value to give the maximum suppression time.
	DEFault	The **halflife** is multiplied by 4 to give the maximum suppression time.
Tip The shortest string you can enter is shown in capital letters.		

Examples

To set BGP route flap damping parameter set 3 to have a halflife of 5 minutes and a suppression threshold of 3000, use the command:

```
set bgp damping parameterset=3 halflife=5 suppression=3
```

This set is more tolerant of route instability than the default.

Related Commands

add bgp peer
disable bgp damping
enable bgp damping
show bgp damping
show bgp damping routes

set bgp import

Syntax

```
SET BGP IMPort={INTerface|OSPF|RIP|STAtic}
   [ROUTEMap[=routemap]]
```

Description

This command associates a different route map with a BGP import entry, or disassociates a route map. The import entry instructs BGP to import routes from that route source into the BGP route table, and the route map allows filtering of routes and setting of BGP attributes.

Parameter	Description	
IMPort	The BGP import entry to be modified. This must already have been added to BGP by using the **add bgp import** command on page 5-60. Default: no default	
	INTerface	Imports interface routes.
	OSPF	Imports OSPF routes.
	RIP	Imports RIP or RIP2 routes.
	STAtic	Imports statically configured routes.
ROUTEMap	The route map used to filter the routes imported into BGP and to set attributes for the routes as advertised by BGP. The *routemap* is the name of the appropriate pre-existing map. The route map can **match** on origin, next hop, prefix list or tag, and can use any of the **set** parameters. If you specify the **routemap** parameter without specifying a routemap name, the existing route map is removed from the import entry. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To change the route map for the import entry for importing OSPF routes into BGP to route map *ospf_bgp_map1*, use the command:

```
set bgp imp=ospf routem=ospf_bgp_map1
```

Related Commands

add bgp import
delete bgp import
show bgp import

set bgp memlimit

Syntax `SET BGP MEMlimit[=0..100]`

Description This command limits the percentage of system memory available to BGP.

The **memlimit** parameter specifies the maximum percentage of system memory that BGP can use. When BGP exceeds this percentage, the switch shuts down BGP peers and drops all routes learnt from the peers. The default is 95.

Example To limit BGP to 90% of system memory, use the command:

```
set bgp mem=90
```

Related Commands

set bgp backoff
show bgp memlimit
show bgp memlimit scan

set bgp peer

Syntax

```
SET BGP PEer=ipadd [AUthentication={MD5|NONE}]
   [CLIEnt={NO|YES}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description] [EHOps={DEFault|1..255}]
   [FASTFallover={NO|YES}] [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=[routemap]] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=[routemap]]
   [PASSword=password] [PRIVateasfilter={NO|YES}]
   [REMoteas=1..65534] [SENdcommunity={NO|YES}]

SET BGP PEer=ipadd POLICYTemplate=

SET BGP PEer=ipadd [POLICYTemplate=1..30]
   [AUthentication={MD5|NONE}] [DESCription=description]
   [EHOps={DEFault|1..255}] [FASTFallover={NO|YES}]
   [PASSword=password] [REMoteas=1..65534]
```

Description

This command modifies parameters for an existing BGP peer in the switch. The changes take effect immediately if you have enabled automatic updating by using the **enable bgp autosoftupdate** command on page 5-97. Otherwise, you need to trigger an update by using the **reset bgp peer soft** command on page 5-102.

Parameter	Description	
PEer	The IP address of the peer, in dotted decimal notation. Default: no default	
AUthentication	Whether or not to use MD5 authentication for the BGP peer. If you specify **md5**, you must also specify **password**. Default: **none**	
	MD5	An MD5 digest is added to every BGP packet sent over the TCP connection and is authenticated at the other end. If any part of the digest cannot be verified, the packet is dropped with no response sent.
	NONE	The BGP session is not authenticated.

Parameter	Description	
CLIEnt	Whether the peer is a client of the switch if the switch is a route reflector (RR). RRs selectively advertise routes they learn from their IBGP peers to their other IBGP peers. The switch is a route reflector if it has at least one client peer, so if **client=yes** for at least one of its peers. For more information about route reflection, and client and non-client peers, see "How to Improve IBGP Scalability" on page 5-42. Route reflection is only valid for IBGP peers, so **client=yes** is only valid if the local ASN and the remote ASN are the same. Default: **no**	
	NO	The peer is a non-client peer of the RR. When the RR receives a route from a non-client peer, the RR reflects the route only to its client peers, not to any other non-client peers it has.
	YES	The peer is a client peer of the RR. When the RR receives a route from a client peer, the RR reflects the route to all its peers, both client and non-client.
CONnectretry	The time interval between attempts to establish a BGP connection to the peer, in seconds. Default: **120**	
	0	The switch does not repeat an attempt to establish a BGP connection.
	1..4294967295	The switch waits the specified number of seconds between attempts.
	DEFault	The switch waits 120 seconds between attempts.
DESCription	A description of the peer, which has no effect on its operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
EHOps	The number of hops put in the *TTL* (Time To Live) field of BGP messages for external BGP. Normally, EBGP requires that BGP peers be connected to a common network, which means they are separated by a single hop. Setting **ehops** to a value greater than 1 indicates that multihop EBGP is allowed. Default: **1**	
	1..255	The specified number of hops is put into the TTL field.
	DEFault	The number of hops put in the TTL field is 1.
FASTfallover	Whether fast fallover is enabled on the link to the peer. If fast fallover is enabled, the peer session is reset as soon as the interface that supports the session goes down. If fast fallover is disabled, the session is reset only when its keepalive timer expires. Default: **no** (fast fallover is disabled)	

Parameter	Description	
HOLdtime	The value in seconds that this switch proposes for the time interval between reception of keepalive and/or update messages from this peer. The actual hold time used on a peer connection is negotiated when the connection is opened, as the lower of the hold times proposed. Default: **90**	
	0	This switch proposes not to have a hold time on this BGP connection.
	3..65535	This switch proposes the specified number of seconds as hold time.
	DEFault	This switch proposes a hold time of 90 seconds.
INFilter	The IP routing filter that acts as a prefix filter to filter any prefixes advertised via incoming BGP update messages from this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command and create a filter with a number from 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
INPathfilter	The AS path list that filters the BGP update messages from this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
INRoutemap	The route map that filters and/or modifies prefixes from this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. If you specify the inroutemap parameter without specifying a route map name, the current route map is disassociated from the peer. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
KEEpalive	The time in seconds that this switch would prefer to leave between keepalive messages to this peer. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**	
	1..21845	This switch prefers the specified number of seconds as keepalive interval.
	DEFault	This switch prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the switch uses this address as the source for BGP packets it generates and sends to this BGP peer. For a description of when the switch uses the local interface, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50. Default: **none**	
MAXPREFIX	The maximum number of network prefixes that the switch expects to receive from this peer. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the switch expects to receive from this peer. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The switch logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The switch resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the switch to this peer, of routes that originate in the switch's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.

Parameter	Description	
MINRouteadvert	The minimum time in seconds between advertisements, from the switch to this peer, of routes that originate outside the switch's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this switch advertises to this peer that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the switch sends to this peer specify this switch as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter to filter the prefixes sent in BGP update messages to this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
OUTPathfilter	The AS path list that filters the BGP update messages sent to this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
OUTRoutemap	The route map that filters and/or modifies prefixes sent to this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify the **outroutemap** parameter without specifying a route map name, the current route map is disassociated from the peer. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
PASSword	The key used by the authentication algorithm. Two BGP peers can communicate with each other only if they have the same key. *password* is a character string from 1 to 80 characters long. All printable characters are valid except the question mark and double quotes. If *password* contains spaces, it must be in double quotes. Only valid if **authentication=md5** Default: no default	
POLICYTemplate	The ID number of the peer policy template that applies to this peer. The specified policy template must already exist. To create a template, use the **add bgp peertemplate** command on page 5-68. You can only specify **remoteas**, **description**, **authentication**, **password**, **fastfallover,** and **ehops** at the same time as **policytemplate**. The template provides all other configuration values. Specifying **policytemplate=** with no number disassociates the template and the peer. The peer retains the template's settings. You can then modify the peer.	
PRIVateasfilter	Whether private AS numbers (from 64512 to 65535) are stripped from the AS PATH attribute on update messages the switch sends to the peer. Default: **yes**, if **remoteas** is a public AS number; **no** if **remoteas** is a private AS number.	
REMoteas	The remote Autonomous System to which this peer belongs. If the remote AS number is the same as this switch's AS number, the peer is an internal BGP (IBGP) peer. If the remote AS number is different from this switch's AS number, the peer is an external BGP (EBGP) peer. If the remote AS numbers are different but the switches have the same confederation peer, the peer is a confederation BGP peer. The AS number is assigned by the IANA. Default: no default	
SENdcommunity	Whether the switch includes the community attribute in update messages that it sends to this peer. Default: no	
	YES	The community attribute is set in update messages to this peer. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to this peer. To create a route map use the **add ip routemap** command on page 5-79.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peer.

Tip The shortest string you can enter is shown in capital letters.

Examples To set authentication for a BGP peer whose IP address is 192.168.1.1, use the command:

```
set bgp pe=192.168.1.1 au=md5 password=a-very-secret-password
```

To modify the keepalive time and hold time for a BGP peer whose IP address is 192.168.1.1, use the command:

```
set bgp pe=192.168.1.1 kee=10 hol=30
```

Related Commands

add bgp peer
add ip aspathlist
add ip filter
add ip routemap
delete bgp peer
disable bgp peer
enable bgp peer
reset bgp peer
show bgp peer

set bgp peertemplate

Syntax

```
SET BGP PEERTemplate=1..30 [CLIEnt={NO|YES}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description]
   [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=routemap]
   [PRIVateasfilter={NO|YES}] [SENdcommunity={NO|YES}]
```

Description This command modifies a template for use on BGP peers. BGP applies the changes to all peers that are using the template.

Parameter	Description	
PEERTemplate	The ID number of the template. Default: no default	
CLIEnt	Whether peers that use the template are clients of the switch if the switch is a route reflector (RR). RRs selectively advertise routes they learn from their IBGP peers to their other IBGP peers. The switch is a route reflector if it has at least one client peer, so if **client=yes** for at least one of its peers. See "How to Improve IBGP Scalability" on page 5-42 for more information about route reflection, and client and non-client peers. Route reflection is only valid for IBGP peers, so **client=yes** is only valid if the local ASN and the remote ASN are the same. Default: **no**	
	NO	Peers that use the template are non-client peers of the RR. When the RR receives a route from a non-client peer, the RR reflects the route only to its client peers, not to any other non-client peers it has.
	YES	Peers that use the template are client peers of the RR. When the RR receives a route from a client peer, the RR reflects the route to all its peers, both client and non-client.
CONnectretry	The time interval between attempts to establish a BGP connection to peers that use the template, in seconds. Default: **120**	
	0	The switch does not repeat an attempt to establish a BGP connection.
	1..4294967295	The switch waits the specified number of seconds between attempts.
	DEFault	The switch waits 120 seconds between attempts.

Parameter	Description	
DESCription	A description for the peers that use the template, which has no effect on their operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
HOLdtime	The value in seconds that this switch proposes for the time interval between reception of keepalive and/or update messages from peers that use the template. The actual hold time used on a peer connection is negotiated when the connection is opened, as the lower of the hold times proposed. Default: **90**	
	0	This switch proposes not to have a hold time on this BGP connection.
	3..65535	This switch proposes the specified number of seconds as hold time.
	DEFault	This switch proposes a hold time of 90 seconds.
INFilter	The routing filter that acts as a prefix filter and filters any prefixes advertised via incoming BGP update messages from peers that use the template. You can use a prefix filter to exclude routes to particular networks from update messages. The filter must already exist. To create a filter use the **add ip filter** command. The filter number must be in the range 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
INPathfilter	The AS path list that filters the BGP update messages from peers that use the template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
INRoutemap	The route map that filters and/or modifies prefixes from peers that use the template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
KEEpalive	The time in seconds that this switch would prefer to leave between keepalive messages to peers that use the template. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**	
	1..21845	This switch prefers the specified number of seconds as keepalive interval.
	DEFault	This switch prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the switch uses this address as the source for BGP packets it generates and sends to peers that use this template. For a description of when the switch uses the local interface, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50. Default: **none**	
MAXPREFIX	The maximum number of network prefixes that the switch expects to receive from peers that use the template. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the switch expects to receive from peers that use the template. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The switch logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The switch resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the switch to peers that use the template, of routes that originate in the switch's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.

Parameter	Description	
MINRouteadvert	The minimum time in seconds between advertisements, from the switch to peers that use the template, of routes that originate outside the switch's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this switch advertises to peers that use the template that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the switch sends to peers that use this template specify this switch as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter and filters the prefixes sent in BGP update messages to peers that use this template. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
OUTPathfilter	The AS path list that filters the BGP update messages sent to peers that use this template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 5-73. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
OUTRoutemap	The route map that filters and/or modifies prefixes sent to peers that use this template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 5-79. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the switch applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the switch stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
PRIVateasfilter	Whether private AS numbers (from 64512 to 65535) are stripped from the AS PATH attribute on update messages the switch sends to peers that use this template. Default: **no**	
SENdcommunity	Whether the switch includes the community attribute in update messages that it sends to peers that use this template. Default: no	
	YES	The community attribute is set in update messages to peers that use this template. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to peers that use this template. To create a route map use the **add ip routemap** command on page 5-79.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peers that use this template.

Tip The shortest string you can enter is shown in capital letters.

Examples

To modify a peer policy template 1 to have a hold time of 30 seconds, use the command:

```
set bgp peert=1 hol=30
```

Related Commands

add bgp peer
add bgp peertemplate
set bgp peer
show bgp peer

set ip autonomous

Syntax `SET IP AUtonomous=1..65534`

Description This command sets the switch's autonomous system number (ASN). The switch cannot be configured to use BGP-4 until it is part of an AS and therefore has an ASN.

There are two types of ASNs:

- public ASNs (1 to 64511)

  These are globally unique and assigned by the IANA. They identify the switch's AS when the switch exchanges routes with external organisations.

- private ASNs (64512 to 65534)

  These are non-assigned numbers. You can use them when you are running BGP in an AS confederation, for example. The individual AS numbers in the confederation can be non-assigned numbers.

Caution If the switch has a peer relationship with a public peer, always use an assigned autonomous system number rather than inventing one.

Related Commands
add bgp peer
enable bgp peer
show bgp

set ip prefixlist

Syntax

```
SET IP PREFIXList=name ENTry=1..65535
   [ACTion={MATch|NOMatch}] [MASklength=range]
   [PREfix=ipadd]
```

Description This command modifies an existing entry in a prefix list.

Parameter	Description
PREFIXList	A name that identifies the prefix list. Default: no default
ENTry	An integer that specifies the position of the entry in the prefix list. Default: no default
ACTion	Whether matching prefixes are included or excluded by the process that is using the prefix list. You can use multiple entries in a prefix list with actions of **match** and **nomatch** to build up a list of prefixes. Prefixes with **action=match** are included in the list. Then to use this list of prefixes, create a route map that matches it and apply the route map to a peer. The route map also has an **action** parameter, which determines whether the peer includes or excludes the prefixes in the list. Default: **match** MATch — The prefix list includes the prefix. NOMatch — The prefix list excludes the prefix.
MASklength	The range of prefix mask lengths matched by this entry in the prefix list. The *range* is either a single CIDR mask from 0 to 32, or two masks separated by a hyphen. These options are valid for setting the mask length: • as a mask length range (**masklength=*a*-*b*)**. For a route to match against this entry, its prefix mask length must be between *a* and *b* inclusive. *a* must be less than *b*. • as a single mask length (**masklength=*a***). For a route to match against this entry, its prefix mask length must be exactly *a*. • as an implicit mask length, by not specifying **masklength** (for example, **prefix=192.168.0.0**). For a route to match against this entry, its prefix mask length must correspond exactly to the mask for the class of the given address; in this example, 24. Default: The natural mask for the prefix, based on whether it is a class A, B, or C network
PREfix	The network address matched by this entry in the prefix list, specified in dotted decimal notation. If you do not specify a prefix, the switch sets it to 0.0.0.0. This is correct if you are matching all routes or the default route. Default: 0.0.0.0

Tip The shortest string you can enter is shown in capital letters.

Examples To modify entry 1 in prefix list sample1 so that it matches only routes from the 192.168.0.0/16 network, use the command:

```
set ip prefixlist=sample1 entry=1 action=match
   prefix=192.168.0.0 masklength=16
```

Related Commands

add ip routemap
delete ip prefixlist
set ip routemap
show ip prefixlist

set ip routemap

Syntax to change the action

```
SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}]
```

Syntax to change a *match* clause

```
SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch ASPath=1..99

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch COMmunity=1..99
   [EXAct={NO|YES}]

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch NEXThop=ipadd

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch
   ORIGin={EGP|IGP|INCOmplete}

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch PREFIXList=name

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch TAG=1..65535
```

Syntax to change a *set* clause

```
SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ASPath={1..65534[,...]}

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET
   COMmunity={NOExport|NOAdvertise|NOEXPORTSubconfed|
   AA:XX}[,...] [ADD={NO|YES}]

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET BGPDampid=1..100

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET LOCalpref=0..4294967295

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET MED={0..4294967295|REMOVE}

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ORIGin={IGP|EGP|INCOmplete}
```

Description

This command does one of the following:

- changes the **action** of an entry in a route map
- modifies an entry's **match** clause
- modifies an entry's **set** clause

This command does not create or delete an entry or clause. To create a new entry or clause, use the **add ip routemap** command on page 5-79. To delete an entry or clause, use the **delete ip routemap** command on page 5-93.

Parameters for both *match* and *set* clauses

Parameter	Description
ROUTEMap	The name of the route map that the entry or clause belongs to. Default: no default
ENTry	The ID number of the entry to change. Default: no default
ACtion	Whether matching prefixes or update messages are included or excluded by the process that is using the route map. The **action** parameter applies to the entire entry. It is not meaningful to have **action=exclude** in an entry with a set clause. Default: the current setting. If there is no current setting, **include**

Tip The shortest string you can enter is shown in capital letters.

Parameters for *match* clauses

Parameter	Description
MAtch	Modifies a **match** clause in the entry. The match clause determines which update messages the entry applies to.
ASPath	The ID number of an AS path list. An update message matches the route map entry if its AS path attribute matches the AS path list. To configure an AS path list use the **add ip aspathlist** command on page 5-73. Default: no default
COMmunity	The ID number of a community list. An update message matches the route map entry if its community attribute matches the community list.To configure a community list use the **add ip communitylist** command on page 5-75. Default: no default
EXAct	Whether the community attribute in an update message must precisely match the route map's community list. Valid only when you specify both **match** and **community**. Default: **no**
NEXThop	The IP address of the next node in the path to the route's destination, specified in dotted decimal notation. An update message matches the route map entry if its next_hop attribute matches this address. Default: no default

YES	An update message only matches the route map entry if its community attribute contains all the communities specified in the community list and only those communities.
NO	An update message still matches the route map entry if its community attribute contains all the communities specified in the community list plus extra communities.

Parameter	Description	
ORIGin	An origin attribute value, which indicates BGP's source for the routes at their originating AS. An update message matches the route map entry if its origin attribute matches this value. Default: no default	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.
PREFIXList	The name of a prefix list. A route matches the route map entry if the prefix list contains that route. To create a list use the **add ip prefixlist** command on page 5-77. Default: no default	
TAG	A tag that identifies a particular static route. A route matches this route map entry if it has been tagged with this value by using the **tag** parameter of the **add ip route** command. Use a route map that matches on **tag** when you use the **add bgp network** and **add bgp import** commands to import static routes from IP to BGP. You cannot use **tag** to filter routes that are sent to BGP peers or to match update messages that are received from BGP peers. Default: no default	
Tip The shortest string you can enter is shown in capital letters.		

Parameters for *set* clauses

Parameter	Description	
SET	Modifies a **set** clause in the entry. The set clause changes route attributes.	
ASPath	A comma-separated list of 1 to 10 AS numbers. These numbers are added to the beginning of the update message's AS path attribute. Default: no default	
COMmunity	A comma-separated list of 1 to 10 communities identified by name or number. If the **add** parameter is **yes**, these communities are added to the update message's community attribute. If the **add** parameter is **no** (its default), these communities replace the update message's community attribute. Note that you must also set the peer's **sendcommunity** parameter to **yes** if you want the peer to include the community attribute in the update messages it sends. By default, peers do not include the community attribute in outgoing updates. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	AA:*XX*	The number of a community. *AA* and *XX* are both integers in the range 0 to 65534. *AA* is the AS number. *XX* is a value chosen by the ASN administrator.
ADD	Whether the list of communities specified by the **community** parameter is added to the community attribute, or replaces the community attribute. Only valid when you specify both **set** and **community**. Default: **no**	
	YES	The communities are added to the update message's community attribute.
	NO	The communities replace the update message's community attribute.
BGPDampid	The BGP route flap damping ID that is given to matching routes. This is the same as the ID number of the parameter set that maintains that route's FoM upon it exhibiting instability. If the parameter set does not exist, the default parameter set is applied to matching routes. See "Damping routes on specific peers" for more information about using route maps when configuring route flap damping. Default: no default	

Parameter	Description	
LOCalpref	The metric to write into the update message's local_preference attribute. IBGP uses the local preference to determine which path it should use inside the AS to reach the advertised prefix. A lower metric indicates a preferred path. EBGP does not use this attribute. Default: no default	
MED	The metric to write into the update message's Multi_Exit_Discriminator attribute. EBGP uses the MED to determine the optimal path for reaching the advertised prefixes. A lower metric indicates a preferred path. IBGP does not use this attribute. Default: no default	
	0..4294967295	This value is written into the MED attribute of the matched update message.
	REMOVE	The MED attribute is removed from the matched update message.
ORIGin	The value to write into the update message's origin attribute. The origin indicates BGP's source for the routes at their originating AS.	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.
Tip The shortest string you can enter is shown in capital letters.		

Examples

To change the MED for an existing set MED clause in entry 10 of the route map called set_med, use the command:

```
set ip routem=set_med ent=10 set med=234
```

Related Commands

add ip routemap
delete ip routemap
show ip routemap

show bgp

Syntax `SHow BGP`

Description This command displays information about BGP global configuration and operation (Figure 5-16, Table 5-14).

Figure 5-16: Example output from the **show bgp** command

```
BGP router ID ................. 192.168.1.1
BGP Cluster ID ................ 192.168.1.1
Local autonomous system ....... 123
Confederation ID .............. 1234
Local preference .............. 100 (default)
Multi Exit Discriminator ...... -
Route table route map ......... -
Auto soft reconfiguration ..... Disabled

Number of peers
  Defined ..................... 4
  Established ................. 2
BGP route table
  Iteration ................... 231
  Number of routes ............ 12654
  Route table memory .......... 431872

BGP route flap damping ........ Enabled
```

Table 5-14: Parameters in the output of the **show bgp** command

Parameter	Meaning
BGP router ID	The ID for BGP for this switch. This is the router ID if one has been set using the **set bgp** command. Otherwise it is the local interface if one has been configured. Otherwise it is the highest IP address configured on any of the switch's interfaces. For more information, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50.
BGP Cluster ID	The cluster ID of the cluster for which the switch is a route reflector (RR). See "How to Improve IBGP Scalability" on page 5-42 for more information.
Local autonomous system	The number of the Autonomous System to which this switch belongs.
Confederation ID	The AS number of the AS confederation to which this switch belongs.
Local preference	The value of the local preference for this switch. This is sent to internal BGP peers to help in the decision process for deciding which BGP routes go into the main routing table.
Multi-Exit Discriminator	The value of the multi-exit discriminator for this switch. This is sent to external BGP peers to help in the decision process for deciding which BGP routes go into the main routing table.
Route table route map	The name of the route map that BGP uses before entering a route into the route table.
Auto soft reconfiguration	Whether the switch automatically updates modified peers.

Table 5-14: Parameters in the output of the **show bgp** command (Continued)

Parameter	Meaning
Number of peers	Counters giving the number of configured and established peers.
Defined	The number of peers currently configured on the switch.
Established	The number of peers currently in the established state.
BGP route table	Information about the BGP route table.
Iteration	The number of times the BGP route table has been modified.
Number of routes	The number of routes in the BGP route table.
Route table memory	The amount of switch memory currently used in the BGP route table.
BGP route flap damping	Whether route flap damping is enabled.

Examples To show general BGP parameters and a summary of BGP operations, use the command:

```
sh bgp
```

Related Commands

show bgp aggregate
show bgp import
show bgp network
show bgp peer
show bgp route

show bgp aggregate

Syntax `SHow BGP AGGregate`

Description This command displays information about the BGP aggregate entries configured in this switch (Figure 5-17, Table 5-15).

Figure 5-17: Example output from the **show bgp aggregate** command

```
BGP aggregate entries

Prefix               Summary   Route map
-------------------------------------------
192.168.248.0/21     Yes       aggregate_map
192.168.16.0/21      No        -
-------------------------------------------
```

Table 5-15: Parameters in the output of the **show bgp aggregate** command

Parameter	Meaning
Prefix	Prefix for this aggregate entry. This is the prefix that BGP advertises for this aggregate as long as a route that is a subset of this prefix is present in the BGP routing table.
Summary	Either Yes or No. If yes, the aggregate route is advertised and no subset routes. If no, subset routes are also advertised.
Route map	Name of the route map used to filter routes for this aggregate entry and to set the BGP attributes for the aggregate route entry.

Examples To display information about BGP aggregates, use the command:

```
sh bgp agg
```

Related Commands

add bgp aggregate
delete bgp aggregate
set bgp aggregate
show bgp
show ip routemap

show bgp confederation

Syntax `SHow BGP CONfederation`

Description This command displays information about the BGP confederation setup of this switch (Figure 5-18, Table 5-16).

Figure 5-18: Example output from the **show bgp confederation** command

```
BGP confederation information

Local AS ................ 60001
Confederation ID ........ 1234
Confederation peers ..... 60002
                          60003
Peers ................... 192.168.1.1 (AS 60001, IBGP)
                          192.169.3.2 (AS 60002, CBGP)
                          192.170.4.5 (AS 7658, EBGP)
```

Table 5-16: Parameters in the output of the **show bgp confederation** command

Parameter	Meaning	
Local AS	The AS number of the AS to which this switch belongs.	
Confederation ID	The AS number of the AS confederation to which this switch belongs. The AS confederation behaves as this AS to external BGP peers.	
Confederation peers	The AS numbers of Autonomous Systems in the switch's AS confederation.	
Peers	A list of the configured BGP peers from the point of view of AS confederation configuration. For each peer, the peer address, peer AS, and BGP type is given. BGP type is:	
	EBGP	External BGP. The peer is in a different AS and outside the AS confederation.
	CBGP	Confederation BGP. The peer is in a different AS but inside the AS confederation.
	IBGP	Internal BGP. The peer is in the same AS as this switch.

Examples To display information concerning the AS confederation to which this switch belongs, use the command:

```
sh bgp con
```

Related Commands

add bgp confederationpeer
delete bgp confederationpeer
set bgp
set ip autonomous
show bgp

show bgp backoff

Syntax `SHow BGP BACkoff`

Description This command displays BGP backoff details (Figure 5-19, Table 5-17).

Figure 5-19: Example output of the **show bgp backoff** command

```
BGP Backoff Stats:
 Stat                          Value
---------------------------------
state                          NORMAL
total hist backOffs               0
total backOffs                    0
total backOff Limit               0
was backedOff                  FALSE
consecutive backOffs              0
consecutive backOffs limit        5
base Timeout                     60
Timeout multiplier              100%
Timeout step                      1
Timeout length (sec)             60
Trigger on Mem use               95%
Current Mem use                   7%
---------------------------------
```

Table 5-17: Parameters in the output of the **show bgp backoff** command

Parameter	Meaning
state	Whether the BGP backoff state is NORMAL, BACKING OFF, or DISABLING PEERS.
total hist backOffs	The total number of backoffs that have occurred. Unlike **total backOffs**, this value is not reset when BGP disables peers because it reaches the total or consecutive backoff limit. You can use this value to determine the optimal setting for the total backoff limit.
total backOffs	The number of times that BGP has backed off since it last reached the total backoff limit. Note that this counter is not reset when BGP disables its peers because the consecutive backoff limit is reached.
total backoff limit	The total number of backoffs that cause BGP to disconnect its peers.
was backedOff	Whether BGP was backed off during the last operation.This is used to detect consecutive backoffs.
consecutive backoffs	The number of times in a row that BGP has reached the end of its backoff time and found that system memory is high enough that it backs off again immediately.
consecutive backoffs limit	The number of consecutive backoffs that causes BGP to disable all peers.
base Timeout	The number of seconds used to calculate the total backoff time for the first backoff iteration. The first backoff time is calculated as: `basetime x multiplier/100`

Table 5-17: Parameters in the output of the **show bgp backoff** command (Continued)

Parameter	Meaning
Timeout multiplier	A multiplier for increasing or decreasing the backoff time at each backoff iteration. The change in backoff time at each step is calculated as: `current backoff time x multiplier/100`
Timeout step	The number of backoff iterations after which the backoff time is recalculated.
Timeout length	The current backoff time.
Trigger on Mem use	The percentage of system memory use that triggers BGP to back off.
Current Mem use	The amount of memory used by the system at the moment the command was executed.

Example To see the existing BGP backoff settings, use the command:

```
sh bgp bac
```

Related Commands

set bgp backoff
show bgp memlimit

show bgp counters

Syntax `SHow BGP COUnters[={RIB|UPdate|DB|DB-All|PROCess|NEXThop}[,...]]`

Description This command displays counter information for BGP.

Figure 5-20: Example output of the **show bgp counters=update** command

```
Update Counters:
-------------------------------------------------

  Update Message:
  Header too small ................ 0
  Header too long ................. 0
  Withdrawn too long .............. 0
  Total Path too long ............. 0

  Prefix:
  NLRI error ...................... 0
  Withdrawn errors ................ 0
    Mask > 32bits ................. 0
    Data too long ................. 0
    Invalid Address ............... 0

  Path attributes ................. 0
    Data shorter .................. 0
    Seen twice .................... 0
    Missing mandatory ............. 0

    Origin ........................ 0
      Length wrong ................ 0
      Flags wrong ................. 0
      Unknown origin .............. 0

    AS path ....................... 0
    Silently dropped .............. 0
      Flags wrong ................. 0
      List get failed ............. 0
      Unknown Seg type ............ 0
      Non-confed peer ............. 0
      Data too long ............... 0
      Data too short .............. 0
      AS path loop ................ 0
      Confed seg order ............ 0

    Next hop ...................... 0
      Length wrong ................ 0
      Flags wrong ................. 0
      Address Zero ................ 0
      Interface found ............. 0
```

Figure 5-20: Example output of the **show bgp counters=update** command (Continued)

```
    Med ............................ 0
      Length wrong ................. 0
      Flags wrong .................. 0

    Local preference ............... 0
      Length wrong ................. 0
      Flags wrong .................. 0
      External peer ................ 0

    Atomic aggregate ............... 0
      Length wrong ................. 0
      Flags wrong .................. 0

    Aggregate ...................... 0
      Length wrong ................. 0
      Flags wrong .................. 0

    Community ...................... 0
      Length wrong ................. 0
      Flags wrong .................. 0

    Originator ..................... 0
      Length wrong ................. 0
      Flags wrong .................. 0
      From eBGP Peer ............... 0
      Loops detected ............... 0

    Cluster List ................... 0
      Flags wrong .................. 0
      From eBGP Peer ............... 0
      Loops detected ............... 0

    Unknown Attributes ............. 0
      Flag wrong ................... 0
      Non-transitive ............... 0
      Transitive ................... 0

  Memory:
    Low memory drops ............... 0

  Filter:
    Path exclude ............... 67022
    Prefix exclude ................. 0
    Routemap exclude ............... 0

  Route Selection Fail ............. 0
    Match List empty ............... 0
    Select List empty .............. 0
    NextHop No Route ............... 0

  Internal Control:
    Control Pointers ............... 0
    Message Pointer ................ 0
    Dropped Pointer ................ 0
```

Figure 5-21: Example output of the **show bgp counters=rib** command

```
Total Nodes: 290268
Split Nodes: 135002
Paths:       155266
Withdrawn Paths: 0
Aggregate Paths: 0
nexthop List size: 1
interface route List size: 0
routemap cache List size: 0
BGP Idle Flag: 00000000
Free buffers: 57193

RIB Counters:
--------------------------------------------------

Add to IPG:
  Peer lookup failed ..............  0
  Next hop find failed ............  0
  Next hop no route ...............  0
  IP Log Index NULL ...............  0
  Add failed ......................  0
Delete all from peer:
  Walk pointer NULL ...............  0

Interface list:
  Delete search failed ............  0

Next hop list:
  Delete search failed ............  0

Node Copy:
  Sending route NULL ..............  0
  IP route NULL ...................  0

Withdrawn Route:
  Unlink Error ....................  0
```

Figure 5-22: Example output of the **show bgp counters=db** command

```
AsPathSegDb Entries: 25741
AsSegListDb Entries: 25741
PathAttribDb Entries: 25741
PrefixDb Entries: 0
UnknownAttribDb Entries: 0
UnknownAttribListDb Entries: 0
CommunityListDb Entries: 0
```

Figure 5-23: Example output of the **show bgp counters=db-all** command

```
AsPathSegDb Entries: 25741
AsSegListDb Entries: 25741
PathAttribDb Entries: 25741
PrefixDb Entries: 0
UnknownAttribDb Entries: 0
UnknownAttribListDb Entries: 0
CommunityListDb Entries: 0
```

Figure 5-24: Example output of the **show bgp counters=process** command

```
Last run process: SEND_REACHABLES
Current running process: none
Current waiting processes:

Process back-off metric: 1

Process Stats:
Process                      Start   Continue       Time
-----------------------------------------------------------
DELETE_ALL_FROM_PEER             9     943852          4
IMPORTS_RECHECK                  0          0          0
NEXT_HOPS_RECHECK            16503          0          0
ROUTE_SELECTION              16626   40129126        490
RIB_IN_WITHDRAWN            132933   19489261        287
RIB_IN_REACHABLES          9178505   22910398       3532
UDPATE_MESSAGE             9311509          0       1013
UDPATE_IPG_TABLES          1307362          0        304
SEND_WITHDRAWN              429911          0          5
FLAG_REACHABLES            1304726          0         10
FLAG_UNSYNCED_PEERS         777554          0          3
SEND_REACHABLES          100858683          0       2580
```

Figure 5-25: Example output of the **show bgp counters=nexthop** command

```
nexthop List size: 1
Next hop: 211.30.1.1
  have ip route: 1
    ip route: 0.0.0.0
    ip int index:      1 log int index: 0
    ip metric: 1
  routePt: 06aac0f8
  currentState: 1
  ipRouteRefCount: 167450016
```

show bgp damping

Syntax `SHow BGP DAMping`

Description This command displays information about the BGP route flap damping configuration and operation (Figure 5-26, Table 5-18).

Figure 5-26: Example output from the **show bgp damping** command

```
BGP Route Flap Damping
  Status ..................... ENABLED
  Routes in Engine ........... 40
    Monitored Routes ......... 2
    Suppressed Routes ......... 9
  Forgotten Routes ........... 46

Parameterset 0
  Default configuration
  Current State ..... DISABLED
  Suppression ....... 2000        Reuse .......... 750
  Half life ......... 15 min      Maximum Hold ... 1:4

Parameterset 1
  Severely penalise unreachable to test network
  Current State ..... DISABLED
  Suppression ....... 1200        Reuse .......... 500
  Half life ......... 15 min      Maximum Hold ... 1:4

Parameterset 6
  <Parameterset6>
  Current State ..... ENABLED
  Suppression ....... 1500        Reuse .......... 950
  Half life ......... 10 min      Maximum Hold ... 1:5
```

Table 5-18: Parameters in the output of the **show bgp damping** command

Parameter	Meaning
Status	Whether BGP route flap damping is enabled on the switch.
Monitored Routes	Number of routes that are not suppressed but are being monitored by the suppression engine.
Routes in Engine	Number of routes that the suppression engine is currently maintaining an FoM for.
Suppressed Routes	Number of routes that are currently being suppressed by the suppression engine.
Forgotten Routes	Number of routes that incurred a damping penalty in the past, but have had that penalty forgotten due to those routes experiencing a sufficient period of stability. Note that if the same route is forgotten more than once, it is counted more than once. The counter does not decrement when a previously forgotten route incurs a new damping penalty.
Parameterset n	ID number of the parameter set.

Table 5-18: Parameters in the output of the **show bgp damping** command

Parameter	Meaning
[Description line]	The user-defined description of the parameter set. If the parameter set has no description, the display is "<Parametersetn>" where n is the number of the parameter set.
Current state	Whether the parameter set is enabled.
Suppression	FoM value above which a route advertisement is suppressed.
Reuse	FoM value below which a suppressed route becomes selectable again.
Half Life	Time interval within which the route's FoM will halve if the route remains stable, in minutes.
Maximum Hold	Ratio of half life to the maximum time a route may be suppressed for, regardless of its stability history. For example, if Half Life is 15 and Maximum Hold is 1:4, a route becomes available again after 60 minutes, even if its FoM still exceeds the Reuse value.

Examples To check if parameter set 3 is enabled, use the command:

```
sh bgp dam
```

Related Commands

create bgp damping parameterset
disable bgp damping
enable bgp damping
show bgp damping routes

show bgp damping routes

Syntax `SHow BGP DAMping ROUtes`

Description This command displays information about the routes in the route flap damping suppression engine. It lists all monitored and suppressed routes (Figure 5-27, Table 5-19).

Figure 5-27: Example output from the **show bgp damping routes** command

```
-------------------------------------------------------------------------------
Par  Prefix/Mask         Next Hop     Current     Pen    Num   Last St   Next St
Set                                     State   (FoM)  Flaps    Change    Change
-------------------------------------------------------------------------------
  0  192.168.5.0/24      1.1.1.1          >eM     992      1  00:00:10  01:23:40
  0  192.168.10.0/24     1.1.1.1          >eM     992      1  00:00:10  01:23:40
  0  192.168.7.0/24      1.1.1.1          >eS    2961      3  00:00:20  00:29:45
  0  192.168.3.0/24      1.1.1.1          >eS    4938      5  00:00:20  00:40:50
  0  192.168.9.0/24      1.1.1.1          >eM     992      1  00:00:10  01:23:40
  0  192.168.6.0/24      1.1.1.1          >eM    1976      2  00:00:20  01:38:40
  0  192.168.4.0/24      1.1.1.1          >eS    1984      2  00:00:10  00:21:05
-------------------------------------------------------------------------------
```

Table 5-19: Parameters in the output of the **show bgp damping routes** command

Parameter	Meaning
Par Set	ID of the parameter set used to maintain the FoM for the given route.
Prefix/Mask	Network IP address and CIDR mask of the given route.
Next Hop	IP address of the next hop for the given route.
Current State	Current state of the given route: • Status flags: > the best route for the given prefix * next hop unreachable a aggregate route s aggregate suppressed • Origin flags: i internal e external ? incomplete ! unreachable or withdrawn • Damping flags: S Damping suppressed M Damping monitored
Pen (FoM)	Current FoM value for the route.
Num Flaps	Number of times the route has become unreachable.
Last St Change	Time passed since the route entered its current state.
Next St Change	Time that the route must remain stable in order to change state from suppressed to monitored, or from monitored to ignored.

Examples To find out which routes are currently suppressed, use the command:

```
sh bgp dam rou
```

Related Commands
create bgp damping parameterset
disable bgp damping
enable bgp damping
show bgp damping

show bgp import

Syntax `SHow BGP IMPort`

Description This command displays information about the BGP import entries present in the switch (Figure 5-28, Table 5-20).

Figure 5-28: Example output from the **show bgp import** command

```
BGP import entries

Proto      Route map
--------------------------
OSPF       ospf_proto_map
RIP        rip_proto_map
--------------------------
```

Table 5-20: Parameters in the output of the **show bgp import** command

Parameter	Meaning
Protocol	The routing protocol whose routes are to be imported into BGP; either Interface, OSPF, RIP, or Static.
Route map	The name of the route map used to filter routes and set attributes for routes imported into BGP.

Examples To show the BGP import entries on this switch, use the command:

```
sh bgp imp
```

Related Commands
add bgp import
delete bgp import
set bgp import
show ip routemap

show bgp memlimit

Syntax `SHow BGP MEMlimit`

Description This command displays the percentage of system memory that BGP is limited to, and the current actual memory use by BGP (Figure 5-29).

Figure 5-29: Example output of the **show bgp memlimit** command

```
BGP Memory Limit:  95%,   Actual Use:  0%
```

Example To display the amount of memory BGP is currently using, and its limit, use the command:

```
sh bgp mem
```

Related Commands

set bgp backoff
set bgp memlimit
show bgp memlimit scan

show bgp memlimit scan

Syntax `SHow BGP MEMlimit SCAN`

Description This command displays information about the freelists that are registered to a given module. This output is useful to display a detailed state of BGP memory use, at a given moment in time. (Figure 5-30, Table 5-21 on page 5-148).

Figure 5-30: Example output of the **show bgp memlimit scan** command

```
BGP Memory Limit:  65%,   Actual Use:  0%
Module Freelist Stats: moduleId = 5
module buffer use:      3
module percent use:     0%
         list  unitSize     freeUsed  buffersUsed
-------------------------------------------------
     008418d0        80            0            0
     0084fae0        12            0            0
     0084f104        88            0            0
     00841784        68            0            0
     00841900        48            2            1
     0084fb70        12            0            0
     00855398        32            3            1
     0085b8c4       228            2            1
-------------------------------------------------
Module Freelist Stats: moduleId = 103
module buffer use:      4
module percent use:     0%
         list  unitSize     freeUsed  buffersUsed
-------------------------------------------------
     0081306c        24            0            0
     0081300c        12            0            0
     008131c0        32            0            0
     0081361c        44            0            0
     00813220         8            0            0
     00812fdc       512            0            0
     00813124         8            0            0
     0081353c         8            0            0
     00813190         8            0            0
     00812654        40            0            0
     00813154       520            0            0
     008135c4       112            0            0
     00813350        32            0            0
     00812684       128            0            0
     00811e6c        24            0            0
     008120dc        56            0            0
     008130bc       268            0            0
     008126b4        20            0            0
     00813380        16            0            0
     0081356c        52            0            0
     008131f0        36            0            0
     008123a4        16            0            0
     0081364c        20            1            1
     00811f9c        40            0            0
     00812374      1060            1            1
     008136a4        40            0            0
     0081303c        16            0            0
     0085b8c4       228            2            1
 -------------------------------------------------
```

Table 5-21: Parameters in the output of the **show bgp memlimit scan** command

Parameter	Meaning
BGP Memory Limit	Percentage of system memory that limits BGP.
Actual Use	Percentage of memory BGP currently uses.
Module Freelist Stats	Statistics relating to the freelists for each module. Freelists divide memory buffers into small segments to increase efficiency of memory use.
moduleId	ID number of the software module (see "Module Identifiers and Names" in Appendix B, Reference Tables).
module buffer use	Number of buffers currently in use by the module.
module percent use	Number of buffers currently used by the module, as a percentage of the total number of buffers.
list	Freelists (as a hexadecimal address) registered to the module.
unitSize	Number of bytes each freelist segment uses.
freeUsed	Number of segments of the freelist currently used by the module.
buffersUsed	Number of memory buffers the freelist is currently using.

Example To display the detailed state of current BGP memory use, use the command:

```
sh bgp mem scan
```

Related Commands

set bgp backoff
set bgp memlimit
show bgp memlimit

show bgp network

Syntax `SHow BGP NETwork`

Description This command displays information about the BGP network entries configured in this switch (Figure 5-31, Table 5-22).

Figure 5-31: Example output from the show bgp network command

```
BGP network entries

Prefix                Route map
-----------------------------------
192.168.248.0/21      network_map
192.168.16.0/21       -
-----------------------------------
```

Table 5-22: Parameters in the output of the **show bgp network** command

Parameter	Meaning
Prefix	Prefix for this network entry. This is the prefix that BGP advertises for this network as long as a route that matches this prefix exactly is present in the BGP routing table.
Route map	Name of the route map that is used to filter routes and set the BGP attributes for this network entry.

Examples To display information about BGP networks, use the command:

```
sh bgp net
```

Related Commands

add bgp network
delete bgp network
show bgp
show ip routemap

show bgp peer

Syntax `SHow BGP PEer[=ipadd]`

Description This command displays:

- summary information about all BGP peers if you do not specify a peer address (Figure 5-32, Table 5-23)
- detailed information about a peer, if you specify the IP address of the peer (Figure 5-33 on page 5-151, Table 5-24 on page 5-152). Addresses are specified in dotted decimal notation.

Figure 5-32: Example summary output from the **show bgp peer** command

```
BGP peer entries
Peer          State     AS     InMsg  OutMsg  Template  Role
-------------------------------------------------------------------
192.168.2.254 Estab     12345  23456  3245    -         non-client
192.168.3.16  Idle(D)   123    2      3       2         client
```

Table 5-23: Example summary output from the **show bgp peer** command

Parameter	Meaning
Peer	IP address of the BGP peer.
State	BGP peer state, one of: • Idle • Idle (D)—Idle and also disabled • Connect • Active • OpenSent • OpenConf—OpenConfirm • Estab—Established For more information about the states, see "BGP Operation" on page 5-6.
AS	Number of the autonomous system to which this peer belongs.
InMsg	Number of messages received from this peer since the TCP connection opened.
OutMsg	Number of messages sent to this peer since the TCP connection opened.
Template	ID number of the peer policy template that provides the peer with its settings.
Role	Whether the peer is an EBGP peer of the switch, or for IBGP peers, whether the peer is a client or non-client peer for route reflection. For information about route reflection, and client and non-client peers, see "How to Improve IBGP Scalability" on page 5-42.

Figure 5-33: Example output from the **show bgp peer** command for a specific peer

```
Peer ................ 192.168.10.1
Description ......... -
State ............... Idle
Policy Template ..... 4
  Description ....... Test Template 1
Private AS filter ... Yes
Remote AS ........... 3
BGP Identifier ...... 172.20.25.2
Authentication ...... None
  Password .......... -
Fast Fall-Over ...... Enabled
Role ................ Client
Connect retry ....... 120s
Hold time ........... 90s
Keep alive .......... 30s
Min AS originated ... 15
Min route advert .... 30
Local Interface ..... Not defined

Filtering
  In filter ......... -
  In path filter .... -
  In route map ...... -
  Out filter ........ -
  Out path filter ... -
  Out route map ..... -

Max prefix .......... OFF
External hops ....... 1 (EBGP multihop disabled)
Next hop self ....... No
Send community ...... No
Messages In/Out ..... 0/0
Debugging ........... -
  Device ............ -

Capabilties ......... Route Refresh

Established transitions ........... 0

Session Message counters:
  inOpen ........................... 0       outOpen ......................... 0
  inKeepAlive ...................... 0       outKeepAlive .................... 0
  inUpdate ......................... 0       outUpdate ....................... 0
  inNotification ................... 0       outNotification ................. 0
  inRouteRefresh ................... 0       outRouteRefresh ................. 0

Total Message counters:
  inOpen ........................... 0       outOpen ......................... 0
  inKeepAlive ...................... 0       outKeepAlive .................... 0
  inUpdate ......................... 0       outUpdate ....................... 0
  inNotification ................... 0       outNotification ................. 0
  inRouteRefresh ................... 0       outRouteRefresh ................. 0
```

Table 5-24: Parameters in output of the **show bgp peer** command for a specific peer

Parameter	Meaning
Peer	IP address of the BGP peer.
Description	Description of the peer if it has one.
State	BGP peer state, one of: • Idle • Idle (D)—Idle and also disabled • Connect • Active • OpenSent • OpenConfirm • Established For more information about the states, see "BGP Operation" on page 5-6.
Policy Template	ID number of the peer policy template that provides the peer with its settings.
Description	Description of the peer policy template if it has one.
Private AS filter	Whether private AS numbers (from 64512 to 65535) are stripped from the AS PATH attribute on update messages the switch sends to the peer. "Yes" indicates private AS numbers are stripped. "No" indicates they are not.
Remote AS	Number of the autonomous system to which this peer belongs.
BGP identifier	The ID this switch uses to identify itself to the peer. For more information, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50.
Authentication	Authentication type used for communication with this BGP peer, or None if the connection is not using authentication.
Password	Password for this peer if the connection uses authentication.
Fast-Fallover	Whether fast fallover is enabled on the peer. Fast fallover improves convergence when topology changes, by resetting the BGP session as soon as the switch's interface to the peer goes down.
Role	Whether the peer is an EBGP peer of the switch, or for IBGP peers, whether the peer is a client or non-client peer for route reflection. For information about route reflection, and client and non-client peers, see "How to Improve IBGP Scalability" on page 5-42.
Connect Retry	The time interval for retrying the initial TCP connection to this peer in the event of a connection failure.
Hold time	The configured and actual hold times for this peer. The actual hold time is the lower of the configured hold times of the peer and this switch.
Keep alive	The configured and actual keepalive times for this peer. The actual keepalive time is set by the actual hold time in such a way that the ratio of actual keepalive to hold time is the same as the ratio of configured keepalive to hold time.
Min AS originated	Minimum time between advertisements of routes that originate in this autonomous system.
Min route advert	Minimum time between advertisements of routes that originate outside this autonomous system.

Table 5-24: Parameters in output of the **show bgp peer** command for a specific peer

Parameter	Meaning
Filtering	Settings for inward and outward filtering of routing information via BGP.
In filter	Traffic filter used for filtering incoming routes from this peer.
In path filter	AS path filter used for filtering incoming routes from this peer.
In route map	Route map used for filtering incoming routes from this peer.
Out filter	Traffic filter used for filtering outgoing routes to this peer.
Out path filter	AS path filter used for filtering outgoing routes to this peer.
Out route map	Route map used for filtering outgoing routes to this peer.
Max prefix	Maximum number of route prefixes that may be received from this peer, and the action taken when this number is exceeded. The action is WARNING or TERMINATE.
External hops	Number of hops that can be used to reach this peer when it is an EBGP peer. Having this number exceed 1 allows multihop EBGP.
Next hop self	Whether this switch advertises to this peer that the next hop for all routes is itself.
Send community	Whether this switch sends the community attribute in the path attributes of update messages.
Messages In/Out	Number of incoming/outgoing BGP messages from/to this peer.
Debugging	Debugging types enabled for this peer; one or more of MSG, STATE, UPDATE, or ALL.
Device	Device where debugging output is sent.
Local Interface	The local interface. In certain circumstances, the switch uses this address as the source for BGP packets it generates and sends to the peer. For a description of when the switch uses the local interface, see "How to Set the IP Address By Which the Switch Identifies Itself" on page 5-50.
Capabilities	Extra capabilities negotiated between the peer and the switch. "Route Refresh" indicates that the switch automatically sends route refresh messages to the peer and process route refresh messages from the peer. Route refresh messages request a new update message, and are used after a BGP peer has been modified, to reset its routes.
Established transitions	Number of times the peer session has become established (entered the Established state).
Message Counters	**Session Message Counters** give the number of messages received or sent for the most recently established session with the peer. If the peer session is torn down at any time, the session counters are reset and accumulate from zero when a new session is established. **Total Message Counters** give the number of messages received or sent for all sessions ever established with the peer. For more information about messages, see "BGP Operation" on page 5-6.
InOpen	Number of *open* messages received from this peer. BGP peers use open messages to identify themselves to each other and negotiate settings.

Table 5-24: Parameters in output of the **show bgp peer** command for a specific peer

Parameter	Meaning
OutOpen	Number of *open* messages sent to this peer. BGP peers use open messages to identify themselves to each other and negotiate settings.
InKeepAlive	Number of keepalive messages received from this peer. Keepalive messages maintain the BGP session when the peer has not needed to send update messages.
OutKeepAlive	Number of keepalive messages sent to this peer. Keepalive messages maintain the BGP session when the switch has not needed to send update messages.
InUpdate	Number of update messages received from this peer. BGP peers use update messages to inform each other of route changes.
OutUpdate	Number of update messages sent to this peer. BGP peers use update messages to inform each other of route changes.
InNotification	Number of notification messages received from this peer. BGP peers use notification messages to inform each other of errors.
OutNotification	Number of notification messages sent to this peer. BGP peers use notification messages to inform each other of errors.
InRouteRefresh	Number of route refresh messages received from this peer.
OutRouteRefresh	Number of route refresh messages sent to this peer.

Example To display summary information for all BGP peers, use the command:

```
sh bgp pe
```

To display detailed information for the BGP peer 192.168.1.1, use the command:

```
sh bgp pe=192.168.1.1
```

Related Commands

add bgp peer
delete bgp peer
set bgp peer
show bgp
show ip routemap

show bgp peertemplate

Syntax `SHow BGP PEERTemplate[=1..30]`

Description This command displays information about all BGP peer policy templates, or about the specified BGP peer policy template (Figure 5-34, Table 5-25).

The **peertemplate** parameter specifies the identification number of the BGP peer policy template. If you do not specify a value, information is displayed for all BGP peers.

Figure 5-34: Example output from the **show bgp peertemplate** command

```
BGP Peer Template Information
------------------------------------------------------------
Template.............. 1
Description ......... -
Role ................. Client
Connect retry ....... 120s
Hold time ........... 90s
Keep alive .......... 30s
Min AS originated ... 15
Min route advert .... 30

Filtering
  In filter ......... -
  In path filter .... -
  In route map ...... -
  Out filter ........ -
  Out path filter ... -
  Out route map ..... -

Max prefix .......... OFF
Next hop self ....... No
Send community ...... No

Private AS Numbers .. Don't Filter
------------------------------------------------------------
```

Table 5-25: Parameters in output of the **show bgp peertemplate** command

Parameter	Meaning
Template	ID number of the template.
Description	Description for peers that use the template.
Role	Whether peers that use the template are EBGP peers of the switch, or for IBGP peers, whether the peers are client or non-client peers for route reflection. For information about route reflection, and client and non-client peers, see "How to Improve IBGP Scalability" on page 5-42.
Connect Retry	Interval for retrying the initial TCP connection to peers that use the template in the event of a connection failure.
Hold time	The configured hold time for peers that use the template. The actual hold time is the lower of the configured hold times of the peer and this switch.

Table 5-25: Parameters in output of the **show bgp peertemplate** command (Continued)

Parameter	Meaning
Keep alive	The configured keepalive time for peers that use the template. The actual keepalive time is set by the actual hold time in such a way that the ratio of actual keepalive to hold time is the same as the ratio of configured keepalive to hold time.
Min AS originated	Minimum seconds between advertisements from the switch to peers that use the template for routes that originate in this autonomous system.
Min route advert	Minimum seconds between advertisements from the switch to peers that use the template for routes that originate outside this autonomous system.
Filtering	Settings for inward and outward filtering of routing information via BGP.
In filter	Traffic filter used for filtering incoming routes from peers that use the template.
In path filter	AS path filter used for filtering incoming update messages from peers that use the template.
In route map	Route map used for filtering incoming routes or update messages from peers that use the template, and/or for setting their attributes.
Out filter	Traffic filter used for filtering outgoing routes to peers that use the template.
Out path filter	AS path filter used for filtering outgoing update messages to peers that use the template.
Out route map	Route map used for filtering outgoing routes or update messages to peers that use the template, and/or for setting their attributes.
Max prefix	Maximum number of route prefixes that may be received from peers that use the template, and the action taken when this number is exceeded. The action is WARNING or TERMINATE.
Next hop self	Whether this switch advertises to peers using the template that the next hop for all routes is itself.
Send community	Whether this switch sends the community attribute in the path attributes of update messages to peers that use the template.
Private AS Numbers	Whether private AS numbers (from 64512 to 65535) are stripped from the AS PATH attribute on update messages the switch sends to peers that use the template. "Filter" indicates private AS numbers are stripped. "Don't Filter" indicates they are not.

Example To display the settings of peer template 1, use the command:

```
sh bgp peert=1
```

Related Commands

add bgp peer
add bgp peertemplate
delete bgp peertemplate
set bgp peer
show bgp

show bgp route

Syntax

```
SHow BGP ROUte[=prefix] [REGexp=aspathregexp]
   COMmunity={INTernet|NOAdvertise|NOExport|
   NOEXPORTSubconfed|AA:XX}[,...]}
```

Description

This command displays information about some or all routes in the BGP routing table (Figure 5-35 on page 5-158 and Table 5-26 on page 5-158).

Parameter	Description	
ROUte	The network prefix of the routes to display. The *prefix* is an IP address in dotted decimal notation, optionally followed by the CIDR mask. All routes that match the prefix or that are subnets of the prefix are displayed. If you do not specify a prefix, the switch displays all BGP routes that match the **regexp** and **community** specified. Default: no default	
REGexp	An AS path regular expression. The switch only displays routes with an AS path attribute that matches the regular expression. Regular expressions are a list of one or more AS numbers separated by spaces. To match from the first number in the list, start the expression with the ^ character. To match the last number, end with the $ character. If the expression contains spaces, surround it with double quotes. For more information about valid syntax, see Table 5-8 on page 5-28. For example: • **regexp="23334 45634 88988"** displays any route with a path containing these numbers • **regexp="^23334 45634 88988$"** displays any route with that exact path • **regexp=^23334** displays any route with a path beginning with 23334 Default: no default	
COMmunity	A community name or number. The switch only displays routes with this value of the community attribute. Note that if you specify a community, routes that do not contain a community attribute are not displayed. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	AA:*XX*	The number of a community. *AA* and *XX* are both integers in the range 0 to 65534. *AA* is the AS number. *XX* is a value chosen by the ASN administrator.

Tip The shortest string you can enter is shown in capital letters.

Figure 5-35: Example output from the **show bgp route** command

```
BGP route table
--------------------------------------------------------------------------------
   Prefix              Next hop          Origin      MED           Local pref
   Originator      Path
--------------------------------------------------------------------------------
 > 10.0.0.0/8          0.0.0.0           INCOMPLETE  0             100
   -               EMPTY

 * 12.12.0.0/16        11.0.0.1          INCOMPLETE  0             100
   172.20.193.145  SEQ 1023 1024 1025;

 > 192.168.1.0/24      10.89.0.1         IGP         0             100
   -               EMPTY
--------------------------------------------------------------------------------
```

Table 5-26: Parameters in the output of the **show bgp route** command

Parameter	Meaning
Prefix	Network prefix for this route.
Next hop	IP address of the next hop for this route. A next hop of 0.0.0.0 indicates it is an interface route.
Origin	Origin attribute for this route; one of IGP, EGP, or Incomplete.
MED	Multi Exit Discriminator attribute for this route.
Local pref	Local preference attribute for this route.
Originator	The router ID of the IBGP peer from which the switch learned this route, if any.
Path	The AS path attribute for this route.
>	The best route for the prefix.
*	This route has an unreachable next hop and has been suppressed from selection.
A	An aggregate route.
S	This route has been suppressed as part of aggregation.
D	This route is suppressed by route flap damping.

Examples To display the BGP route table for routes that pass through AS 1234, use the command:

```
sh bgp rou reg=1234
```

Related Commands

show bgp
show bgp aggregate
show bgp import
show bgp network
show bgp peer

show ip aspathlist

Syntax `SHow IP ASPATHlist[=1..99]`

Description This command displays information about a specific AS path list or all lists in the switch (Figure 5-36, Table 5-27).

Figure 5-36: Example output from the **show ip aspathlist** command

```
IP AS path lists

List    Entry    Regular expression
----------------------------------------
1       1        Include ^$
        2        Exclude .*
----------------------------------------
34      1        Exclude ^123
        2        Include 345 234.+123
        3        Exclude .*
----------------------------------------
```

Table 5-27: Parameters in the output of the **show ip aspathlist** command

Parameter	Meaning
List	AS path list number from 1 to 99.
Entry	Entry in the AS path list from 1 to the number of entries in the list.
Regular expression	AS path regular expression for this entry. This is preceded by "exclude" or "include" to indicate what the switch does when there is a a match. For a description of regular expressions, see Table 5-8 on page 5-28.

Examples To display AS path list number 23, use the command:

```
sh ip aspath=23
```

Related Commands

add ip aspathlist
delete ip aspathlist

show ip communitylist

Syntax `SHow IP COMmunitylist[=1..99] [OLDcommunityformat]`

Description This command displays information about a specific community list or all lists in the switch (Figure 5-37, Table 5-28).

The **communitylist** parameter specifies the community list to display. If a list is not specified, all are displayed.

The **oldcommunityformat** parameter specifies that community numbers are displayed in the old format. This is an integer calculated by:

```
AS number x 65536 + community value
```

Figure 5-37: Example output from the **show ip communitylist** command

```
IP community lists

List      Entry     Community list
-----------------------------------------------
 1        1         Include noexport,1234:2345
          2         Exclude 34567:123
 -----------------------------------------------
 23       1         Exclude 12:34
          2         Include internet
 -----------------------------------------------
```

Table 5-28: Parameters in the output of the **show ip communitylist** command

Parameter	Meaning
List	Number of community list from 1 to 99.
Entry	Entry in the community list from 1 to the number of entries in the list.
Community list	The community list for this entry, preceded by "exclude" or "include" to indicate whether a match means that the community attribute should be excluded or included in the function for which the community list is being used.

Examples To display all IP community lists, use the command:

```
sh ip com
```

Related Commands

add ip communitylist
delete ip communitylist

show ip prefixlist

Syntax `SHow IP PREFIXList[=name]`

Description This command displays information about prefix lists on the switch. If you specify a prefix list name, detailed information about that prefix list and its entries is displayed (Figure 5-39, Table 5-30). Otherwise, summary information about all existing prefix lists is displayed (Figure 5-38, Table 5-29).

Figure 5-38: Example summary output from the **show ip prefixlist** command

```
IP Prefix Lists
Name              Entries     In Use
-------------------------------------
Sample                 11     Yes
Test                    3     No
-------------------------------------
```

Table 5-29: Parameters in the output of the **show ip prefixlist** command

Parameter	Meaning
Name	The name of the prefix list.
Entries	The number of entries in the prefix list.
In Use	Whether the prefix list is currently assigned to a route map.

Figure 5-39: Example detailed output from the **show ip prefixlist** command

```
IP Prefix List
-----------------------------------------------------------------------
 Name ................... Sample
 In Use ................. Yes
 Entries:
   Number       Action         Prefix            Length Range
   ---------------------------------------------------------------------
        1       Match          192.168.0.0                16
        3       No Match       0.0.0.0                 25-30
       10       No Match       10.10.10.0              24-30
   ---------------------------------------------------------------------
```

Table 5-30: Parameters in the detailed output of the **show ip prefixlist** command

Parameter	Meaning
Name	Name of the prefix list.
In Use	Whether the prefix list is currently assigned to a route map.
Number	The entry number of the prefix list entry. The switch checks entries in order, starting with the lowest entry number.
Action	Whether the prefix list includes (“match”) or excludes (“nomatch”) any prefix that is within the entry’s prefix range.
Prefix	IP network address for the entry to match on.
Length Range	Range of CIDR mask lengths that the entry can match on.

Examples To see the entries in prefix list “office”, use the command:

```
sh ip prefixl=office
```

Related Commands

add ip prefixlist
add ip routemap
delete ip prefixlist
set ip routemap

show ip routemap

Syntax `SHow IP ROUTEMap[=`*routemap*`] [OLDcommunityformat]`

where *routemap* is a character string 0 to 15 characters long. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character ("_").

Description This command displays information about all IP route maps or a specific one (Figure 5-40, Table 5-31).

The **routemap** parameter specifies the name of the route map to display. If one is not specified, information about all route maps is displayed.

The **oldcommunityformat** parameter specifies that community numbers are displayed in the old format. This is an integer calculated by:

```
AS number x 65536 + community value
```

Figure 5-40: Example output from the **show ip routemap** command

```
IP route maps

Map name
  Entry        Action
      Clauses
-------------------------------------------------------
test
  1            Include
     match     Community   12 Exact=no
       set     LocalPref   3245
       set     Med         8726
       set     Origin      incomplete
  12345        Include
       set     Community   12 noadvertise Add=yes
  4294967295   Include
       set     AS-path     44
       set     Local Pref  3245
       set     Med         8762
       set     Origin      igp
-------------------------------------------------------
```

Table 5-31: Parameters in the output of the **show ip routemap** command

Parameter	Meaning
Map name	Name of the route map.
Entry	Entry number for the route map entry. Entry numbers can be any number, but all entries within a route map are sorted by entry number.
Action	Whether the action for this route map entry is include or exclude.
Clauses	The match and set clauses for this route map entry. Each entry can have only one match clause, and only one set clause of a given type.

Examples To display the IP route map with the name "ospf_bgp_map", use the command:

```
sh ip routem=ospf_bgp_map
```

Related Commands add ip routemap
delete ip routemap
set ip routemap

Chapter 6

Simple Network Management Protocol (SNMP)

Introduction 6-3
Network Management Framework 6-3
Structure of Management Information 6-5
- Names 6-6
- Instances 6-6
- Syntax 6-7
- Access 6-7
- Status 6-8
- Description 6-8

The SNMP Protocol 6-8
- SNMP Versions 6-9
- SNMP Messages 6-9
- Polling versus Event Notification 6-10
- Message Format for SNMPv1 and SNMPv2c 6-10
- SNMP Communities (Version v1 and v2c) 6-11
- SNMPv3 Entities 6-12
- SNMPv3 Message Protocol Format 6-13

SNMPv1 and SNMPv2c on the Switch 6-15
- SNMP MIB Views for SNMPv1 and SNMPv2c 6-15
- SNMP Communities 6-16

Configuration Example (SNMPv1 and v2) 6-19
SNMPv3 on the Switch 6-21
- SNMP MIB Views for SNMPv3 6-21
- SNMP Defined MIB Names 6-22
- SNMP Groups 6-23
- SNMP Users 6-23
- SNMP Target Addresses 6-23
- SNMP Target Params 6-23

Configuration Example (SNMPv3) 6-24
Command Reference 6-25
- add snmp community 6-25
- add snmp group 6-27
- add snmp targetaddr 6-29
- add snmp targetparams 6-30
- add snmp user 6-31
- add snmp view 6-33
- create snmp community 6-34
- delete snmp community 6-36
- delete snmp group 6-37
- delete snmp targetaddr 6-37
- delete snmp targetparams 6-38

delete snmp user 6-38
delete snmp view 6-39
destroy snmp community 6-40
disable snmp 6-40
disable snmp authenticate_trap 6-41
disable snmp community 6-41
enable snmp 6-42
enable snmp authenticate_trap 6-42
enable snmp community 6-43
purge snmp 6-43
set snmp community 6-44
set snmp engineid 6-45
set snmp group 6-46
set snmp local 6-47
set snmp targetaddr 6-48
set snmp targetparams 6-49
set snmp user 6-50
show snmp 6-51
show snmp community 6-54
show snmp group 6-55
show snmp targetaddr 6-56
show snmp targetparams 6-57
show snmp user 6-58
show snmp view 6-59

Introduction

The Simple Network Management Protocol (SNMP) is the network management protocol of choice for the Internet and IP-based internetworks.

This chapter describes the main features of SNMP Version 1 (SNMPv1), SNMP Version 2c (SNMPv2c) and Version 3 (SNMPv3). It also describes support for SNMP on the switch, and how to configure the switch's SNMP agent. See Appendix C, SNMP MIBs for a detailed description of all MIBs (Management Information Bases) and MIB objects supported by the switch.

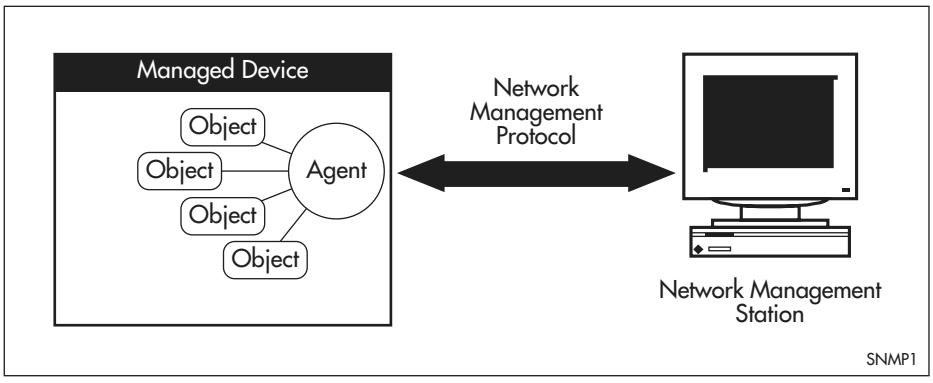
Unless a particular version of SNMP is named, "SNMP" in this chapter refers to versions SNMPv1, SNMPv2c and SNMPv3.

Network Management Framework

A network management system has three components (Figure 6-1 on page 6-3):

- One or more *managed devices*, each containing an agent that provides the management functions. A managed device may be any computing device with a network capability, for example, a host system, workstation, terminal server, printer, router, switch, bridge, hub or repeater.
- One or more *Network Management Stations* (NMS). An NMS is a host system running a network management protocol and network management applications, enabling the user to *manage* the network.
- A *network management protocol* used by the NMS and agents to exchange information.

Figure 6-1: Components of a network management system



The *Internet-standard Network Management Framework* is the framework used for network management in the Internet. The framework was originally defined by three documents:

- RFC 1155, "*Structure and identification of management information for TCP/IP-based internets*" (referred to as the SMI), details the mechanisms used to describe and name the objects to be managed.
- RFC 1213, "*Management Information Base for network management of TCP/IP-based internets: MIB-II*" (referred to as MIB-II), defines the core set of managed objects for the Internet suite of protocols. The set of managed

objects can be extended by adding other MIBs specific to particular protocols, interfaces or network devices.

- RFC 1157, "*A Simple Network Management Protocol (SNMP)*" (referred to as SNMP), is the protocol used for communication between management stations and managed devices.

Subsequent documents that have defined SNMPv2c are:

- RFC 1901 "*Introduction to Community-based SNMPv2*"
- RFC 1902 "*Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)*"
- RFC 1903 "*Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)*"
- RFC 1904 "*Conformance Statements for Version 2 of the Simple Network Management Protocol*"
- RFC 1905 "*Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)*"
- RFC 1906 "*Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)*"
- RFC 1907 "*Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)*"
- RFC 2576 "*Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*"
- RFC 2578 "*Structure of Management Information Version 2 (SMIv2)*"
- RFC 2579 "*Textual Conventions for SMIv2*"
- RFC 2580 "*Conformance Statements for SMIv2*"

Subsequent documents that have defined SNMPv3 are:

- RFC 3410 "*Introduction and Applicability Statements for Internet Standard Management Framework*"
- RFC 3411 "*An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*"
- RFC 3412 "*Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)*"
- RFC 3413 "*Simple Network Management Protocol (SNMP) Applications*"
- RFC 3414 "*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*"
- RFC 3415 "*View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*"
- RFC 3416 "*Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)*"
- RFC 3417 "*Transport Mappings for the Simple Network Management Protocol (SNMP)*"
- RFC 3418 "*Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)*"

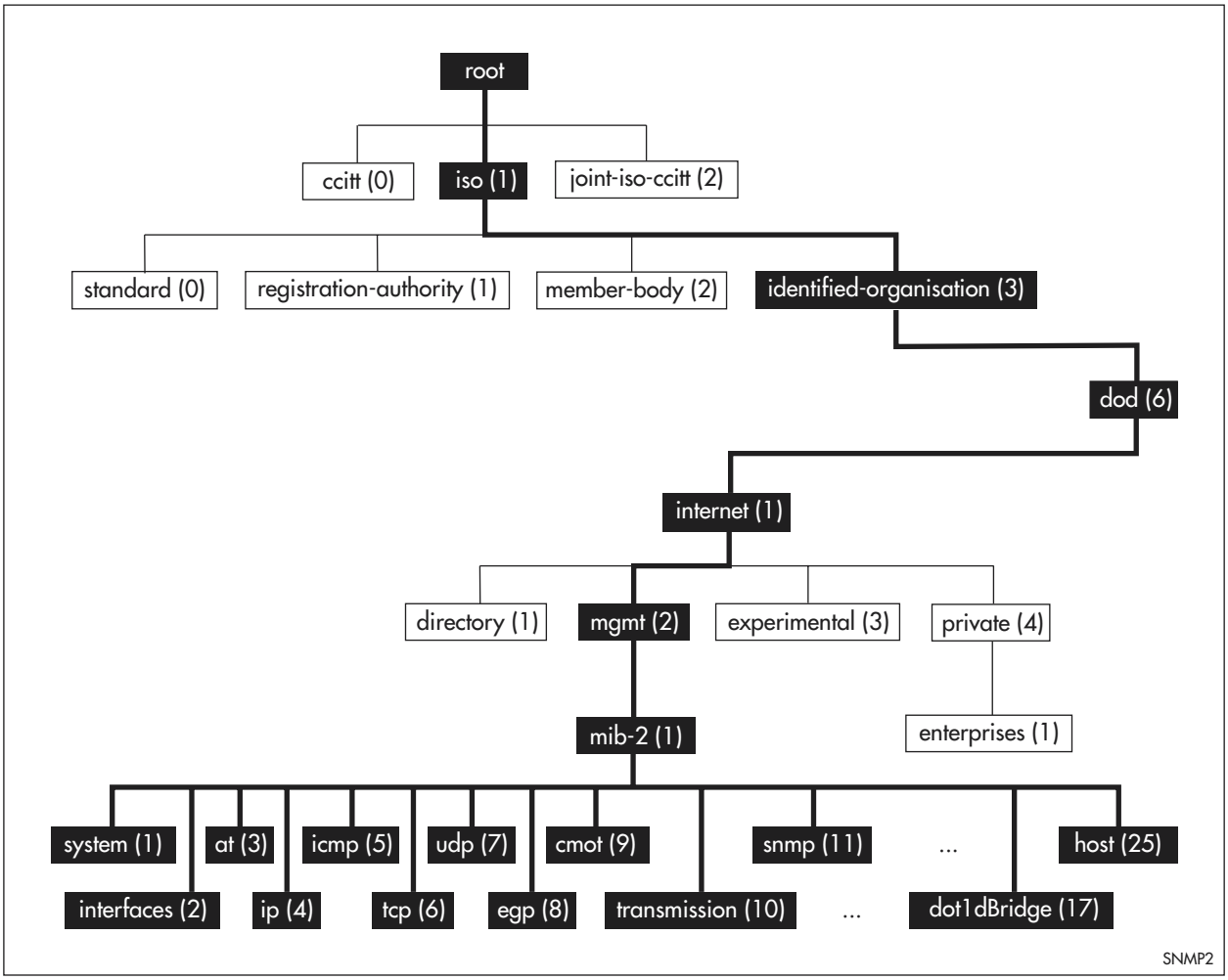
Structure of Management Information

The structure of management information (SMI) defines the schema for a collection of managed objects residing in a virtual store called the *management information base* (MIB). The information in a MIB includes administrative and operational configuration information, as well as counters of system events and activities.

The MIB is organised into a tree-like hierarchy in which nodes are each assigned an identifier consisting of a non-negative integer and an optional brief textual description. The top of the MIB, as it relates to the management of Internet protocols is summarised in Figure 6-2 on page 6-5.

Each managed object is represented by a leaf node and is defined by its name, syntax, access mode, status and description. It can also be specifically identified by its unique position within the tree. This position is expressed as a series of dot-delimited sub-identifiers that start at the root node and end in the sub-identifier at the particular object's leaf node. For example, in Figure 6-2 the object named interfaces would be uniquely identified by the string of individual sub-identifiers, 1.3.6.1.2.1.2.

Figure 6-2: The top levels of the Internet-standard Management Information Base (MIB)



Objects defined in the Internet-standard MIB (MIB-II) reside in the mib(1) sub-tree.

Names

Names are used to identify managed objects, and are hierarchical in nature. An *object identifier* is a globally unique, authoritatively assigned sequence of non-negative integers which traverse the MIB tree from the root to the node containing the object.

Object identifiers may be represented in one of the following forms:

- **Dotted notation** lists the integer values found by traversing the tree from the root to the node in question, separated by dots. For example, the following identifies the MIB-II sub-tree:

  1.3.6.1.2.1

  The following identifies the *sysDescr* object in the system group of MIB-II:

  1.3.6.1.2.1.1.1

- **Textual notation** lists the textual descriptions found by traversing the tree from the root to the node in question, separated by spaces and enclosed in braces. For following example identifies the *internet* sub-tree:

  { iso org dod 1 }

  The name may be abbreviated to a relative form. The following example identifies the first (*directory*) node of the *internet* sub-tree:

  { internet 1 }

- **Combined notation** lists both the integer values and textual descriptions found by traversing the tree from the root to the node in question. The integer value is placed in parentheses after the textual description. The labels are separated by spaces and enclosed in braces. For example, the following identifies the first (*directory*) node in the *internet* sub-tree:

  { iso(1) org(3) dod(6) internet(1) 1 }

  The name may be abbreviated to the following:

  directory(1)

Since there is no effective limit to the magnitude of non-negative integers, and no effective limit to the depth of the tree, the MIB provides an unlimited name space.

An object is also usually assigned an *object descriptor*. The object descriptor is a unique, mnemonic, printable string intended for humans to use when discussing the MIB. Examples are *sysDescr*, *ifTable* and *ipRouteNextHop*.

Instances

Objects are just templates for data types. An actual value that can be manipulated by an NMS is an *instance* of an object. An instance is named by appending an *instance identifier* to the end of the object's object identifier. The instance identifier depends on the object's data type:

- If the object is not a column in a table, the instance identifier is 0 (zero). For example, the instance of the *sysDescr* object is:

  sysDescr.0
  *or*1.3.6.1.2.1.1.1.0

- If the object is a column in a table, the method used to assign an instance identifier varies. Typically, the value of the index column or columns is used.

  The object *ifTable* in MIB-II contains information about interfaces and is indexed by the interface number, *ifIndex*. The instance of the *ifDescr* object for the first interface is:

  ifDescr.1
  or 1.3.6.1.2.1.2.2.1.2.1

  If the index column is an IP address, the entire IP address is used as the instance identifier. The object *ipRouteTable* in MIB-II contains information about IP routes and is indexed by the destination address, *ipRouteDest*. The instance of the *ipRouteNextHop* object for the route 131.203.9.0 is:

  ipRouteNextHop.131.203.9.0
  or 1.3.6.1.2.1.4.21.1.7.131.203.9.0

  If the table has more than one index, the values of all the index columns are combined to form the instance identifier. The object *tcpConnTable* in MIB-II contains information about existing TCP connections and is indexed by the local IP address (*tcpConnLocalAddress*), the local port number (*tcpConnLocalPort*), the remote IP address (*tcpConnRemAddress*) and the remote port number (*tcpConnRemPort*) of the TCP connection. The instance of the *tcpConnState* object for the connection between 131.203.8.36,23 and 131.203.9.197,1066 is:

  tcpConnState.131.203.8.36.23.131.203.9.197.1066
  or 1.3.6.1.2.1.6.13.1.1.131.203.8.36.23.131.203.9.197.1066

Syntax

The syntax of an object describes the abstract data structure corresponding to that object type. For example, INTEGER or OCTET STRING.

Access

The access mode of an object describes the level of access for the object (Table 6-1 on page 6-7).

Table 6-1: Access modes for MIB objects

Access	Description
Read-only	The object's value can be read but not set.
Read-write	The object's value can be read and set.
Write-only	The object's value can be set but not read.
Not-accessible	The object's value cannot be read or set.

Status

The status of an object describes the implementation requirements for the object (Table 6-2 on page 6-8).

Table 6-2: Status values for MIB objects

Status	Description
Mandatory	Managed devices must implement the object.
Optional	Managed devices may implement the object.
Obsolete	Managed devices need no longer implement the object.
Deprecated	Managed devices should implement the object. However, the object may be deleted from the next version of the MIB. A new object with equal or superior functionality is defined.

Description

The definition of an object may include an optional textual description of the meaning and use of the object. This description is often essential for successful understanding of the object.

The SNMP Protocol

The SNMP protocol provides a mechanism for management entities, or stations, to extract information from the *Management Information Base* (MIB) of a managed device.

The normal method of accessing information in a MIB is to use a Network Management Station (NMS), typically a PC or workstation, to send commands to the managed device (in this case the switch) using the SNMP protocol.

SNMP can use a number of different protocols as its underlying transport mechanism, but the most common transport protocol, and the only one supported by the switch, is UDP. Therefore the IP module must be enabled and properly configured in order to use SNMP. SNMP *trap* messages are sent to UDP port 162; all other SNMP messages are sent to UDP port 161. The switch's SNMP agent accepts SNMP messages up to the maximum UDP length the switch can receive.

Note Other transport mappings have been defined (e.g. OSI [RFC 1418], AppleTalk [RFC 1419] and IPX [RFC 1420]), but the standard transport mapping for the Internet (and the one used by the switch) is UDP. The IP module must be enabled and configured correctly. See the Internet Protocol (IP) chapter for detailed descriptions of the commands required to enable and configure IP.

SNMP Versions

The switch supports SNMP version 1 (SNMPv1), SNMP version 2c (SNMPv2c) and SNMP Version 3 (SNMPv3). The three versions operate similarly. SNMPv2c updated the original protocol, and offered the following main enhancements:

- a new format for trap messages.
- the *get-bulk-request* PDU allows for the retrieval of large amounts of data, including tables, with one message.
- more error codes mean that error responses to *set* messages have more detail than is possible with SNMPv1.
- three new exceptions to errors can be returned for *get*, *get-next* and *get-bulk-request* messages. These are: *noSuchObject*, *noSuchInstance*, and *endOfMibView*.

SNMPv3 provides significant enhancements to address the security weaknesses existing in the earlier versions. This is achieved by implementing two new major features:

- Authentication - by using password hashing and time stamping.
- Privacy - by using message encryption.

Support for multiple versions of SNMP is achieved by responding to each SNMP request with a response of the same version. For example, if an SNMPv1 request is sent to the switch, an SNMPv1 response is returned. If an SNMPv2c request is sent, an SNMPv2c response is returned. Therefore, authentication and encryption functions are not invoked when messages are detected as having either an SNMPv1 or SNMPv2c protocol format.

SNMP Messages

The SNMP protocol is termed *simple* because it has only six operations, or messages—*get*, *get-next*, *get-response*, *set*, and *trap*, and SNMPv2c also has the *get-bulk-request* message (Table 6-4 on page 6-10). The replies from the managed device are processed by the NMS and generally used to provide a graphical representation of the state of the network. The two major SNMP operations available to a management station for interacting with a client are the *get* and *set* operations. The SNMP *set* operator can lead to security breaches, since SNMP is not inherently very secure. When forced to operate in either SNMPv1 or v2 mode, when operating with older management stations for example, care must be taken in the choice and safe-guarding of community names, which are effectively passwords for SNMP. See Appendix C, SNMP MIBs for a description of the switch's implementation of each MIB object with read-write access.

Polling versus Event Notification

SNMP employs a *polling* paradigm. A Network Management Station (NMS) polls the managed device for information as and when it is required, by sending *get-request*, *get-next-request*, and/or *get-bulk-request* PDUs to the managed device. The managed device responds by returning the requested information in a *get-response* PDU. The NMS may manipulate objects in the managed device by sending a *set-request* PDU to the managed device.

The only time that a managed device may initiate an exchange of information is the special case of a *trap* PDU. A managed device may generate a limited set of traps to notify the NMS of critical events that may affect the ability of the NMS to communicate with the managed device or other managed devices on the network, and therefore to "manage" the network. Such events include the restarting or re-initialisation of a device, a change in the status of a network link (up or down), or an authentication failure.

Message Format for SNMPv1 and SNMPv2c

Figure 6-3 on page 6-10 shows the format of an SNMP message for v1 and v2c versions. The function of the fields are described in Table 6-3 on page 6-10. There are five different SNMP PDUs (Table 6-4 on page 6-10) and six generic traps (Table 6-5 on page 6-11).

Figure 6-3: Format of an SNMP message

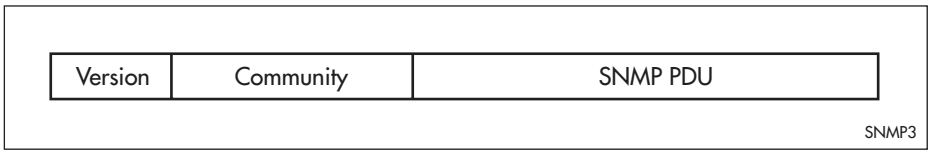


Table 6-3: Fields in an SNMP message

Field	Function
Version	The version of the SNMP protocol. The value is version-1 (0) for the SNMP protocol as defined in RFC 1157, or version-2c (1) for the SNMP protocol as defined in RFC 1902.
Community	The name of an SNMP community, for authentication purposes.
SNMP PDU	An SNMP Protocol Data Unit (PDU).

Table 6-4: SNMP PDUs .

PDU	Function
get-request	Sent by an NMS to an agent, to retrieve the value of an object.
get-next-request	Sent by an NMS to an agent, to retrieve the value of the next object in the sub-tree. A sub-tree is traversed by issuing a get-request PDU followed by successive get-next-request PDUs.
get-bulk-request	Sent by an NMS to an agent to request a large amount of data with a single message. This is for SNMPv2c messages.
set-request	Sent by an NMS to an agent, to manipulate the value of an object.

Table 6-4: SNMP PDUs (Continued).

PDU	Function
get-response	Sent by an agent to an NMS in response to a get-request, get-next-request, get-bulk-response, or set-request PDU.
trap	Sent by an agent to an NMS to notify the NMS of a extraordinary event.
report	Although not explicitly defined in the RFCs, reports are used for specific purposes such as EngineID discovery and time synchronisation.

Table 6-5: Generic SNMP traps

Value	Meaning
coldStart	The agent is re-initialising itself. Objects may be altered.
warmStart	The agent is re-initialising itself. Objects are not altered.
linkDown	An interface has changed state from up to down.
linkUp	An interface has changed state from down to up.
authenticationFailure	An SNMP message has been received with an invalid community name.
egpNeighborLoss	An EGP peer has transitioned to down state.

SNMP Communities (Version v1 and v2c)

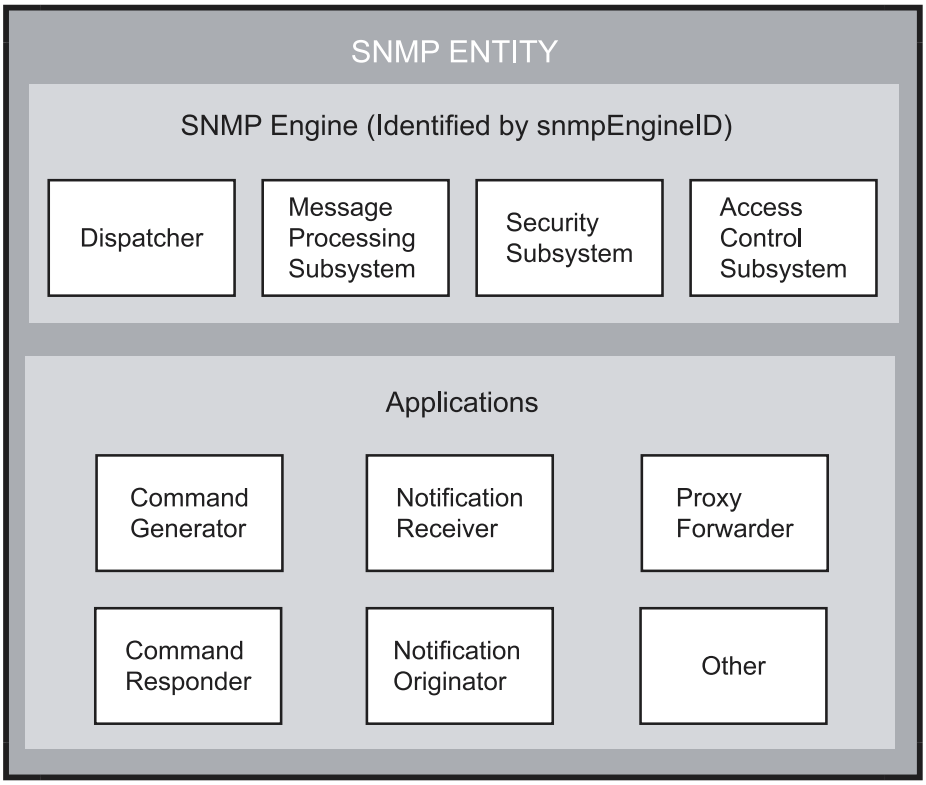
A community is a relationship between an NMS and an agent. The community name is used like a password for a trivial authentication scheme. Both SNMPv1 and SNMPv2c provide security based on the community name only. The concept of communities does not exist for SNMPv3, which instead provides for a far more secure communications method using *entities*, *users* and *groups*.

Note Removing community membership from all SNMPv3 configured devices is strongly recommended. This is to prevent alternative access to these devices via SNMPv1 and SNMv2c that could bypass the additional SNMPv3 security features.

SNMPv3 Entities

Entities comprise one of the basic components of the SNMPv3 enhanced architecture. They define the functionality and internal structure of the SNMP managers and agents. An in depth description of entities can be found in RFC 3411, on which the following text is based. SNMPv3 defines two entity types, a *manager* and an *agent*. Both entity types contain two basic components: an *SNMP engine* and a set of *applications*.This concept is illustrated in Figure 6-4 on page 6-12.

Figure 6-4: SNMPv3 Entity Components



SNMP Engine

The engine provides the basic services to support the agents component applications, in this respect it performs much of the functionality expected of the ISO Session and Presentation layers. These functions include, message transmission and reception, authentication and encryption, and access control to its managed objects database (MIB). The SNMP engine comprises the following components:

- Dispatcher
- Message processing Subsystem
- Security Subsystem
- Access Control Subsystem

Note The only security subsystem presently supported is the user based security model (USM).

Each SNMP engine is identified by an *snmpEngineID* that must be unique within the management system. A one to one association exists between an engine and the entity that contains it.

Entity Applications

The following applications are defined within the agent applications:

- Command Generator
- Notification Receiver
- Proxy Forwarder
- Command Responder
- Notification Originator
- Other

SNMPv3 Message Protocol Format

Figure 6-5 on page 6-13 and Table 6-6 on page 6-14.explain the protocol format of an SNMPv3 message.

Figure 6-5: SNMPv3 Protocol Format

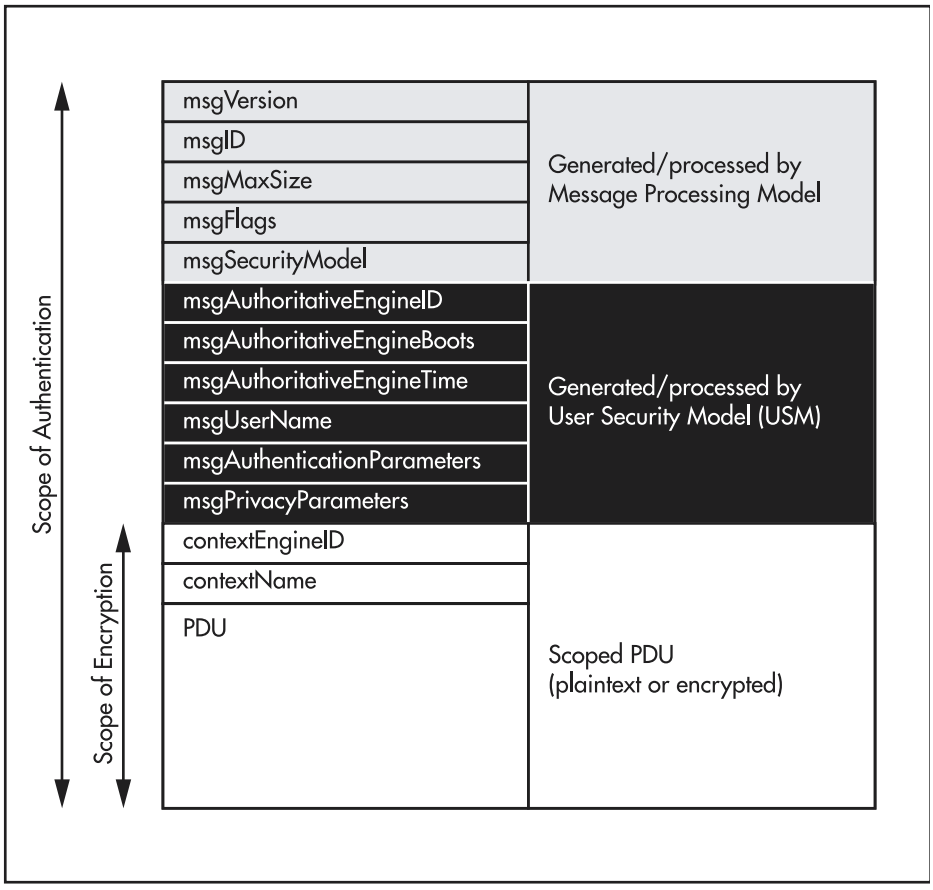


Table 6-6: SNMPv3 PDUs

Value	Meaning
msgVersion	Identifies the message format to be SNMPv3.
msgID	An identifier used between SNMP entities to coordinate message requests and responses. Note that a message response takes the msgID value of the initiating message.
msgMaxSize	Conveys the maximum message size (in octets) supported by the sender of the message. Specified as an integer between 484 and 2^{31}-1.
msgFlags	A single octet whose last three bits indicate the operational mode for privacy, authentication, and report.
msgSecurityModel	An identifier used to indicate the security mode (i.e. SNMPv1, SNMPv2c or SNMPv3 to be used when processing the message. Note that although only the SNMPv3 identifier is accepted by the switch, these earlier version message formats are detected by the *msgVersion* field and processed appropriately.
msgAuthoritativeEngineID	The ID of the authoritative engine that relates to a particular message, i.e. the source engine ID for Traps, Responses and Reports, and the destination engine for Gets, GetNexts, Sets, and Informs.
msgAuthoritativeEngineBoots	A value that represents the number of times the authoritative engine has rebooted since its installation. Its value has the range 1 to 2^{31}-1.
msgAuthoritativeEngineTime	The number of seconds since the authoritative engine snmpEngineBoots counter was last incremented.
msgUserName	The name of the user (principal) on whose behalf the message is being exchanged.
msgAuthenticationParameters	If the message has been authenticated, this field contains a serialized OCTET STRING representing the first 12 octets of the HMAC-MD5-96 output done over the whole message.
msgPrivacyParameters	For encrypted data, this field contains the "salt" used to create the DES encryption Initialisation Vector (IV).
ContextEngineID	Within a particular administrative domain, this field uniquely identifies an SNMP entity that may realize an instance of a context with a particular contextName.
ContextName	A unique name given to a context within a particular SNMP entity.

SNMPv1 and SNMPv2c on the Switch

Although current software levels, 2.6.3 and higher, still support the specific facilities of SNMP v1 and v2, their documentation and use is supplied primarily to provide backward compatibility with older network management systems. The far superior security features offered by implementing SNMPv3 should be used wherever possible. See "SNMPv3 on the Switch" on page 6-21.

The switch's implementation of SNMPv1 is based on RFC 1157 "*A Simple Network Management Protocol (SNMP)*", and RFC 1812, "*Requirements for IP Version 4 Routers*".

The switch's implementation of SNMPv2c is based on the RFCs listed in "Network Management Framework" on page 6-3.

The SNMP agent can be enabled or disabled by using the commands:

```
enable snmp
disable snmp
```

When the SNMP agent is disabled, the agent does not respond to SNMP request messages. The agent is disabled by default. The current state and configuration of the SNMP agent can be displayed by using the command:

```
show snmp
```

SNMP MIB Views for SNMPv1 and SNMPv2c

An SNMP MIB *view* is a arbitrary subset of objects in the MIB. Objects in the view may be from any part of the object name space, and not necessarily the same sub-tree. An *SNMP community profile* is the pairing of an *SNMP access mode* (*read-only* or *read-write*) with the access mode defined by the MIB for each object in the view. For each object in the view, the community profile defines the operations that can be performed on the object (Table 6-7 on page 6-15).

Table 6-7: Community profiles for objects in a MIB view

SNMP Access Mode	Object Access Defined by MIB			
	Read-Only	**Read-Write**	**Write-Only**	**Not Accessible**
Read-Only	get, get-next, trap	get, get-next, trap	*None*	*None*
Read-Write	get, get-next, trap	get, get-next, set, trap	get, get-next, set, trap(*)	*None*

Pairing an SNMP community with an SNMP community profile determines the level of access that the agent affords to an NMS that is a member of the specified community. When an agent receives an SNMP message, it checks the community name encoded in the message. If the agent knows the community name, the message is deemed to be authentic and the sending SNMP entity is accepted as a member of the community. The community profile associated with the community name then determines the sender's view of the MIB and the operations that can be performed on objects in the view.

SNMP Communities

SNMP communities were introduced into SNMPv1 and retained in version 2c. Although the switch's software still supports communities, this is to provide backward compatibility with legacy management systems. For security reasons communities should NOT be used within an SNMPv3 environment.

An SNMP *community* is a pairing of an SNMP agent with a set of SNMP application entities. Communities are the main configuration item in the switch's implementation of SNMPv1 and v2, and are defined in terms of a list of IP addresses which define the SNMP application entities (trap hosts and management stations) in the community. An SNMP community is created by using the command:

```
create snmp community=name [access={read|write}]
    [traphost=ipadd] [manager=ipadd]
    [open={on|off|yes|no|true|false}] [v1traphost=ipadd]
    [v2ctraphost=ipadd]
```

which defines the name of the community (e.g. "public"), and specifies the IP address of a trap host and/or a management station. This command also specifies the version of SNMP received by trap hosts. A community can be modified by using the command:

```
set snmp community=name [access={read|write}]
    [open={on|off|yes|no|true|false}]
```

Community names act as passwords and provide only trivial authentication. Any SNMP application entity that knows a community name can read the value of any instance of any object in the MIB implemented in the switch. Any SNMP application entity that knows the name of a community with write access can change the value of any instance of any object in the MIB implemented in the switch, possibly affecting the operation of the switch. For this reason, care must be taken with the security of community names.

Users are strongly advised not to use SNMP communities where secure network management is required, but instead use the secure network features offered by SNMPv3.

An SNMP community is destroyed by using the command:

```
destroy snmp community=name
```

Additional trap hosts and management stations can be added to or removed from a community by using the commands:

```
add snmp community=name [traphost=ipadd] [manager=ipadd]
    [v1traphost=ipadd] [v2ctraphost=ipadd]
delete snmp community=name [traphost=ipadd] [manager=ipadd]
    [v1traphost=ipadd] [v2ctraphost=ipadd]
```

When a trap is generated by the SNMP agent it is forwarded to all trap hosts in all communities. The community name and manager addresses are used to provide trivial authentication. An incoming SNMP message is deemed authentic if it contains a valid community name and originated from an IP address defined as a management station for that community.

An SNMP community, or the generation of traps by the community, can be temporarily enabled or disabled by using the commands:

```
disable snmp community=name trap
enable snmp community=name trap
```

When a community is disabled, the SNMP agent behaves as if the community does not exist and generates authentication failure traps for messages directed to the disabled community. Information about the configuration of SNMP communities can be displayed by using the command:

```
show snmp community=name
```

Note The SNMP agent does not support a default community called "public" with read-only access, traps disabled and open access as mandated in RFC 1812, as this is a security hole open for users who wish to use the switch with minimal modification to the default configuration. The default configuration of the switch has no defined communities. Communities must be explicitly created. The defaults for other parameters such as the open access flag and the trap enabled flag also follow the principle of security first, access second.

SNMP *authentication* (for SNMPv1 and v2) is a mechanism whereby an SNMP message is declared to be authentic, that is from an SNMP application entity actually in the community to which the message purports to belong. The mechanism may be trivial or secure. The only form of SNMP authentication implemented by the switch's SNMP agent is trivial authentication. The authentication failure trap may be generated as a result of the failure to authentication an SNMP message. The generation of authentication failure traps may be enabled or disabled by using the commands:

```
enable snmp authenticate_trap
disable snmp authenticate_trap
```

Link up/down traps can be enabled or disabled on a per-interface basis by using the commands:

```
enable interface={ifIndex|interface|dynamic} linktrap
disable interface={ifIndex|interface|dynamic} linktrap
```

where *ifIndex* is the value of *ifIndex* for the interface in the Interface Table and *interface* is the name of the interface. If link traps are enabled, when an interface changes to or from the "Down" state an SNMP trap is sent to any defined trap hosts. Link traps are disabled by default on the switch. The current settings for link traps can be displayed by using the command:

```
show interface={ifIndex|interface}
```

The maximum number of link traps generated per minute can be set for each static interface or for all dynamic interfaces by using the command:

```
set interface={ifIndex|interface|dynamic} traplimit=1..60
```

See the Interfaces chapter for a detailed description of the commands for configuring and monitoring link up/down traps.

Switch interfaces can be enabled or disabled via SNMP by setting the *ifAdminStatus* object in the *ifTable* of MIB-II MIB to 'Up(1)' or 'Down(2)' for the corresponding *ifIndex*. If it is not possible to change the status of a particular interface the switch returns an SNMP error message.

The switch's implementation of the *ifOperStatus* object in the *ifTable* of MIB-II MIB supports two additional values—"Unknown(4)" and "Dormant(5)" (e.g. an inactive dial-on-demand interface).

An unauthorised person, with knowledge of the appropriate SNMP community name, could bring an interface up or down. Community names act as passwords for the SNMP protocol. Care should be taken when creating an SNMP community with write access

to select a secure community name and to ensure that this name is known only to authorised personnel.

An SNMP MIB *view* is a subset of objects in the MIB that pertain to a particular network element. For example, the MIB view of a hub would be the objects relevant to management of the hub, and would not include IP routing table objects, for example. The switch's SNMP agent does not allow the construction of MIB views. The switch supports all relevant objects from all MIBs that it implements.

Note The switch's standard **set** and **show** commands can also be used to access objects in the MIBs supported by the switch.

Defining Management Stations Within Communities

You can add management stations to a community either individually, by entering just its IP address, or you can enter a **range** of management stations by entering an IP address that ends with a '/' character followed by a number between 1 and 32. The number that follows the '/' character operates as an address mask to define a range of addresses for the management stations. The following example shows how to allocate a band of three binary addresses to a portion of the subnet 146.15.1.X

Example In this example we will make provision for up to 8 possible management stations within a community called "admin".

1. Decide on the number of management stations that you want to assign to a particular subnet, then decide how many binary digits are required to define this number of addresses. In this case we need up to 8 management stations, so we will assign 3 binary digits (3 binary digits can provide 8 different values). To assign the last 3 binary digits for management stations, we assign a prefix that is a count of all binary digits in the address minus those to be assigned as management stations. In this case the prefix is 29; this being the number of binary digits in an IP address (32) minus the number of digits assigned to the management stations (3).
2. The method used in this step depends on whether or not the community already exists.

a. If the community called "admin" does not exist, create a **new** community called "admin" and allocate a three binary digit block of addresses to the address subnet 146.15.1.X. To do this, enter the IP address of one of the management stations and attach the management station prefix /29,. Use the command:

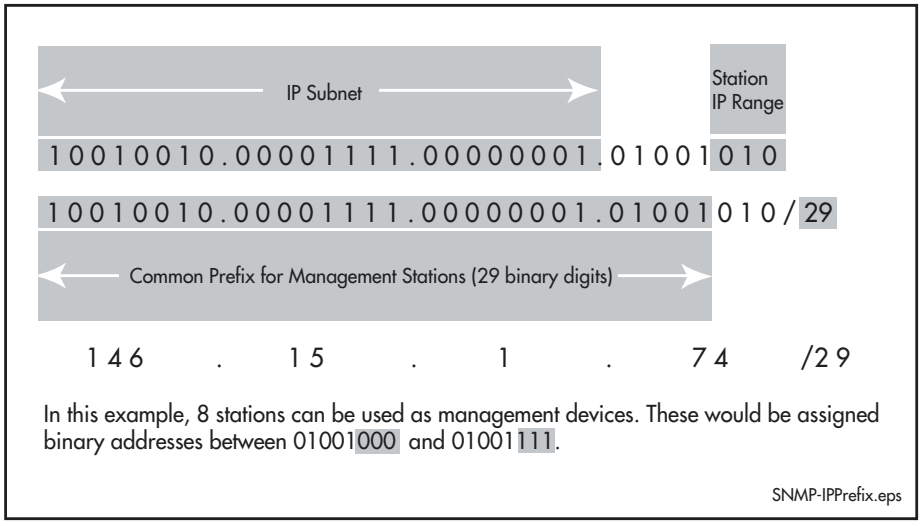
```
create snmp community=admin manager=146.15.1.74/29
```

b. If the community called "admin" already exists, allocate a three binary digit block of addresses to an **existing** community called "admin" with the address subnet 146.15.1.X. To do this enter the IP address of one of the management stations and attach the management station prefix /29, use the command:

```
add snmp community=admin manager=146.15.1.74/29
```

The management IP range and prefix used in this example is illustrated in Figure 6-6.

Figure 6-6: Adding SNMP management stations by assigning an IP prefix



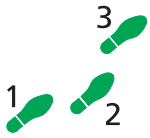
Note For security reasons the common management prefix should be larger than the IP subnet. This is to prevent stations on one subnet being considered valid management stations of a different subnet.

Configuration Example (SNMPv1 and v2)

The following example illustrates the steps required to configure the switch's SNMP agent. In this example, two network management stations have been set up on a large network. The central NMS (IP address 192.168.11.5) is used to both monitor devices on the network and use SNMP *set* messages to manage the devices on the network. Trap messages are sent to this management station. The regional network management station (IP addresses 192.168.16.1) is used just to monitor devices on the network using SNMP *get* messages. Link traps are enabled for all interfaces on this particular switch.

Note Some interface and port types mentioned in this example may not be supported on your switch. The interface and port types that are available vary depending on your product's model, and whether an expansion unit (PIC, NSM) is installed. For more information, see the Hardware Reference.

Note The IP module must be enabled and correctly configured in order to access the SNMP agent in the switch. This is because the IP module handles both the TCP transport functions, and the UDP functions that enable datagrams to transport SNMP messages. See the Internet Protocol chapter for a detailed description of the commands required to enable and configure IP.



To configure SNMP

1. **Enable the SNMP agent.**

   Enable the SNMP agent and enable the generation of authenticate failure traps to monitor unauthorised SNMP access.

   ```
   enable snmp
   enable snmp authenticate_trap
   ```

2. **Create a community with write access for the central NMS.**

   Create a community called “private”, with write access for use only by the central network management station at 192.168.11.5. All traps are sent to this NMS.

   ```
   create snmp community=private access=write
      traphost=192.168.11.5 manager=192.168.11.5 open=no
   ```

   Enable sending of trap messages for this community.

   ```
   enable snmp community=private trap
   ```

Do not use the name “private” in a real network because it is too obvious. Community names act as passwords and provide only trivial authentication. Any SNMP application entity that knows a community name can read the value of any instance of any object in the MIB implemented in the switch. Any SNMP application entity that knows the name of a community with write access can change the value of any instance of any object in the MIB implemented in the switch, possibly affecting the operation of the switch. For this reason, care must be taken with the security of community names.

3. **Create a community with read-only access for the regional NMS.**

   Create a community called “public”, with read-only access for use by the regional network management station at 192.168.16.1. To define a range of management stations, see “Defining Management Stations Within Communities” on page 6-18.

   ```
   create snmp community=public access=read
      manager=192.168.16.1 open=no
   ```

   Enable sending of trap messages for this community.

   ```
   enable snmp community=public trap
   ```

4. **Enable link traps.**

   This switch has static interfaces ppp0, fr3 and x25t0. Additional dynamic interfaces may be created and destroyed as the result of ISDN or ACC calls. Enable link traps for these interfaces, and set a limit of 30 traps per minute for dynamic interfaces.

5. **Enable link traps.**

   If the switch has a ppp0 static interface, additional dynamic interfaces may be created and destroyed as the result of ACC calls. Enable link traps for these interfaces, and set a limit of 30 traps per minute for dynamic interfaces.

   ```
   enable interface=ppp0 linktrap
   enable interface=fr3 linktrap
   enable interface=x25t0 linktrap
   enable interface=dynamic linktrap
   set interface=dynamic traplimit=30
   ```

6. Enable link traps.

Enable link traps for the switch's VLAN interfaces.

```
enable interface=vlan0 linktrap
```

7. Check the configuration.

Check that the current configuration of the SNMP communities matches the desired configuration:

```
show snmp
show snmp community
```

Check that the interface link up/down traps have been correctly configured:

```
show interface=ppp0
show interface=fr3
show interface=x25t0
show interface
```

SNMPv3 on the Switch

SNMPv3 is the third version of the Simple Network Management Protocol. The architecture comprises the following:

- entities - that may be either managers, agents, or a combination of both
- a management information base (MIB),
- a transport protocol.

At least one manager node runs the SNMP management software in every configuration. Managed devices such as routers, servers, and workstations are equipped with an agent software module. The agent provides access to local objects in the MIB that reflect activity and resources at the node. The agent also responds to manager commands to retrieve values from, and set values in the MIB.

SNMP MIB Views for SNMPv3

An SNMP MIB *view* is a arbitrary subset of objects in the MIB. Objects in the view may be from any part of the object name space, and not necessarily the same sub-tree. Views are created by using the command:

```
add snmp view=view-name {oid=oid-tree|mib=mib-name}
   [type={include|exclude}]
```

For a practical MIB view see "Allied Telesyn Enterprise MIB" in Appendix C, SNMP MIBs.

SNMP Defined MIB Names

Table 6-8 on page 6-22 lists the MIB names that are defined within the ATR switch. These names can be used in commands instead of using the MIB tree character string.

Table 6-8: SNMP Defined Mib Names

Value	Meaning
internet	1.3.6.1
mib-2	1.3.6.1.2.1
system	1.3.6.1.2.1.1
interfaces	1.3.6.1.2.1.2
at	1.3.6.1.2.1.3
ip	1.3.6.1.2.1.4
icmp	1.3.6.1.2.1.5
tcp	1.3.6.1.2.1.6
udp	1.3.6.1.2.1.7
egp	1.3.6.1.2.1.8
transmission	1.3.6.1.2.1.10
snmp	1.3.6.1.2.1.11
bgp	1.3.6.1.2.1.15
rmon	1.3.6.1.2.1.16
bridge	1.3.6.1.2.1.17
host	1.3.6.1.2.1.25
mau	1.3.6.1.2.1.26
if	1.3.6.1.2.1.31
private	1.3.6.1.4
alliedTelesyn	1.3.6.1.4.1.207
snmpV2	1.3.6.1.6
snmpModules	1.3.6.1.6.3
snmpFramework	1.3.6.1.6.3.10
snmpMPD	1.3.6.1.6.3.11
snmpTarget	1.3.6.1.6.3.12
snmpUsm	1.3.6.1.6.3.15
snmpVacm	1.3.6.1.6.3.16

SNMP Groups

Groups were introduced as part of SNMPv3. They are the means by which users are assigned their views and access control policy. Groups are created by using the command:

```
add snmp group=group-name
    securitylevel={authnopriv|noauthnopriv|authpriv}
    [readview=view-name] [writeview=view-name]
    [notifyview=view-name]
```

Once a group has been created, users can be added to them. In practice a number of groups would be created, each with varying views and access security requirements. Users would then be added to their most appropriate groups.

Note Each Group name and Security Level pair must be unique within a switch.

SNMP Users

Users were introduced as part of SNMPv3. From a system perspective a user is represented as an entity stored in a table that defines the access and authentication criteria to be applied to access or modify the SNMP MIB data. Users are created by using the command:

```
add snmp user=user-name [group=group-name]
    [authprotocol={none|md5|sha}] [authpassword=password]
    [privprotocol={none|des}] [privpassword=password]
```

SNMP Target Addresses

Target addresses were introduced as part of SNMPv3. They specify the destination and user that receives outgoing notifications such as trap messages. SNMP target address names must be unique within the managed device. Target addresses are created by using the command:

```
add snmp targetaddr=address-name ip=target-ipadd
    [udp=udp-port] params=params-name
```

SNMP Target Params

Target params were introduced as part of SNMPv3. They specify an entry in the snmpTargetParamsTable. SNMP target params names must be unique within the managed device. Target params are created by using the command:

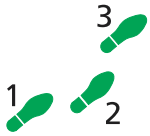
```
add snmp targetparams=params-name
    securitylevel={noauthnopriv|authnopriv|authpriv}
    user=user-name
```

An entry in the target-params table can be used to apply the same security profile to multiple management targets, and is used in conjunction with the **add snmp targetaddr** command on page 6-29 to specify the security profile for a Target Address.

Configuration Example (SNMPv3)

The following example illustrates the steps required to configure the switch's SNMP agent. In this example, two network management stations have been set up on a large network. The central NMS (IP address 192.168.11.5) is used to both monitor devices on the network and use SNMP *set* messages to manage the devices on the network. Trap messages are sent to this management station.

Note The IP module must be enabled and correctly configured in order to access the SNMP agent in the switch, since the IP module handles the UDP datagrams used to transport SNMP messages. See the Internet Protocol (IP) chapter for a detailed description of the commands required to enable and configure IP.



To configure SNMP

1. **Enable the SNMP agent.**

   Enable the SNMP agent and enable the generation of authenticate failure traps to monitor unauthorised SNMP access.

   ```
   enable snmp

   enable snmp authenticate_trap
   ```

2. **Add SNMP views**

   You can specify views using their OID or the predefined MIB name.

   ```
   add snmp view=atmib oid=1.3.6.1.2.14 type=include

   add snmp view=atmib mib=alliedtelesyn type=include
   ```

3. **Add SNMP group**

   ```
   add snmp group=ord-user securitylevel=noauthnopriv
      readview=atmib

   add snmp group=admin-user securitylevel=authnopriv
      readview=atmib
      writeview=atmib
      notifyview=atmib
   ```

4. **Add SNMP users**

   Add users to the groups, using commands like:

   ```
   add snmp user=ken group=admin-user authprotocol=md5
      authpassword=mercury
   ```

5. **Add SNMP targetparams**

   ```
   add snmp targetparams=netmonpc securitylevel=authnopriv
      user=ken
   ```

6. **Add SNMP target address**

   ```
   add snmp targetaddr=target ip=192.168.11.5 udp=162
      params=netmonpc
   ```

Command Reference

This section describes the commands available on the switch to configure and manage the SNMP agent. The IP module must be enabled and correctly configured in order to access the SNMP agent in the switch, since the IP module handles the UDP datagrams used to transport SNMP messages. See the Internet Protocol (IP) chapter for a detailed description of the commands required to enable and configure IP.

The shortest valid command is denoted by capital letters in the Syntax section.

add snmp community

Syntax

```
ADD SNmp COMmunity=name [TRAPhost=ipadd] [MAnager=ipadd]
    [V1traphost=ipadd] [V2ctraphost=ipadd]
```

where:

- *name* is a string 1 to 15 characters long. Valid characters are any printable ASCII character and is case sensitive, that is “Public” is a different name from “public”.
- *ipadd* is an IP address in dotted decimal notation.
- *prefix* is an IP address range in dotted decimal notation followed by a ‘/’ character and a decimal number from 0 to 32.

Description

This command adds a trap host or a management station to a specific SNMP community. Note that for security reasons, communities should not be used when operating SNMPv3.

The **community** parameter specifies the SNMP community. The community must already exist on the switch.

The **traphost**, **v1traphost** and **v2ctraphost** parameters specify a trap host for the SNMP community. This is the IP address of a device to which traps generated by the switch are sent. A community may have more than one trap host, but only one can be specified when the community is created. If the parameter is not specified, the community has no defined trap host. If the **traphost** or **v1traphost** parameter is used to specify a trap host, the switch sends SNMPv1 format traps to this host. If the **v2ctraphost** parameter is used to specify a trap host, the switch sends SNMPv2c format traps to this host. The same trap host can be used for sending version 1 and version 2c traps, but you have to add it to the SNMP community twice.

The **manager** parameter can specify either a single management station (by specifying a single IP address) or a range of management stations (by using the ‘/’ character followed a decimal number from 0 to 32 to specify a range of IP addresses) for this SNMP community. If no parameter is specified, then the community will have no management station defined to it. Note that additional management stations or ranges can be created once the community has been created.

Examples To add the host 192.168.1.1 as both a trap host and a management station to the existing SNMP community “Administration”, use the command:

```
add snmp community=administration traphost=192.168.1.1
   manager=192.168.1.1
```

To add the host 192.168.1.2 as both a version 1 and a version 2c trap host to the existing SNMP community “Administration”, use the command:

```
add snmp community=administration traphost=192.168.1.2
   v2ctraphost=192.168.1.2
```

To add a block of 16 management stations to a community named administration.

```
add snmp community=administration manager=192.168.7.31/28
```

Note that this will add management station addresses of 192. 168.7.16 to 192.168.7.31.

See Also
create snmp community
Defining Management Stations Within Communities
delete snmp community
destroy snmp community
disable snmp community
enable snmp community
show snmp community

add snmp group

Syntax

```
ADD SNmp GROup=group-name
   SECuritylevel={AUTHnopriv|NOAUthnopriv|AUTHPRiv}
   [READview=view-name] [WRITEview=view-name]
   [NOTifyview=view-name]
```

where:

- *group-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *group-name* contains spaces, it must be in double quotes.
- *view-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *view-name* contains spaces, it must be in double quotes.

Description

This command is used with SNMP version 3 only, and adds an SNMP group, optionally setting the security level and view access modes for the group. You must specify at least one of the optional parameters **readview**, **writeview** or **notifyview**.

The **group** parameter names a group to which security and authorisation levels can be applied. The security and access levels defined for the group represent the minimum required of its users in order to gain access.

The practical number of groups that may be added is limited by the memory available within the device.

The **securitylevel** parameter specifies the minimum access and privacy levels for the group. **noauthnopriv** specifies that users belonging to this group need not apply authentication or privacy (encryption). **authnopriv** specifies that users belonging to this group must apply authentication based on either MD5 or SHA algorithms, but they need not apply privacy (encryption). **authpriv** specifies that users belonging to this group must apply authentication based on either MD5 or SHA algorithms, together with encryption based on the DES algorithm. This option provides both security and privacy.

The **readview** parameter specifies the MIB contents that this group can read. Note that this content is specified for each view-name by using the **add snmp view** command. Group members are not able to read any MIB objects unless this parameter is specified.

The **writeview** parameter specifies the MIB contents that this group can modify. Note that this content is specified for each view-name by using the **add snmp view** command. Group members cannot modify MIB objects unless this parameter is specified.

The **notifyview** parameter specifies the notify contents that this group can receive. Note that this content is specified for each view-name by using the **add snmp view** command. Group members cannot read MIB objects unless this parameter is specified. Group members cannot receive notifications unless this parameter is specified.

Examples To add SNMP group, for ordinary users, use the command:

```
add snmp group=usergroup securitylevel=noauthpriv
    readview=useraccess writeview=useraccess
```

To add SNMP group, for network administrators, use the command:

```
add snmp group=admingroup securitylevel=authpriv
    readview=adminaccess writeview=adminaccess
```

See Also **delete snmp group**
set snmp group
show snmp group

add snmp targetaddr

Syntax

```
ADD SNmp TARgetaddr=address-name IP=target-ipadd
    [UDP=udp-port] PARams=params-name
```

where:

- *address-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *address-name* contains spaces, it must be in double quotes.
- *target ipadd* is an IP address in dotted decimal notation.
- *udp-port* is a decimal number from 1 to 255.
- *params-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *params-name* contains spaces, it must be in double quotes.

Description

This command adds an SNMP target address entry to the *snmpTargetAddrTable*, and is used with SNMP version 3 only.

The **targetaddr** parameter specifies the target parameters for outgoing notifications such as trap messages. SNMP target address names must be unique within the managed device. The practical number of addresses that may be added is limited by the memory available within the device.

The **ip** parameter specifies the IP address to which notifications are sent.

The **udp** parameter specifies the identification number of a UDP port. In accordance with RFC recommendations, the default value is 162.

The **params** parameter specifies the reference to the entry in *snmpTargetParamsTable*. To create a target **params**, use the **add snmp targetparams** command on page 6-30.

Examples

To add an **snmp targetaddr** called "target" and a **params** called “params” on address "127.0.0.1", use the command:

```
add snmp targetaddr=target ip=127.0.0.1 params=params
```

See Also

delete snmp targetaddr
set snmp targetaddr
show snmp targetaddr

add snmp targetparams

Syntax

```
ADD SNmp TARGETParams=params-name
   SECuritylevel={NOAUthnopriv|AUTHnopriv|AUTHPRiv}
   USer=user-name
```

where:

- *params-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *params-name* contains spaces, it must be in double quotes.
- *user-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *user-name* contains spaces, it must be in double quotes.

Description

This command adds an SNMP target params entry to the *snmpTargetParamsTable*, and is used with SNMP version 3 only.

The **targetparams** parameter specifies an SNMP target params entry in the *snmpTargetParamsTable*. An SNMP target params entry with the specified name must not exist in the router.

The **securitylevel** parameter specifies the security level for this target parameters. **noauthnopriv** specifies that no authentication or privacy is used. **authnopriv** specifies that authentication is based on the MD5 or SHA algorithm. This option provides security but no privacy. **authpriv** specifies that authentication be based on the MD5 or SHA algorithm, and to base encryption on the DES algorithm.

The **user** parameter specifies the SNMP user. Although a user with the specified name need not pre-exist in the switch, it must be added (using the **add snmp user** command) before user access can be enabled.

If the user's authentication and privacy protocols do not support the specified target parameters security level, then the SNMP message is not sent.

Examples

To add an SNMP target params "params" with security level “noAuthNoPriv” for user "test", use the command:

```
add snmp targetparams=params seclevel=noauthnopriv user=test
```

See Also

delete snmp targetparams
set snmp targetparams
show snmp targetparams

add snmp user

Syntax

```
ADD SNmp USer=user-name [GROup=group-name]
    [AUTHprotocol={NONE|MD5|SHa}] [AUTHPassword=password]
    [PRIVprotocol={NONE|DES}] [PRIVPassword=password]
```

where:

- *user-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *user-name* contains spaces, it must be in double quotes.
- *group-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *group-name* contains spaces, it must be in double quotes.
- *password* is a string 8 to 32 characters long. It may contain any printable character and is case sensitive. If *password* contains spaces, it must be in double quotes.

When passwords need to be entered on both the local and remote stations, these should be entered in such a way as not to compromise the system security. Note that telnet should not be used for this purpose, as this transmits the passwords across the network in cleartext. Refer to RFC 3414 for more information on key management. For more information on key management contact your authorised Allied Telesyn distributor or reseller.

Description

This command is used with SNMP version 3 only, and adds an SNMP user as a member to a specific SNMP group. Additionally it provides the option of selecting an authentication protocol and (where appropriate) an associated password. Similar options are offered for selecting a privacy protocol and password. Note that each SNMP user must be configured on “both” the manager and agent entities. Where passwords are used, these must be the same for both entities.

The **user** parameter specifies the SNMP user. The user name is used to reference the SNMP user in all other SNMP commands. A user with the specified name must not exist in the switch.

The **group** parameter specifies the SNMP group to which the user becomes a member. Although a group with the specified name need not pre-exist in the switch, it must be added (using the **add snmp group** command) before user access can be enabled.

The **authprotocol** parameter specifies the authentication protocol for the SNMP user (either **md5**, **sha**, or **none**). If this parameter is not specified, then the default is none. Note that the authentication specified for the SNMP user must match that specified for its declared group. For example, a user configured with the authentication default of **none**, cannot access a group whose authentication is defined as **md5**.

The **authpassword** parameter specifies the authentication password for the SNMP user. If **authprotocol** is set to **md5** or **sha**, the **authpassword** must be specified.

The **privprotocol** parameter specifies privacy protocol for the SNMP user (either **des** or **none**). If this parameter is not specified, then the default is none. The **privprotocol** cannot be set to **des** if **authprotocol** is set to **none**.

The **privpassword** parameter specifies the privacy password for the SNMP user. If **privprotocol** is set to DES, the **privpassword** must be specified.

Examples To add SNMP user "authuser" as a member of group "usergroup", with authentication protocol "MD5", authentication password "Authpass", privacy protocol "DES" and privacy password "Privpass", use the command:

```
add snmp user=authuser group=usergroup authprotocol=md5
   authpassword=authpass privprotocol=des
   privpassword=privpass
```

See Also **delete snmp user**
set snmp user
show snmp user

add snmp view

Syntax

```
ADD SNmp VIEW=view-name {OID=oid-tree|MIB=mib-name}
    [TYpe={INCLude|EXCLude}]
```

where:

- *vew-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *vew-name* contains spaces, it must be in double quotes.
- *oid-tree* is string in a decimal and dot format that is from 1 to 32 sub-identifiers long. For a an explanation of the OID tree and a definition of sub-identifiers, see "Structure of Management Information" on page 6-5.
- *mib-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *mib-name* contains spaces, it must be in double quotes. Note that the mib name must be one that is defined to the switch, see "SNMP Defined MIB Names" on page 6-22.

Description

This command adds an SNMP view and specifies the initial sub-tree. Further sub-trees can then be added by specifying a new OID to an existing view.

The **view** parameter specifies the name of the SNMP view. View names are used to reference the SNMP views in all other SNMP commands.

The **oid** parameter specifies the object identifier of the ASN.1 sub-tree to be included or excluded from the view. The sub-tree has to be specified as a character string consisting of numbers, for example, 1.3.6.1.2.1. If this parameter is specified, the **mib** parameter cannot be specified.

The **mib** parameter specifies a predefined MIB node to be included or excluded from the view. If this parameter is specified, the **oid** parameter cannot be specified. See "SNMP Defined MIB Names" on page 6-22.

The **type** parameter specifies whether a particular OID sub-tree entry is included or excluded from the SNMP view.

Examples

To add SNMP view "mib2view" that includes all objects in the MIB-II sub-tree, use the command:

```
add snmp view=mib2view oid=1.3.6.1.2.1 type=include
```

See Also

delete snmp view
show snmp view

create snmp community

Syntax

```
CREate SNmp COMmunity=name [ACCess={read|write}]
   [TRAPhost=ipadd] [MAnager={ipadd|prefix}]
   [OPen={ON|OFF|YES|NO|True|False}] [V1traphost=ipadd]
   [V2ctraphost=ipadd]
```

where:

- *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If the string contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.
- *prefix* is an IP address range in dotted decimal notation followed by a '/' character and a decimal number from 0 to 32.

Description

This command creates an SNMP community, optionally setting the access mode for the community and defining a trap host and manager.

The **community** parameter specifies the name of the community. The community name is used to reference the SNMP community in all other SNMP commands. A community with the specified name must not already exist in the switch.

The **access** parameter specifies the access mode for this community. If **read** is specified, management stations in this community can only read MIB variables from the switch, that is perform SNMP *get* or *get-next* operations. If **write** is specified, management stations in this community can read and write MIB variables, that is perform SNMP *set*, *get* and *get-next* operations. The default is **read**.

The **traphost**, **v1traphost** and **v2ctraphost** parameters specify a trap host for the SNMP community. This is the IP address of a device to which traps generated by the switch are to be sent. A community may have more than one trap host, but only one can be specified when the community is created. If the parameter is not specified, the community has no defined trap host. If the **traphost** or **v1traphost** parameter is used to specify a trap host, the switch sends SNMPv1 format traps to this host. If the **v2ctraphost** parameter is used to specify a trap host, the switch sends SNMPv2c format traps to this host. The same trap host can be used for sending version 1 and version 2c traps, but you have to add it to the SNMP community twice.

The **manager** parameter can specify either a single management station (by specifying a single IP address) or a range of management stations (by using the '/' character followed a decimal number from 0 to 32 to specify a range of IP addresses) for this SNMP community. If no parameter is specified, then the community will have no management station defined to it. Note that additional management stations or ranges can be created once the community has been created.

The **open** parameter allows access to this community by any management station, and overrides the management stations defined with the **manager** parameter. The default is **off**.

Additional trap hosts and management stations can be defined for a the community using the **add snmp community** command on page 6-25.

Examples To create an SNMP community called "public" with read only access to all MIB variables from any management station, use the command:

```
create snmp community=public open=on
```

See Also
add snmp community
Defining Management Stations Within Communities
delete snmp community
destroy snmp community
disable snmp
disable snmp community
enable snmp
enable snmp community
show snmp community

delete snmp community

Syntax

```
CREate SNmp COMmunity=name [ACCess={read|write}]
   [TRAPhost=ipadd] [MAnager={ipadd|prefix}]
   [OPen={ON|OFF|YES|NO|True|False}] [V1traphost=ipadd]
   [V2ctraphost=ipadd]
```

where:

- *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If the string contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.
- *prefix* is an IP address range in dotted decimal notation followed by a '/' character and a decimal number from 0 to 32.

Description

This command deletes a trap host or management station from the specified SNMP community.

The **community** parameter specifies the SNMP community. The community must already exist on the switch.

The **traphost**, **v1traphost** or **v2ctraphost** parameters specify a trap host to be deleted for the SNMP community. This is the IP address of a device to which traps generated by the switch are currently sent. The **traphost** or **v1traphost** parameters specify an SNMPv1 trap host. The **v2ctraphost** parameter specifies an SNMPv2c trap host. If both SNMPv1 and SNMPv2c trap hosts exist for the same trap host, you must delete the two versions separately using the same **delete** command.

The **manager** parameter can specify either a single management station (by specifying a single IP address) or a range of management stations (by using the '/' character followed a decimal number from 0 to 32 to specify a range of IP addresses) for this SNMP community.

Examples

To delete the host 192.168.1.1 as a trap host from the community named Administration, use the command:

```
del sn com=administration traphost=192.168.1.1
```

See Also

add snmp community
create snmp community
Defining Management Stations Within Communities
destroy snmp community
disable snmp community
enable snmp community
show snmp community

delete snmp group

Syntax `DELete SNmp GROup=`*`group-name`*

where *group-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *group-name* contains spaces, it must be in double quotes.

Description This command deletes an SNMP group, and is used with SNMP version 3 only.

The **group** parameter specifies the SNMP group. A group with the specified name must already exist in the router.

Examples To delete SNMP group "usergroup", use the command:

```
delete snmp group=usergroup
```

See Also **set snmp group**
show snmp group

delete snmp targetaddr

Syntax `DELete SNmp TARgetaddr=`*`address-name`*

where *address-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *address-name* contains spaces, it must be in double quotes.

Description This command deletes the SNMP target address entry from the *snmpTargetAddrTable*, and is used with SNMP version 3 only.

The **targetaddr** parameter specifies an SNMP target address entry in the *snmpTargetAddrTable*. An SNMP target address entry with the specified name must exist in the router.

Examples To delete SNMP target address "target", use the command:

```
deleted snmp targetaddr=target
```

See Also **set snmp targetaddr**
show snmp targetaddr

delete snmp targetparams

Syntax `DELete SNmp TARGETParams=`*`params-name`*

where *params-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *params-name* contains spaces, it must be in double quotes.

Description This command deletes the SNMP target params entry from the *snmpTargetParamsTable*, and is used with SNMP version 3 only.

The **targetparams** parameter specifies an SNMP target params entry in the *snmpTargetParamsTable*. An SNMP target params entry with the specified name must exist in the router.

Examples To delete SNMP target params "params", use the command:

```
deleted snmp targetparams=params
```

See Also **add snmp targetparams**
set snmp targetparams
show snmp targetparams

delete snmp user

Syntax `DELete SNmp USer=`*`user-name`*

where *user-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *user-name* contains spaces, it must be in double quotes.

Description This command deletes an SNMP user and is used with SNMP version 3 only.

The **user** parameter specifies the SNMP user. A user with the specified name must already exist in the router.

Examples To delete the SNMP user “noauthuser,” use the command:

```
delete snmp user=noauthuser
```

See Also **set snmp user**
show snmp user

delete snmp view

Syntax

```
DELete SNmp VIEw=view-name
   {OID=oid-tree|MIB={mib-name|ALL}}
```

where:

- *view-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *view-name* contains spaces, it must be in double quotes.
- *oid-tree* is character string in a format of decimal and dot that is from 1 to 32 sub-identifiers long.
- *mib-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *mib-name* contains spaces, it must be in double quotes.

Description

This command deletes the specified pair of SNMP view and the sub-tree. A specified pair must already exist in the switch.

The **view** parameter specifies the SNMP view. A view with the specified name must already exist in the switch.

The **oid** parameter specifies the object identifier of the ASN.1 sub-tree to be included or excluded from the view. The sub-tree has to be specified as a character string consisting of numbers, for example, 1.3.6.1.2.1. If this parameter is specified, the **mib** parameter cannot be specified. A sub-tree with the specified **oid** must already exist in the switch.

The **mib** parameter specifies a predefined MIB node to be included or excluded from the view. If this parameter is specified, the **oid** parameter cannot be specified. A sub-tree with the specified MIB node must already exist in the router. If **all** is specified, then all pairs of the specified SNMP view and associated the sub-trees are deleted.

Examples

To delete SNMP view “mib2view”, with all associated sub-trees, use the command:

```
delete snmp view=mib2view mib=all
```

See Also

add snmp view
show snmp view

destroy snmp community

Syntax `DESTroy SNmp COMmunity=`*`name`*

where *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If *name* contains spaces, it must be in double quotes.

Description This command destroys an existing SNMP community.

The **community** parameter specifies the SNMP community. The community must already exist on the switch.

See Also
add snmp community
create snmp community
disable snmp community
enable snmp community
show snmp community

disable snmp

Syntax `DISable SNmp`

Description This command disables the switch's SNMP agent, and disables the RMON MIB from gathering information (Appendix C, SNMP MIBs). SNMP packets sent to the switch are treated as unknown protocol packets by the underlying transport layer (UDP) and traps are not generated by the switch.

See Also
disable snmp community
enable snmp
enable snmp community
show snmp community

disable snmp authenticate_trap

Syntax `DISable SNmp AUTHenticate_trap`

Description This command disables the generation of authentication failure traps by the SNMP agent whenever an SNMP authentication failure occurs.

See Also **disable snmp**
enable snmp
enable snmp authenticate_trap
show snmp community

disable snmp community

Syntax `DISable SNmp COMmunity=`*`name`* `TRAP`

where *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If *name* contains spaces, it must be in double quotes.

Description This command disables a particular SNMP community or disables the generation of trap messages for the community.

The **community** parameter specifies the SNMP community. The community must already exist on the switch. When a community is disabled, packets for the community are processed as if the community does not exist and traps are not generated for the community. The SNMP agent generates an authentication error if a packet is received for a disabled community.

The **trap** parameter specifies that only traps for the community should be disabled, not the entire operation of the community. Trap messages are not sent to the community's trap host(s), but all other SNMP operations proceed normally.

See Also **disable snmp**
enable snmp
enable snmp community
show snmp community
show snmp community

enable snmp

Syntax `ENAble SNmp`

Description This command enables the switch's SNMP agent. The SNMP agent receives and processes SNMP packets sent to the switch and generate traps.

This command enables the switch's SNMP agent, and enables the RMON MIB to gather information (Appendix C, SNMP MIBs). The SNMP agent receives and processes SNMP packets sent to the switch and generate traps.

By default, the SNMP agent is disabled. This command is required to enable SNMP to operate at boot.

See Also **disable snmp**
disable snmp community
enable snmp community
show snmp community

enable snmp authenticate_trap

Syntax `ENAble SNmp AUTHenticate_trap`

Description This command enables the generation of authentication failure traps by the SNMP agent whenever an SNMP authentication failure occurs.

By default, the generation of authentication traps is disabled. This command is required to enable SNMP authentication failure traps at boot.

See Also **disable snmp**
disable snmp authenticate_trap
enable snmp
show snmp community

enable snmp community

Syntax `ENAble SNmp COMmunity=`*`name`* `TRap`

where *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If *name* contains spaces, it must be in double quotes.

Description This command enables a particular SNMP community or enables the generation of trap messages for the community. This command is accepted only when the user has Security Officer privilege.

The **community** parameter specifies the SNMP community. The community must already exist on the switch. When a community is enabled, the SNMP agent processes SNMP packets for the community and generates traps to trap hosts in the community, if traps are also enabled. SNMP communities are enabled when they are created, but traps are not enabled for the community.

The **trap** parameter specifies that only traps for the community should be enabled, not the entire operation of the community. Trap messages are sent to the community's trap host(s).

Examples To create an SNMP community and enable traps on it, use the following commands:

```
create snmp community=private traphost=192.168.1.1
   manager=192.168.1.1
enable snmp community=private trap
```

See Also **disable snmp**
disable snmp community
enable snmp
show snmp community

purge snmp

Syntax `PURge SNmp`

Description This command disables the SNMP agent, and deletes all SNMP configuration elements (communities, groups, target addresses, users and views). This command applies to all versions of SNMP (v1, v2 and v3)

See Also disable snmp

set snmp community

Syntax

```
SET SNmp COMmunity=name [ACCess={Read|Write}]
   [OPen={ON|OFF|YES|NO|True|False}]
```

where *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If *name* contains spaces, it must be in double quotes.

Description

This command modifies the access mode and open access configuration for the specified SNMP community. This command is accepted only when the user has Security Officer privilege.

The **community** parameter specifies the name of the community. A community with the specified name must already exist in the switch.

The **access** parameter specifies the access mode for this community. If **read** is specified, management stations in this community can only read MIB variables from the switch, that is perform SNMP *get* or *get-next* operations. If **write** is specified, management stations in this community can read and write MIB variables, that is perform SNMP *set*, *get* and *get-next* operations. The default is **read**.

The **open** parameter allows access to this community by any management station, and overrides the management stations defined with the **manager** parameter. The default is **off**.

Examples

To disable access from any management station for an SNMP community called “public”, use the command:

```
set snmp community=public open=off
```

See Also

create snmp community
destroy snmp community
show snmp community

set snmp engineid

Syntax `SET SNmp ENGineid=`*`snmpEngineID`*

where *snmpEngineID* is a string of hexadecimal characters having a minimum length of 5 octets and maximum length of 32 octets. There are two hexadecimal characters in each octet.

Description This command modifies the local *snmpEngineID*, and is used with SNMP version 3 only.

The **engineid** parameter specifies the *snmpEngineID*. The value for this object shall not be all zeros or ‘ff’H or the empty (zero length) string.

A user's password (entered on the command line) is converted to an MD5 or SHA security digest. This digest is based on both the password and the local engine ID. The command line password is then destroyed, as required by RFC 3414. Because of this deletion, if the local value of engineID changes, the security digests of SNMPv3 users are invalid, and the users must be recreated.

The default engine ID is a string of 11 octets and is constructed as follows:

- Octets 1 through 4 represent the agent's SNMP management private enterprise number as assigned by the Internet Assigned Numbers Authority (IANA). For example, “000000CF” (decimal 207) is the enterprise number for Allied Telesyn in hexadecimal. The most significant bit of octet 1 is always equal to "1", so the first four octets in the default SNMP engine ID is always “800000CF”.
- Octet 5 is always 03 in hexadecimal and indicates that the next set of values represent a MAC address. Octets 6 through 11 comprise the MAC address. The Engine ID must be a unique number among the various SNMP engines in the management domain. Using the default engine ID based on MAC addresses should ensure this uniqueness.

MAC addresses are initially assigned to an interface card by its manufacturer and should be unique. However, it is possible for users to change these values. This text assumes that the MAC addresses contained within the MAC address table are those originally supplied by the card manufacturer.

Examples To add or modify an SNMP local engine ID containing the AlliedTelesyn IANA enterprise number and a MAC address of 00-00-CD-12-34-56, use the command:

```
set snmp engineid=800000cf030000cd123456
```

In this example, the IANA enterprise number, with initial bit set to 1 is 800000CF, the fifth octet is 03, and the MAC address is 0000CD123456.

See Also show snmp

set snmp group

Syntax

```
SET SNmp GROup=group-name [READview=view-name]
   [WRITEview=view-name] [NOTIFyview=view-name]
```

where:

- *group-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *group-name* contains spaces, it must be in double quotes.
- *view-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *group-name* contains spaces, it must be in double quotes.

Description

This command modifies an SNMP group parameters, and is used with SNMP version 3 only.

The **group** parameter specifies the SNMP group. A group with the specified name must already exist in the router.

The **readview** parameter specifies the MIB contents that this group can read.

The **writeview** parameter specifies the MIB contents that this group can modify.

The **notifyview** parameter specifies the notify contents that this group can receive.

Examples

To set read view “user-view” in SNMP group “usergroup”, use the command:

```
set snmp group=user-group readview=user-view
```

See Also

add snmp group
delete snmp group
show snmp group

set snmp local

Syntax `SET SNmp LOCal={NONE|1..15} [VERsion={V1|V2|V3|ALL}]`

Description This command sets the local interface to be used with a particular version of SNMP. Once set, the IP address of the local interface specified is used as the source IP address for all SNMP packets of the version specified.

The **version** parameter specifies the version of SNMP packets to which the local interface applies. The default is **all**.

The **local** parameter specifies a local interface to be used as the source IP address for all packets of a particular SNMP version that the switch generates and sends. The local interface IP address is also be used as the SNMP agent IP address in these outgoing packets. The local interface must already be configured and fall in the range 1-15. If no local interface has been set for SNMP, the switch selects a source address on the basis of the route. This means the source address is the IP address from which the SNMP packet is issued.

Examples To set the local interface 5 for SNMPv3 packets, use the command:

```
set snmp loc=5 ver=v3
```

Related Commands show snmp

set snmp targetaddr

Syntax

```
SET SNmp TARgetaddr=address-name [IP=target-ipadd]
   [UDP=udp-port] [PARams=params-name]
```

where:

- *address-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *address-name* contains spaces, it must be in double quotes.
- *target-ipadd* is an IP address in dotted decimal notation.
- *udp-port* is a decimal number from 1 to 255.
- *params-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *params-name* contains spaces, it must be in double quotes.

Description

This command modifies the SNMP target address entry in the *snmpTargetAddrTable*, and is used with SNMP version 3 only.

The **targetaddr** parameter specifies the SNMP target address entry in the *snmpTargetAddrTable*. An SNMP target address entry with the specified name must already exist in the router.

The **ip** parameter specifies the IP address to which notification are sent.

The **udp** parameter specifies the identification number of a UDP port. If this parameter is not specified then the default value of 162 is used.

The **params** parameter specifies the reference to the entry in *snmpTargetParamsTable*.

Examples

To modify the SNMP target address "target" to use the IP address "1272.0.0.1", use the command:

```
set snmp targetaddr=target ip=127.0.0.1
```

See Also

add snmp targetaddr
delete snmp targetaddr
show snmp targetaddr

set snmp targetparams

Syntax

```
SET SNmp TARGETParams=params-name
    [SECuritylevel={NOAUthnopriv|AUTHnopriv|AUTHPRiv}]
    [USer=user-name]
```

where:

- *params-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *params-name* contains spaces, it must be in double quotes.
- *user-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *user-name* contains spaces, it must be in double quotes.

Description

This command modifies the SNMP target params entry in the *snmpTargetParamsTable*, and is used with SNMP version 3 only.

The **targetparams** parameter specifies an SNMP target params entry in the snmpTargetParamsTable. An SNMP target params entry with the specified name must exist in the router.

The **securitylevel** parameter specifies the security level for this target parameters. **noauthnopriv** specifies that no authentication or privacy is used. **authnopriv** specifies that authentication be based on the MD5 or SHA algorithm. This option provides security but no privacy. **authpriv** specifies that authentication be based on the MD5 or SHA algorithm, and encryption is to be based on the DES algorithm.

The **user** parameter specifies the SNMP user. A user with the specified name does not need to exist in the router. It can be added later.

If the user's authentication and privacy protocols do not support the specified target parameters security level, then the SNMP message is not sent.

Examples

To modify the user in the SNMP target params "params" to use user "test", use command:

set snmp targetparams=*params* user=*test*

See Also

add snmp targetparams
delete snmp targetparams
show snmp targetparams

set snmp user

Syntax

```
SET SNmp USer=user-name [GROup=group-name]
   [AUTHprotocol={NONE|MD5|SHa}] [AUTHPAssword=password]
   [PRIVprotocol={NONE|DES}] [PRIVPAssword=password]
```

where:

- *user-name* is a string 1 to 32 characters long. It may contain any printable character, and is case sensitive. If *user-name* contains spaces, it must be in double quotes.
- *group-name* is a string 1 to 32 characters long. It may contain any printable character, and is case sensitive. If *group-nam*e contains spaces, it must be in double quotes.
- *password* is a string 8 to 32 characters long. It may contain any printable character, and is case sensitive. If *password* contains spaces, it must be in double quotes.

Description

This command modifies an existing SNMP user details, and is used with SNMP version 3 only.

The **user** parameter specifies the SNMP user. A user with the specified name must already exist in the switch.

The **group** parameter specifies the SNMP group. Although a group with the specified name need not pre-exist in the switch, it must be added (using the **add snmp group** command) before user access can be enabled.

The **authprotocol** parameter specifies the authentication protocol for the SNMP user (either **md5**, **sha**, or **none**).

The **authpassword** parameter specifies the authentication password for the SNMP user. If **authprotocol** is set to **md5** or **sha**, the **authpassword** must be specified.

The **privprotocol** parameter specifies privacy protocol for the SNMP user (either **des** or **none**). The **privprotocol** cannot be set to **des** if the **authprotocol** is set to **none**.

The **privpassword** parameter specifies the encryption password for the SNMP user. If **privprotocol** is set to **des**, the **privpassword** must be specified.

If authentication and privacy are turned on and you change the authentication type, i.e. **md5** to **sha**, you are prompted to re-enter a privacy password. This is because the password is used when calculating the key (or digest). However, once the key is created, the password is deleted and thus no longer available if a new key is required.

Examples

To set the authentication password for the SNMP user “Sam” to “authpassword” and to use the MD5 authentication protocol, use the command:

```
set snmp user=sam authprotocol=md5 authpassword=authpassword
```

See Also

add snmp user
delete snmp user
show snmp user

show snmp

Syntax `SHow SNmp`

Description This command displays information about the switch's SNMP agent. See Figure 6-7 on page 6-51, and Table 6-9 on page 6-52.

Figure 6-7: Example output from the **show snmp** command

```
SNMP configuration:
  Status ........................... Enabled
  Authentication failure traps .... Enabled
  Local Interface SNMPv1 .......... Not Set
  Local Interface SNMPv2 .......... local5
  Local Interface SNMPv3 .......... local2

  Community ....................... public
    Access ........................ read-only
    Status ........................ Enabled
    Traps ......................... Enabled
    Open access ................... Yes
  Community ....................... Administration
    Access ........................ read-write
    Status ........................ Disabled
    Traps ......................... Disabled
    Open access ................... No
SNMPv3 engine information
snmpEngineID ..................... 800000cf03aabbccdd11
snmpEngineBoots .................. 5
snmpEngineTime ................... 3

SNMP counters:
  inPkts ......................... 0       outPkts ......................... 0
  inBadVersions .................. 0       outTooBigs ...................... 0
  inBadCommunityNames ............ 0       outNoSuchNames .................. 0
  inBadCommunityUses ............. 0       outBadValues .................... 0
  inASNParseErrs ................. 0       outGenErrs ...................... 0
  inTooBigs ...................... 0       outGetRequests .................. 0
  inNoSuchNames .................. 0       outGetNexts ..................... 0
  inBadValues .................... 0       outSetRequests .................. 0
  inReadOnlys .................... 0       outGetResponses ................. 0
  inGenErrs ...................... 0       outTraps ........................ 0
  inTotalReqVars ................. 0
  inTotalSetVars ................. 0
  inGetRequests .................. 0
  inGetNexts ..................... 0
  inSetRequests .................. 0
  inGetResponses ................. 0
  inTraps ........................ 0

SNMPv3 counters:
UnsupportedSecLevels ............. 0       UnknownSecurityModels .............0
NotInTimeWindows ................. 0       InvalidMsgs ...................... 0
Unknown UserNames................. 0       UnknownPDUhandlers................. 0
UnknownEngineIDs ................. 0
WrongDigests ..................... 0
DecryptionErrors ................. 0
```

Table 6-9: Parameters in output of the **show snmp** command

Parameter	Meaning
Status	Whether the SNMP agent or the specified community is enabled or disabled.
Authentication failure traps	Whether the SNMP agent is enabled to generate a trap on an authentication failure for an incoming SNMP packet.
Local Interface	The interface used as the source in outgoing SNMP messages.
Community	The name of an SNMP community on the switch.
Access	Whether access rights for the SNMP community is read-only or read-write.
Status	Whether the community is enabled or disabled.
Traps	Whether the community generates traps.
Open access	Whether the SNMP community is open to access from all IP addresses.
snmpEngineID	The ID assigned to the local SNMP engineID.
snmpEngineBoots	A count of the number of times the SNMP engine has been re-booted or re-initialized since snmpEngineID was last configured.
snmpEngineTime	The number of seconds since the snmpEngineBoots counter was last incremented.
inPkts	The number of SNMP packets received by the switch.
inBadVersions	The number of SNMP packets with a bad version field received by the switch.
inBadCommunityNames	The total number of SNMP PDUs delivered to the SNMP agent that used an unknown SNMP community name.
inBadCommunityUses	The total number of SNMP PDUs delivered to the SNMP agent that represented an SNMP operation not allowed by the SNMP community name in the PDU.
inASNParseErrs	The total number of ASN.1 parsing errors, either in encoding or syntax, encountered by the SNMP agent when decoding received SNMP PDUs.
inTooBigs	The total number of valid SNMP PDUs delivered to the SNMP agent for which the value of the errorStatus component was tooBig.
inNoSuchNames	The number of SNMP packets received with an error status of nosuchname.
inBadValues	The number of SNMP packets received with an error status of badvalue.
inReadOnlys	The number of SNMP packets received with an error status of readonly.
inGenErrs	The number of SNMP packets received with an error status of generr.
inTotalReqVars	The total number of SNMP MIB objects requested.
inTotalSetVars	The total number of SNMP MIB objects which were changed.
inGetRequests	The number of SNMP Get Request packets received by the switch.

Table 6-9: Parameters in output of the **show snmp** command (Continued)

Parameter	Meaning
inGetNexts	The number of SNMP Get Next Request packets received by the switch.
inSetRequests	The number of SNMP Set Request packets received by the switch.
inGetResponses	The number of SNMP Get Response packets received by the switch.
inTraps	The number of SNMP trap message packets received by the switch.
outPkts	The number of SNMP packets transmitted by the switch.
outTooBigs	The number of SNMP packets transmitted with an error status of toobig.
outNoSuchNames	The number of SNMP packets transmitted with an error status of nosuchname.
outBadValues	The number of SNMP packets transmitted with an error status of badvalue.
outGenErrs	The number of SNMP packets transmitted with an error status of generror.
outGetRequests	The number of SNMP Get Request response packets transmitted by the switch
outGetNexts	The number of Get Next response packets transmitted by the switch.
outSetRequests	The number of Set Request packets transmitted by the switch.
outGetResponses	The number of SNMP Get response packets transmitted.
outTraps	The number of SNMP Traps transmitted by the switch.
UnknownSecurityModels	The number of SNMP packets received with a requested for unknown security level.
InvalidMsgs	The number of SNMP packets with invalid components in the SNMP message.
UnknownPDUHandlers	The number of SNMP packets with an unknown PDU.
UnsupportedSecLevels	The number of SNMP packets with an unsupported security level.
NotInTimeWindows	The number of SNMP packets outside the SNMP engine time window.
UnknownUserNames	The number of SNMP packets with an unknown user name.
UnknownEngineIDs	The number of SNMP packets with an unknown SNMP. engineID
WrongDigests	The number of SNMP packets with wrong digest value.
DecryptionErrors	The number of SNMP packets that could not be decrypted.

See Also **show snmp community**
show snmp group
show snmp user
show snmp targetaddr
show snmp view

show snmp community

Syntax `SHow SNmp COMmunity=`*`name`*

where *name* is a string 1 to 15 characters long. It may contain any printable character and is case sensitive. If the string contains spaces, it must be in double quotes.

Description This command displays information about a single SNMP community. See Figure 6-8 on page 6-54, Table 6-10 on page 6-54.

The **community** parameter specifies the name of the community. A community with the specified name must already exist in the switch.

Figure 6-8: Example output from the **show snmp community** command

```
SNMP community information:
  Name ......................... public
  Access ....................... read-only
  Status ....................... Enabled
  Traps ........................ Enabled
  Open access .................. Yes
  Manager ...................... 192.168.1.1
  Manager ...................... 192.168.5.3
  Trap host .................... 192.168.1.1
  Trap host .................... 192.168.6.23
  V2c trap host ................ 192.168.6.23
```

Table 6-10: Parameters in output of the **show snmp community** command

Parameter	Meaning
Name	The name of the community. This identifies the community and appears in SNMP messages for this community.
Access	Whether access rights for the SNMP community is read-only or read-write.
Status	Whether the community is enabled or disabled.
Traps	Whether the community generates trap messages.
Open access	Whether the community is open to access from all IP addresses.
Manager	The IP address of a management station or IP range of stations that can access this switch using this community.
Trap host	The IP address of a trap host to which traps for this community is sent.
V2c trap host	The IP address of a trap host to which SNMP version 2c traps for this community is sent.

See Also **show snmp**
show snmp group
show snmp user
show snmp targetaddr
show snmp view

show snmp group

Syntax `SHow SNmp GROup[=group-name]`

where *group-name* is a string 1 to 32 characters long. It may contain any printable character, and is case sensitive. If *group-name* contains spaces, it must be in double quotes.

Description This command is used with SNMP version 3 only, and displays information about a one or more SNMP groups. See Figure 6-9 on page 6-55 and Table 6-11 on page 6-55.

The **group** parameter specifies the SNMP group. A group with the specified name must already exist in the router. If the group-name is not specified, then all existing groups are presented.

Figure 6-9: Example output from the **show snmp group** command

```
SNMP group information:
  Group Name ............... usergroup
    Security Level ........ noAuthNoPriv
    Read View ............. read-view
    Write View ............ write-view
    Notification View ..... notification-view
    Row Status..............active
```

Table 6-11: Parameters in output of the SHOW **snmp group** command

Parameter	Meaning
Group Name	The name of the SNMP group.
MinimumSecurity Level	The minimum security level for the SNMP group; either "noAuthNoPriv", "authNoPriv" or "authPriv".
Read View	The name of a predefined view that gives read rights to the members of the specified SNMP group.
Write View	The name of a predefined view that gives write rights to the members of the specified SNMP group.
Notification View	The name of a predefined view that gives notification rights to the members of the specified SNMP group.
Row Status	The status of the displayed group, either "active", "notInService" or "notReady".

Examples To display information on the group called "adminusers", use the command:

```
show snmp group=adminusers
```

To display information on all groups, use the command:

```
show snmp group
```

See Also
add snmp group
delete snmp group
set snmp group

show snmp targetaddr

Syntax `SHow SNmp TARgetaddr[=`*`address-name`*`]`

where *address-name* is a string 1 to 32 characters long. It may contain any printable character, and is case sensitive. If *address-name* contains spaces, it must be in double quotes.

Description This command displays information about a one of more SNMP target addresses, and is used with SNMP version 3 only. See Figure 6-10 on page 6-56 and Table 6-12 on page 6-56.

The **targetaddr** parameter specifies the SNMP target address name. A target address with the specified name must already exist in the router. If the target address-name is not specified, then all existing target addresses are presented.

Figure 6-10: Example output from the **show snmp group** command

```
SNMP target addresss information:
  Target Address Name ..... target
    IP address ............ 172.20.70.1
    UDP port .............. 162
    Target Address Params.. user
    Row Status ............ active
```

Table 6-12: Parameters in output of the **show snmp targetaddr** command

Parameter	Meaning
Target Address Name	The name of the SNMP target address
IP address	The IP address of SNMP target address
UDP port	The UDP port of SNMP target address
Target Address Parameter	The reference to the entry specified in the *snmpTargetParamsTable*
Row Status	The status of the displayed SNMP target address, either "active", "notInService", or "notReady"

Example To show the target address for the target called "Adminserver" use the command:

```
show snmp targetaddr=adminserver
```

See Also **add snmp targetaddr**
delete snmp targetaddr
set snmp targetaddr

show snmp targetparams

Syntax `SHow SNmp TARGETParams=[`*`params-name`*`]`

where *params-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *params-name* contains spaces, it must be in double quotes.

Description This command displays information about a single SNMP target params and is used on version 3 only. See Figure 6-11 on page 6-57, and Table 6-13 on page 6-57.

The **targetparams** parameter specifies a SNMP target params entry in the *snmpTargetParamsTable*. A SNMP target params entry with the specified name must exist in the router. If the target *params-name* is not specified, all existing target params are presented.

Figure 6-11: Example output from the **show smnp targetparams** command

```
SNMP target params information:
  Target Params Name ..... params
    Security Level ....... authPriv
    User Name ............ test
    Row Status ........... active
```

Table 6-13: Parameters in output of the **show snmp group** command

Parameter	Meaning
Target Params Name	The name of the SNMP target parameters
Security Level	The security level for the SNMP messages to be sent to the target address; either "noAuthNoPriv", "authNoPriv" or "authPrive"
User Name	The name of user who receives notification on target address
Row Status	The status of the displayed SNMP target address eitherf "active", "notInService" or "notReady"

Example To display the target parameters for the target called “Adminserverparams” use the command:

```
sh sn targetp=adminserverparams
```

See Also **delete snmp targetparams**
set snmp targetparams

show snmp user

Syntax `SHow SNmp USer[=`*`user-name`*`]`

where *user-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *user-name* contains spaces, it must be in double quotes.

Description This command displays information about one or more SNMP users, and is used with SNMP version 3 only. See Figure 6-12 on page 6-58, and Table 6-11 on page 6-55.

The **user** parameter specifies the SNMP user. A user with the specified name must already exist in the router. If the user-name is not specified, then all existing users are presented.

Figure 6-12: Example output from the **show snmp user** command

```
SNMP user information:
  User Name ............... user
    Group Name ............ usergroup
    Auth Protocol ......... MD5
    Priv Protocol ......... DES
    Row Status ............ active
```

Table 6-14: Parameters in output of the **show snmp user** command

Parameter	Meaning
User Name	Name of the SNMP user.
Group Name	Name of the SNMP group.
Auth Protocol	Whether the authentication protocol for the SNMP user is MD5, SHA, or none.
Priv Protocol	Whether the privacy protocol for the SNMP user is DES or none.
Row Status	Whether the status of the displayed SNMP user is active, not in service, or not ready.

Example To display the details for a user called “Sam”, use the command:

```
show snmp user=sam
```

To display the details of all users, use the command:

```
show snmp user
```

See Also **add snmp user**
delete snmp user
set snmp user

show snmp view

Syntax `SHow SNmp VIEW[=view-name]`

where *view-name* is a string 1 to 32 characters long. It may contain any printable character and is case sensitive. If *view-name* contains spaces, it must be in double quotes.

Description This command displays information about specified SNMP view. See Figure 6-13 on page 6-59, and Figure 6-14 on page 6-59.

The **view** parameter specifies the SNMP view. A view with the specified name must already exist in the router. If the view-name is not specified the list of all existing views are presented.

Figure 6-13: Example output from the **show snmp view** command

```
SNMP View information:
  SNMP View Name(s):
    readview
    writeview
    myview
```

Figure 6-14: Example output from the **show snmp view**=*readview* command.

```
View Name ........ readview
  OID ............ 1.3.6.1.2.1
    MIB .......... mib-2
    Type ......... include
    Row Status ... active
  OID ............ 1.3.6.1.2.1.16
    MIB .......... rmon
    Type ......... exclude
    Row Status ... active
  OID ............ 1.3.6.5.6.7.8
    MIB .......... -
    Type ......... exclude
    Row Status ... active
```

Table 6-15: Parameters in output of the **show snmp view** command

Parameter	Meaning
View name	Name of the SNMP view.
OID	Object identifier of the sub-tree to be included or excluded from the view.
MIB	Predefined MIB name of the sub-tree.
Type	Whether the type for this sub-tree is include or exclude.
Row Status	Whether the status of the displayed SNMP view is active, not in service, or not ready.

Example To display the view details for the view called “Myview” use the command:

```
show snmp view=myview
```

To display the details of all views, use the command:

```
show snmp view
```

See Also **add snmp view**
delete snmp view
show snmp view

Chapter 7

Stacking

Introduction 7-2
What is Stacking? 7-2
How Stacking Works 7-4
Configuring a Stack 7-6
Configuration Example 7-14
Command Reference 7-15
- Host-Directed Commands (HDC) 7-16
- add stack interface 7-17
- copy 7-18
- delete stack interface 7-19
- disable stack 7-19
- disable stack debug 7-20
- enable stack 7-21
- enable stack debug 7-22
- set stack authentication 7-23
- set stack stackid 7-24
- set system hostid 7-25
- show stack 7-26

Introduction

This chapter describes the Stacking feature, including how it functions and how to configure it.

Benefits

Stacking affords the following advantages when managing a group of switches:

- Because stack members are connected by open standard Ethernet or uplink switch ports, the switches can be at the same physical location or across geographical areas.
- Management interfaces are conserved because each stack is managed from a single IP address or terminal connection.
- Because a stack has one configuration file that is simple to maintain for all member switches, it efficiently manages individual switches. Stacks are easy to reconfigure in tune with changing network needs.
- Stacks offer an alternative to managing a group of switches by using a CLI or GUI on each switch, which is often tedious and time-consuming.

What is Stacking?

Definition

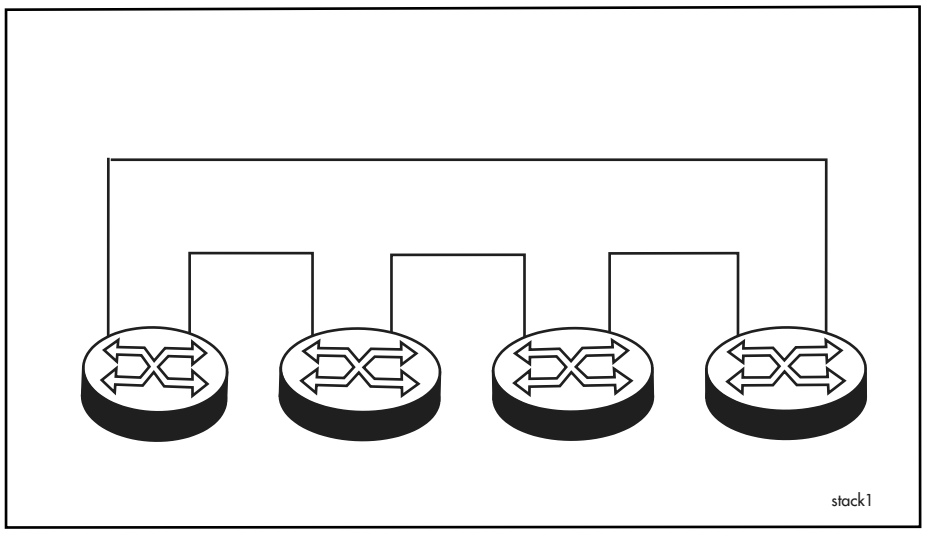
Stacking is a way to synchronise information across multiple switches and manage them as one logical device. Stacking uses a proprietary protocol to manage a group of separate switches as one.

When several switches perform similar functions, you can manage them as one. For ease and simplicity, a stack can be managed from any stack member.

Topologies

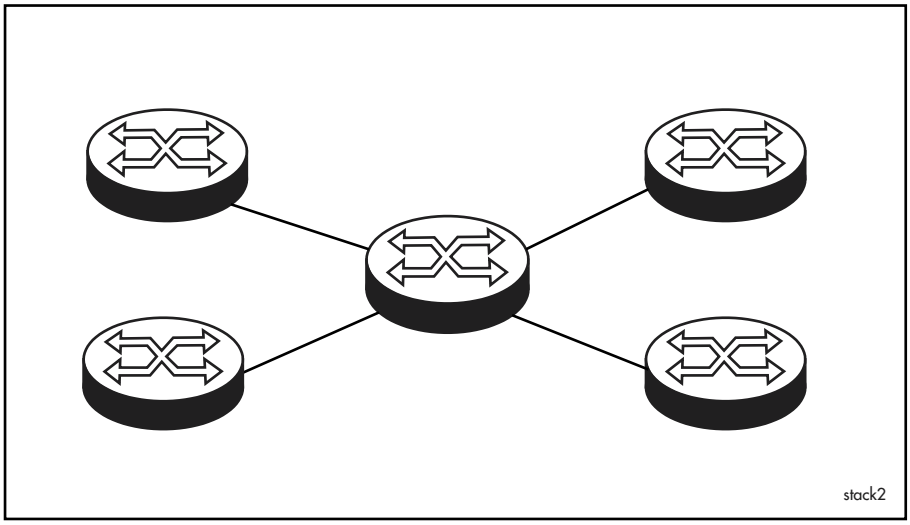
A *stack* consists of a maximum of nine switches connected by switch ports in the same Stacking VLAN. Stack members must be on the same LAN; however, they can be in different physical locations. No extra hardware is required because stack members use open standards interfaces. This allows flexible topologies; typical ones are ring and star.

Figure 7-1: Ring Topology



Ring architecture allows path redundancy; the stack can be managed even if a link in it becomes unavailable. If a ring is used, RSTP (Rapid Spanning Tree Protocol) must be configured on stacking ports.

Figure 7-2: Star Topology



Star architecture lends itself to stacking being deployed on an existing Layer 2 network that does not require redundant paths for management.

How Stacking Works

Shared information

The Stacking feature centralises management by distributing and maintaining system-wide information about stack members. It also:

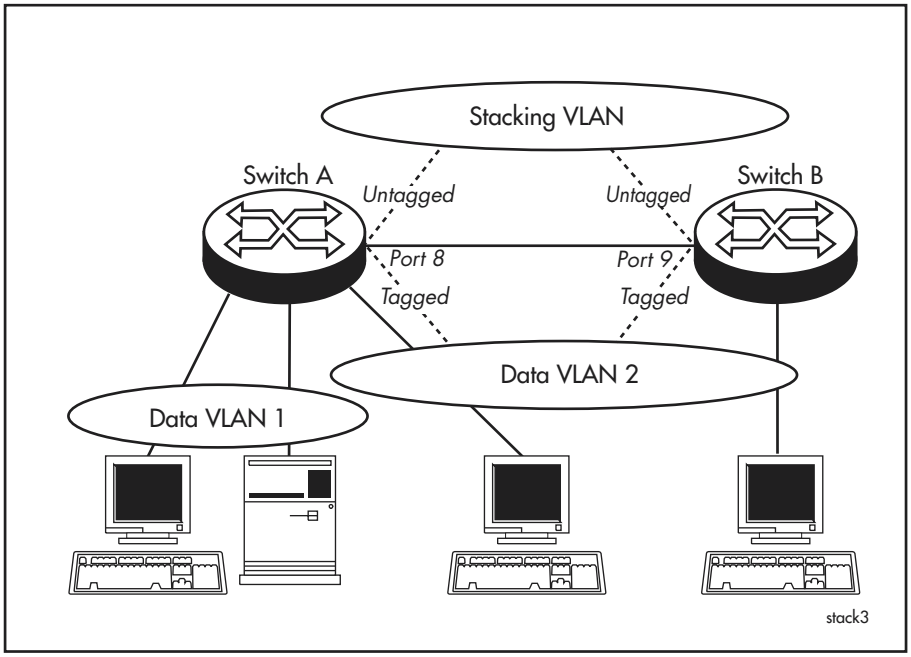
- Synchronises and propagates information about individual stack members
- Propagates CLI and GUI commands
- Manages responses and acknowledgements
- Synchronises the stack configuration file

Switches have individual *host IDs*, which you set, so that they know which device they are in the stack. Stacks have unique *stack IDs*, which you also set, so that switches know to which stack they belong. Stack IDs are essential when you have multiple stacks.

Stacked switches communicate with each other over a *Stacking interface*, which is a user-defined virtual interface such as a VLAN. Ports in the Stacking VLAN should be added as tagged VLAN ports to data VLANs. This ensures that the Stacking VLAN carries user data.

In Figure 7-3, data from Data VLAN 2 shares Ports 8 and 9 because they are untagged ports on the Stacking VLAN and tagged ports on the data VLAN.

Figure 7-3: VLAN Ports for Stacking



See “VLAN Tagging” in the Switching chapter for more information.

Process walkthrough

The following table summarises what happens after switches are configured as a stack and the stack is operational.

Stage	Description
1	When a stack is created, the user defines a virtual interface so that stack members can send each other stacking messages. When stacked switches begin transmitting over the interface, the members discover each other and synchronise to a single time. They learn information such as individual host IDs, network configuration, and set-up. When information changes, it is updated on all members so that all members have their own up-to-date shared information.
2	Stack members learn the name of the stack configuration file. The dynamic configurations of all the stack members merge to create a stack configuration script. This script is used as the boot script for the entire stack. The most recent version of the stack configuration file is synchronised across the stack.
3	The user enters the **create config** command to add the local configuration for a new member to the stack configuration file. The stack configuration is written in a file on the switch where the user enters this command. Even when a switch is added to a stack, users can independently manage it with Host-Directed Commands (HDC). The stack synchronises to the member with the lowest host ID. When a user sets the time through SNMP, NTP, or the CLI, the stack synchronises to that time and periodically checks the time thereafter.
4	When stack members are added or removed, the user enters the **set config** command to tag the stack configuration file as one that must be synchronised across the stack. When a stack member modifies the stack configuration file, the change goes to all members.
5	When a user enters a command that produces output, such as a **show** command, stack members return output to the member that first sent the command.This member then compiles the output. When it is the same, the member merges it and displays it. When the output differs, the member displays it as separate output for each member.

Local commands

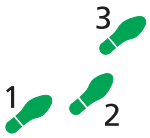
Most commands are propagated throughout the stack **except** for the following, which are local on individual stack members:

- Add Stack Interface
- Clear Flash
- Clear
- Copy
- Create Configuration
- Delete Stack Interface
- Disable Stack
- Enable Stack
- Help
- Set System HostID (other Set System commands are stack-wide)
- Set Time
- Show Ffile
- Show Ffile Check
- Telnet

Configuring a Stack

Planning

- Stacked devices must be within the same L2 forwarding domain.
- Stack members must have the same software release level. New releases should be loaded individually on members and without stacking enabled.
- Regardless of whether a stack is managed from an asynchronous port or an IP address, the stack member from which you manage it must have at least one port in a VLAN that has an IP address. Some protocols, for example RIP and OSPF, require each stack member to have a different IP address.
- The same Stacking VLAN must be created on every switch in the stack. This VLAN should not be used for other data. Stacking ports should be untagged members of the Stacking VLAN.
- To share data VLANs across a stack, ports in the Stacking VLAN should be tagged members of the data VLANs.
- If you need redundancy, use a ring topology. However, do not connect the ring until after you configure and enable RSTP (Rapid Spanning Tree Protocol) on each switch in the stack.
- When configuring SNMP on an existing stack, specify a list of host IDs before each SNMP command.
- If firewall protection is required, connect a separate firewall device as the external gateway for the stack.
- A switch can be a member of only one stack at a time.
- It is good practice to label the outside cases with individual stack IDs and host IDs after you have added switches to a stack.



To create a stack

Before configuring features for the whole stack, use the following steps to configure stacking on **each** switch meant for the stack. This procedure assumes the switch is not pre-configured.

1. Set the system host ID on the switch.

The host ID uniquely identifies this stack member in the stack. Give each switch a different host ID by using the command:

```
set system hostid=1..32
```

2. Create a VLAN.

Create a VLAN to be used for stacking. The same VLAN must be on every switch in the stack. Use the command:

```
create vlan=vlan-name vid=vid
```

where:

- *vlan-name is* a unique name from 1 to 32 characters.
- *vid* is a unique number from 2 to 4094.

See "Virtual Local Area Networks (VLANs)" in the Switching chapter for details about VLANs, including procedures for creating them.

3. **Add ports to the VLAN.**

   Add ports to the Stacking VLAN on the switch by using the command:

   ```
   add vlan=vid port=port-list
   ```

   where *port-list is* a:

   - port number or
   - hyphen-separated range of port numbers or
   - comma-separated list of port numbers and/or ranges.

   Port numbers in the Stacking VLAN are different for each switch; the port number format is *hostid.0.port*, for example, 1.0.24.

   If desired, check the VLAN configuration by using the **show vlan** command.

4. **Enable the Rapid Spanning Tree Protocol (RSTP) for rings.**

   If the stack is in a ring formation, enable RSTP on the Stacking VLAN to prevent traffic loops. Use the commands:

   ```
   create stp=stp-name
   enable stp=stp-name
   add stp=stp-name vlan=vid
   set stp=stp-name mode=rapid
   ```

   See "Configuring STP" in the Switching chapter for details about these commands.

5. **Add the VLAN as a Stacking interface.**

   Add an interface so that stack members can communicate with each other. Use the command:

   ```
   add stack interface=interface
   ```

6. **Set authentication for the stack.**

   If desired, set security for the stack member. The authentication type and password must be the same for all stack members. A stack member ignores messages from switches with a different authentication type and password. Use the command:

   ```
   set stack authentication={md5|plaintext} password=password
   ```

7. **Give the stack a unique identifier.**

   When there is more than one stack on the same network, give them different identifiers. Use the command:

   ```
   set stack stackid=1..16
   ```

   If you later want to change the stack ID for an *individual* switch, disable stacking on it, set the stack ID, and then enable stacking on it again.

   NOTE As you configure each subsequent switch, physically connect it to the port that you configured in Step 3.

8. **Enable stacking on the switch.**

   This command is stored in the stack configuration file and is activated if the switch is restarted. Use the command:

   ```
   enable stack
   ```

   Stack-wide commands are propagated after this point.

9. **Verify discovery.**

   Wait briefly for stack members to discover each other and use the following command to check that all host IDs are present:

   ```
   show stack
   ```

10. **Optionally set the correct time.**

    Switches automatically synchronise to the time on one device; however, this may not be the correct time. From any stack member, verify the time and set it as necessary by using the commands:

    ```
    show time
    set time
    ```

11. **Create and use a stack configuration file for the switch.**

    Create a stack configuration file and update the local configuration file with it. The stack shares the stacking script, which is synchronised across the stack. When a switch modifies the stack configuration file, the change goes to all members.

    Use the commands:

    ```
    create config=stack-filename.cfg
    set config=stack-filename.cfg
    ```

    If desired, check the configuration by using the **show config dynamic** command.

12. **After the first switch, set the others so that they use the stack configuration file.**

    For subsequent switches, execute the stack configuration file and enable stacking with the command:

    ```
    hostid: restart switch
    ```

Next Steps

You are now ready to configure the stack using other commands in the Software Reference. With the exception of the local commands previously noted in this chapter, commands are propagated to all members of the stack.

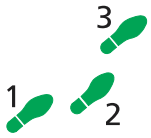
Tips

After stacking is enabled, remember to specify ports in the *hostid.0.port* format.

When you create VLANs for user data, add ports in the Stacking VLAN as tagged ports. This ensures that ports for stacking messages also carry user data.

To configure services on one stack member, a range, or a list, use the Host-Directed Command (HDC) feature whereby you send commands to specific members by typing the host ID, a colon, and then the command.

To later change the stack configuration, connect to any stack member and change the stack configuration file.



To add a switch to a stack

Before you begin:

If a port is not available on an **existing** stack member, add one to the Stacking VLAN where the new switch is to be connected. Use the command:

```
add vlan=vid port=port
```

where *port* is a port number. Port numbers for stacking have the format *hostid.0.port*, for example 1.0.24.

If desired, check the VLAN configuration by using the **show vlan** command.

1. **Turn on the new switch.**

   Power up the new switch without network connections and without pre-configuring it. To ensure a clear starting configuration, use the commands:

   ```
   set config=none
   restart switch
   ```

2. **Set the system host ID on the switch.**

   The host ID uniquely identifies this stack member in the stack. You must be directly connected to the new switch and use the command:

   ```
   set system hostid=1..32
   ```

   If desired, view the host ID by using the **show system** command.

3. **Create a VLAN on the new switch.**

   Use the same VLAN as on other stack members. Use the command:

   ```
   create vlan=vlan-name vid=vid
   ```

   where:

   - *vlan-name is* a unique name from 1 to 32 characters
   - *vid* is a unique number from 2 to 4094.

   See "Virtual Local Area Networks (VLANs)" in the Switching chapter for details about VLANs, including procedures for creating them.

4. **Add ports to the VLAN.**

   Add ports to the Stacking VLAN on the switch by using the command:

   ```
   add vlan=vid port=port-list
   ```

   where *port-list* is a:

   - port number or
   - hyphen-separated range of port numbers or
   - comma-separated list of port numbers and/or ranges

   Port numbers in the Stacking VLAN are different for each switch; the port number format is *hostid.0.port*, for example, 1.0.24.

   If desired, check the VLAN configuration by using the **show vlan** command.

5. **Enable the Rapid Spanning Tree Protocol (RSTP) for rings.**

   If the stack is in a ring formation, enable RSTP on the Stacking VLAN to prevent traffic loops. Use the commands:

   ```
   create stp=stp-name
   enable stp=stp-name
   add stp=stp-name vlan=vid
   set stp=stp-name mode=rapid
   ```

   See "Configuring STP" in the Switching chapter for details about these commands.

6. **Add the VLAN as a Stacking interface.**

   Add an interface so that stack members can communicate with each other. Use the command:

   ```
   add stack interface=interface
   ```

7. **Set authentication for the stack.**

   The authentication type and password must be the same for all stack members. If security is set for the stack, set it for the new switch by using the command:

   ```
   set stack authentication={md5|plaintext} password=password
   ```

8. **Set the stack ID on the new switch.**

   Give the new switch the unique ID for the stack where it is a member. Use the command:

   ```
   set stack stackid=1..16
   ```

   If you later want to change the stack ID for an *individual* switch, disable stacking on it, set the stack ID, and then enable stacking on it again.

9. **Physically connect the new switch using Stacking ports previously configured.**

10. **Enable stacking on the switch.**

    This command is stored in the stack configuration file and is activated if the switch is restarted. Use the command:

    ```
    enable stack
    ```

11. **Verify discovery.**

    Wait briefly for stack members to discover each other and use the following command to check that all host IDs are present:

    ```
    show stack
    ```

12. **Create and use a stack configuration file for the switch.**

    Create a stack configuration file and update the local configuration file with it. The stack shares the stacking script, which is synchronised across the stack. When a switch modifies the stack configuration file, the change goes to all members. The stack configuration file name must be a maximum of 8 characters with a 3-character extension.

    Use the commands:

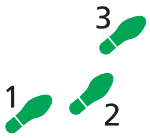
    ```
    create config=stack-filename.cfg
    set config=stack-filename.cfg
    ```

If desired, check the configuration by using the **show config dynamic** command.

13. Start the new switch as a stack member.

Execute the stack configuration file and enable stacking with the command:

```
hostid: restart switch
```



To replace a member of a stack

The replacement must be the same model of switch as the one you are replacing.

1. Turn off and unplug the switch you want to replace.

The switch is removed from the neighbour lists of the stack members.

2. Turn on the replacement switch.

Power up the new switch without network connections and without pre-configuring it. To ensure a clear starting configuration, use the commands:

```
set config=none
restart switch
```

3. Optionally set the time on the replacement.

If the host ID of the replacement will be the lowest in the stack, the stack will sychronise to the time of the replacement, which may be undesirable. Verify the time on the replacement and set it as necessary by using the commands:

```
show time
set time
```

4. Set the system host ID on the replacement.

The host ID uniquely identifies this stack member in the stack. The host ID must be the same as the switch it is replacing. Use the command:

```
set system hostid=1..32
```

5. Create a VLAN.

Use the same VLAN as on other switches in the stack. Use the command:

```
create vlan=vlan-name vid=vid
```

See "Virtual Local Area Networks (VLANs)" in the Switching chapter for details about VLANs, including procedures for creating them.

6. Add ports to the VLAN.

Add ports to the Stacking VLAN on the switch by using the command:

```
add vlan=vid port=port-list
```

where *port-list* is a:

- port number or
- hyphen-separated range of port numbers or
- comma-separated list of port numbers and/or ranges

Port numbers in the Stacking VLAN are different for each switch; the port number format is *hostid.0.port*, for example, 1.0.24.

If desired, check the VLAN configuration by using the **show vlan** command.

7. **Enable the Rapid Spanning Tree Protocol (RSTP) for rings.**

   If the stack is in a ring formation, enable RSTP on the Stacking VLAN to prevent traffic loops. Use the commands:

   ```
   create stp=stp-name
   enable stp=stp-name
   add stp=stp-name vlan=vid
   set stp=stp-name mode=rapid
   ```

   See "Configuring STP" in the Switching chapter for details about these commands.

8. **Add the VLAN as a Stacking interface.**

   Add an interface so that stack members can communicate with each other. Use the command:

   ```
   add stack interface=interface
   ```

9. **Set authentication for the stack.**

   The authentication type and password must be the same for all stack members. If security is set for the stack, set it for the new switch by using the command:

   ```
   set stack authentication={md5|plaintext} password=password
   ```

10. **Set the stack ID on the replacement switch.**

    Give the new switch the unique ID for the stack where it is a member. Use the command:

    ```
    set stack stackid=1..16
    ```

    If you later want to change the stack ID for an *individual* switch, disable stacking on it, set the stack ID, and then enable stacking on it again.

11. **Set the correct stack configuration file.**

    Ensure the most current stack configuration file is loaded on the replacement when it joins the stack. Use the commands:

    ```
    set config=stack-filename.cfg
    ```

    If the file name already exists on the switch, remove the old file with the command:

    ```
    delete file=stack-filename.cfg
    ```

12. **Enable stacking on the replacement switch.**

    The stack configuration file is sent to the replacement switch and it joins the stack when you use the command:

    ```
    enable stack
    ```

    Because this command is stored in the stack configuration file, it is activated if you restart the switch.

    NOTE Connect the replacement switch to the stack using the same ports as the original setup.

13. **Verify discovery.**

Wait briefly for stack members to discover each other, and use the following command to check that all host IDs are present:

```
show stack
```

14. Replace serial numbers in the stack configuration file.

From any stack member, open the stack configuration file by using the command:

```
edit stack-filename.cfg
```

Look for the System Configuration section in the stack configuration file, shown in the following example:

```
#
SYSTEM configuration
#
1: set system hostid=1 serialnumber=49901563
2: set system hostid=2 serialnumber=50429430
3: set system hostid=3 serialnumber=41383493
#
```

Type the serial number of the replacement switch over the number of the switch you removed.

15. Propagate the edited stack configuration file throughout the stack.

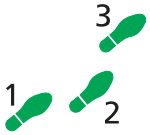
Use the command:

```
set config=stack-filename.cfg
```

16. Start the replacement switch as a stack member.

For the replacement switch to use the edited stack configuration file, use the command:

```
hostid: restart switch
```



To permanently remove a member from a stack

If you want to manage the switch remotely after removing it from the stack, ensure it has an IP address on an interface to which you can connect.

1. Disable stacking on the desired stack member.

Connect directly to an IP address or terminal for the stack member, and use the command:

```
disable stack
```

2. Run the non-stack switch as standalone.

If desired, reset port number and format so that the removed switch uses the simple port numbering format. Set the switch to an individual configuration file, such as the one it previously used or boot.cfg, and ensure it does not contain a host ID. Use the commands:

```
set config=individual-filename.cfg
```

```
restart switch
```

To add the switch to a stack again, you must directly connect to it.

3. **Create a new stack configuration file for the stack.**

   Because you removed a stack member, you must re-synchronise the stack configuration file across the stack. The stack configuration file name must be a maximum of 8 characters with a 3-character extension.

   Connect to any stack member and use the commands:

   ```
   create config=stack-filename.cfg
   set config=stack-filename.cfg
   ```

   If desired, check the configuration by using the **show config dynamic** command.

Configuration Example

This is a basic example of stacking commands to set up two directly connected switches, Switch A and Switch B. It reflects general procedures titled "To create a stack" on page 7-6.

Figure 7-4: Switch A

```
#Set the Hostid
set system hostid=1 serialnumber=40178884

#Save the running configuration
create config=switchA.cfg
set config=switchA.cfg

#Restart the switch to ensure correct port numbering
restart reboot

#Create a VLAN to be used for stacking
create vlan=stack vid=1000

#Add a port to the stacking VLAN
add vlan=stack port=1.0.25

#Set your stacking VLAN as a stacking interface
add stack interface=vlan1000

#Enable stacking on the switch
enable stack

#Create a stack configuration file
create config=stackconf.cfg

#Set the switch to the configuration file you have created
set config=stackconf.cfg
```

Figure 7-5: Switch B

```
#Set the Hostid
set system hostid=2 serialnumber=50435286

#Save the running configuration
create config=switchB.cfg
set config=switchB.cfg

#Restart the switch to ensure correct port numbering
restart reboot

#Create a VLAN to be used for stacking
create vlan=stack VID=1000

#Add a port to the stacking VLAN
add vlan=stack port=2.0.25

#Set the stacking VLAN as a stacking interface
add stack interface=vlan1000

#Enable stacking on the switch
enable stack

#Create a stack configuration file
create config=stackconf.cfg

#Set the switch to the configuration file you have created
set config=stackconf.cfg

#After both switches are connected using pre-defined stacking
ports, verify that all host IDs are present
show stack
```

Command Reference

This section describes the commands available on the switch for the stacking feature.

The shortest valid command is denoted by capital letters in the Syntax section.

Host-Directed Commands (HDC)

Syntax *hostid: non-local command*

where *hostid* is a:

- unique number from 1 to 32
- range of unique numbers from 1 to 32
- comma-separated list

Description The HDC capability sends a command to specific stack members instead of letting it propagate through the entire stack. HDC provides expanded flexibility when configuring large and complex L3 networks and switches are in stacks.

Example To add an IP route to the stack member with host ID 3, use the command:

```
3: add ip route=0.0.0.0 int=vlan100 next=192.168.0.1
   int=vlan1000
```

Only the stack member with host ID 3 accepts the command and adds the IP route. When you enter the **create config** command, the HDC directive is added to the configuration script for the stack.

To enable IP on hosts 1, 3, and 5, use the following command. Other stack members will report that the command is not applicable to them.

```
1,3,5: ena ip
```

To create a Test VLAN (VID 10) on stack members 1 to 5, use the following command. Other stack members will report that the command is not applicable to them.

```
1-5: cre vlan=test vid=10
```

add stack interface

Syntax `ADD STAck INTerface=`*`interface`*

where *interface* is a character string formed by concatenating the interface type, vlan, with the VID (for example, vlan1000)

Description This command adds a pre-defined interface to a local switch. This interface sends and receives Stacking messages in stacks. It can also carry user data, depending on how other VLANs are configured after the stack is operational.

The interface must already exist. To see information about the current VLAN, use the **show vlan** command in the *Switching* chapter.

Example To add the *vlan1000* interface to this stack and enable communication within the stack, use the command:

```
add sta int=vlan1000
```

Related Commands

delete stack interface
enable stack
set stack stackid
show stack

copy

Syntax When Stacking is enabled:

```
COPy [source-hostid:][device:]filename1.ext
   [dest-hostid:][device:]filename2.ext
```

where:

- *device* indicates the physical location of the file, and is either flash, CFlash, or NVS. The default is flash.
- *filename1* is the name of an existing file.
- *filename2* is a valid filename in the format [device:]filename.ext. Invalid characters are * + = " | \ [ ] ; : ? / < > and wildcards are not allowed. Valid characters are:
  - uppercase and lowercase letters
  - digits
  - ~ ' ! @ # $ % ^ &, ( ) _ - { }
- *.ext* is a 3-letter filename extension and can be any text file (for example, TXT, CFG, SCP, HLP, HTM, SPA, and MDS) **except** UKF or STK. The original file and the copy must have the same extension.
- *source-hostid* indicates the host ID of the stack member that holds the file to be copied. This variable is optional but if present, the *dest-hostid* is required; if not present, the **copy** command functions in the standard way.
- *dest-hostid* indicates stack members to receive the copied file:
  - a unique number from 1 to 32
  - a range of unique numbers from 1 to 32
  - a comma-separated list

  This variable is optional but if present, the *source-hostid* is required; if not present, the **copy** command functions in the standard way.

Description This command copies one file to another.

Example To copy file1.txt from a stack member with host ID 1 to stack members with host IDs 2 through 4, and also 6, and rename it "file2.txt", use the command:

```
cop 1:file1.txt 2-4,6:file2.txt
```

To copy file1.txt from the stack member with host ID 1 to the stack member with host ID 2 without renaming it, use the command:

```
cop 1:file.txt 2:file.txt
```

While connected to the stack member with host ID 1, use a host-directed command to direct host ID 2 to copy file.txt and rename it "file2.txt":

```
2:cop file.txt file2.txt
```

Note that you **cannot** use a host-directed command to direct the **copy** command to more than one stack member at a time. For example, "1-3: cop file.txt file2.txt" returns an error.

delete stack interface

Syntax `DELete STAck INTerface=`*`interface`*

where *interface* is a character string formed by concatenating the interface type, vlan, with the VID (for example, vlan1000)

Description This command is local and deletes an interface that previously had been added to this stack member.

Example To delete the *vlan1000* interface from a stack, use the command:

```
del sta int=vlan1000
```

Related Commands
add stack interface
disable stack
enable stack debug
show stack

disable stack

Syntax `DISable STAck`

Description This command is local and disables stacking for the switch where the command is entered. In addition to a direct connection, you can log onto the switch in a telnet or TTY session.

When a switch has been disabled, it is not visible to the stack. You must manage it through another connection, such as a telnet or TTY session, or enable stacking again through the telnet or TTY session. If you disable stacking, you do not need to reboot.

Example To disable stacking on a switch, use the command:

```
dis sta
```

Related Commands
delete stack interface
enable stack
set stack stackid
show stack

disable stack debug

Syntax

```
DISable STAck DEBug[={ALL|STAte|PACket|INFO}]
   [DETail={SUMmary|FULL|RAW}]
```

Description

This command disables stacking debugging on individual stack members, not the entire stack. To disable the debugging feature, you must be logged onto the stack member in a telnet or TTY session. The following table explains the parameters.

Parameter	Value	Description
DEBug		Specifies debug options to disable.
	ALL (default)	Includes Info, State, and Packet (Summary).
	STAte	Stacking status changes.
	PACket	A stacking protocol packet has been sent or received.
	INfo	General information about stacking.
DETail		Specifies details for the **disable stack debug=packet** command only.
	SUMmary (default)	Summary information in the packet header only.
	FULL	Summary and the full data sections.
	RAW	Full summary with the entire packet in natural hexadecimal format.
*All caps denote command shortcuts		

Example

To disable all packet debugging, use the command:

```
dis sta deb=pac
```

Related Commands

delete stack interface
disable stack
enable stack
disable stack debug
set stack stackid
show stack

enable stack

Syntax `ENAble STAck`

Description This command is local and allows stacking on a switch; it is stored in the stack configuration file. Before enabling this feature, you must have already set a host ID on the switch by using the **set system hostid** command. You must also be logged onto the stack member in a telnet or TTY session.

Example To enable stacking on a switch, use the command:

```
ena sta
```

Related Commands

add stack interface
disable stack
set stack authentication
set stack stackid
show stack

enable stack debug

Syntax

```
ENAble STAck DEBug[={ALL|STAte|PACket|INFO}]
   [DETail={SUMmary|FULL|RAW}]
```

Description

This command enables stacking debugging for individual stack members, not the entire stack, and displays information for the specific switch. When a debug option is enabled, major system stacking errors are labelled *Stk_Error*. An error log shows where the error occurred.

To enable this feature, you must be logged onto the stack member in a telnet or TTY session. The following table explains the parameters.

Parameter	Value	Description
DEBug		Specifies debug options to enable.
	ALL (default)	Includes Info, State, and Packet (Summary).
	STAte	Shows stacking status changes.
	PACket	Displays information about stacking packets received and sent.
	INfo	Gives general information about stacking.
DETail		For the **enable stack debug=packet** command only.
	SUMmary (default)	Summary information in the packet header only.
	FULL	Summary and the full data sections.
	RAW	Full summary with the entire packet in natural hexadecimal format.

*All caps denote command shortcuts

Example

To enable packet debugging with summary only, use the command:

```
ena sta deb=pack det=sum
```

Related Commands

disable stack
disable stack debug
enable stack
show stack

set stack authentication

Syntax

```
SET STAck AUthentication={NONE|PLAintext|MD5}
   [PASSword=password]
```

Description

This command sets authentication on a local switch meant for a stack. Stacking packets are then authenticated by the method and password that you set. Authentication type and password must be the same for all stack members.

Authentication is optional and if you want it, you must set it before you enable stacking on the switch. Otherwise, disable stacking on the switch, set authentication, and enable stacking again. The following table explains the parameters.

Parameter	Value	Description
AUthentication		Sets the switch so that communication between stack members is authenticated with the password you define with the command. Packets without the password are discarded when received. Switches in the same stack must have the same password.
	NONE (default)	Disables authentication.
	PLAintext	Sets a cleartext password to authenticate stacking packets.
	MD5	Sets an encrypted password to authenticate stacking packets.
PASSword	*password*	Unique string 1 to 8 characters long that authenticates stacking packets. Contains any printable characters and is case sensitive.

*All caps denote command shortcuts

Example

To set the stack so the unencrypted password “amigo” authenticates stacking packets, use the command:

```
set sta au=pla pass=amigo
```

Related Commands

add stack interface
disable stack
enable stack
set stack stackid
show stack

set stack stackid

Syntax `SET STAck STACKid=1..16`

Description This command sets a switch so that it belongs to a specific stack. Use it when changing the stack ID for the whole stack (all members) or when moving a member from one stack to another. Disable stacking on members that you want to change. Update the stack configuration file after you make changes by using the **create config** command.

When **stackid** is not for an existing stack or the default stack, then a new stack is created and given the number you specify. In this case, you get a message about the new stack. The default is 1.

Example To set the stack ID to 2, use the command:

```
set sta stack=2
```

Related Commands

add stack interface
disable stack
enable stack
enable stack debug
set stack authentication
show stack

set system hostid

Syntax `SET SYStem HOSTid={0..32} SERialnumber=`*`serial-number`*

Description This command sets a host ID for the switch as well as a new port numbering format. This command is local whereas other **set system** commands are propagated throughout a stack when the stacking feature is enabled. The switch must be restarted/rebooted to implement changes made by this command.

A stack can consist of a maximum of nine switches, and each switch must have a host ID before you can enable stacking on it. Host IDs uniquely identify each switch in its stack. The **hostid** parameter lets you specify a unique number from 1 to 32 for each stack member. This parameter also specifies the extended port numbering format for switch ports, which is hostid.board.port. The board ID is 0 (zero). Setting the host ID to 0 removes it and specifies simple port numbering (port format) for the switch ports. The default is 0.

The **serialnumber** parameter is the serial number for the switch and is usually optional; however, it is required when writing a stack configuration. It can be verified with the **show system serialnumber** command. The default is the serial number for the switch.

Example To set 25 as the host ID and 50435286 as the serial number for a switch that you want to include in a stack, use the command:

```
set sys host=25 ser=50435286
```

Related Commands **add stack interface**
enable stack

show stack

Syntax `SHow STAck [COUnters] [DATA]`

Description This command displays general information as well as debugging information about this switch and other members of the stack (Figure 7-6, Figure 7-7 on page 7-27, Table 7-2 on page 7-27).

The **counters** parameter displays various counter values associated with stacking. The **data** parameter displays information about stacking, including the status.

The following example shows a stack with four members, all with identical output.

Figure 7-6: Example output from the **show stack** command

```
Stack Host 1-4
Stacking information
------------------------------------------------------------
Module status ................. Enabled
Stack id ...................... 1
Authentication ................ MD5
Password ...................... friend
Direct connected host ......... 2
Stacking interfaces ........... vlan1000

Stack Members:
HostId MacAddress            State
---------------------------------
     1 00-00-cd-07-8a-00      up
     2 00-00-cd-02-e4-e0      up
     3 00-00-cd-00-ea-f9      up
     4 00-00-cd-02-e4-b0      up
---------------------------------
```

Table 7-1: Parameters in the output of the **show stack** command

Parameter	Meaning
Stack Host 1-2	Stack members to which the output applies.
	Stacking Information
Module Status	Whether Stacking is operative on the switch.
Stack ID	When more than one stack is on the same network, the stack ID to which this member belongs.
Authentication	The type of authentication, if any.
Password	Unique string of characters that authenticates packets.
Direct connected host	Host ID of the stack member to which you are directly connected through a telnet or TTY session.
Stacking Interfaces	The stacking interface for this stack member.
	Stack Members
Host ID	The unique ID for this stack member.
MAC Address	The physical address of this stack member on the network.
State	Whether all stack members are participating in the stack.

Figure 7-7: Example output from the **show stack counters** command

```
Manager >
Stack Host 1:

Stack Counters
-------------------------------------------------------------
General stacking counters:
  stkDebugErrors ........ 2

Stacking packet counters:
  rxStkPktTotal ....... 4871   txStkPktTotal ....... 2646
  rxStkPktDiscard ........ 0   txStkPktFail ........... 0

Stacking database counters:
  stkDbSdrCount .......... 7
-------------------------------------------------------------
```

Table 7-2: Parameters in the output of the **show stack counters** command

Parameter	Meaning
StkDebugErrors	Major stacking system errors that could affect stacking operations.
rxStkPktTotal	Total stacking packets received.
rxStkPktDiscard	Number of stacking packets received and discarded.
txStkPktTotal	Total stacking packets transmitted.
txStkPktFail	Number of stacking packets that were transmitted but failed.
StkDbSdrCount	Total stacking records.

Examples

To display stacking counters, use the command:

```
sh sta cou
```

Related Commands

add stack interface
delete stack interface
disable stack
disable stack debug
enable stack
enable stack debug
set stack stackid

Chapter 8

Port Authentication Enhancements

Overview 8-2
MAC Based Authentication 8-2
Dynamic VLAN Assignment 8-2
802.1x Guest VLANs 8-4
802.1x Authentication Methods Using RADIUS 8-4
    Steps in the Authentication Process 8-4
Command Reference 8-7
    activate portauth port reauthenticate 8-7
    disable portauth 8-7
    disable portauth debug port 8-8
    disable portauth port 8-8
    enable portauth 8-8
    enable portauth debug port 8-9
    enable portauth port 8-10
    purge portauth port 8-14
    reset portauth port 8-15
    reset portauth port multimib 8-15
    set portauth port 8-16
    set portauth port supplicantmac 8-21
    set portauth username 8-23
    show portauth 8-24
    show portauth counter 8-25
    show portauth port 8-26
    show portauth port multisupplicant 8-32
    show portauth timer 8-36
    show switch port 8-37
    show vlan 8-37

Overview

Port authentication now includes:

- "MAC Based Authentication" on page 8-2, as an alternative to the existing 802.1x authentication
- "Dynamic VLAN Assignment" on page 8-2
- "802.1x Guest VLANs" on page 8-4
- enhancements to "802.1x Authentication Methods Using RADIUS" on page 8-4
- see also "Configurable RADIUS Timers" in the Release Note for Software Release 2.7.3

MAC Based Authentication

MAC based authentication is an alternative approach to 802.1x for authenticating hosts connected to a port. It allows only client devices with specified MAC addresses to associate and pass data through an access point. Client devices with MAC addresses that are not in a list of allowed MAC addresses may not associate with the access point.You can create a list of allowed MAC addresses in the access point management system, or on a server used for MAC-based authentication.

When authenticating based on the host's source MAC address, the host is not required to run a client for the 802.1x protocol. The RADIUS server that performs the authentication also returns the VLAN id to which the host should be attached. This allows the device to add the port to the appropriate VLAN (untagged), and to hide traffic on VLANs for hosts of different security levels

MAC-based authentication allows non-802.1x capable clients such as network printers to be added securely.

The **portauth** parameter lets you configure either 802.1x or MAC based authentication on a port. If no value is specified, 802.1x is used.

Dynamic VLAN Assignment

Dynamic VLAN assignment allows a supplicant to be placed into a specific VLAN based on information returned from the RADIUS server during authentication. This limits the network access of a supplicant to a specific VLAN that is tied to their authentication, and prevents supplicants from connecting to VLANs for which they are not authorised. A port's VLAN assignment is determined by the first supplicant to be authenticated on the port.

VLAN assignment is enabled or disabled using the **vlanassignment** parameter of port authentication commands.

The Configured and Actual fields of the **show vlan** command show which ports are configured for the VLAN and which have been dynamically assigned to the VLAN.

Radius attributes

The RADIUS server provides information to the authenticator using RADIUS tunnel attributes, as defined in RFC 2868, *RADIUS Attributes for Tunnel Protocol Support*. The tunnel attributes that must be configured for VLAN assignment are:

- **Tunnel-Type**
  The protocol to be used for the tunnel specified by Tunnel-Private-Group-Id. VLAN (13) is the only supported value.
- **Tunnel-Medium-Type**
  The transport medium to be used for the tunnel specified by Tunnel-Private-Group-Id. 802 (6) is the only supported value.
- **Tunnel-Private-Group-ID**
  The ID of the tunnel the authenticated user should use. This must be the name or ID number of a VLAN on the router or switch.

These tunnel attributes are included in the Access-Accept message from the RADIUS server to the Authenticator.

Single-host mode

In single host mode, VLAN assignment is as follows:

- If authentication fails, the supplicant is denied access to the port. The port is placed in its configured access VLAN, that is, the VLAN it was set up for in the **add vlan** command.
- If the RADIUS server supplies valid VLAN information, the port is placed in the specified VLAN after configuration.
- If the RADIUS server supplies invalid VLAN information, the port is returned to the Unauthorised state, and placed in its configured access VLAN.
- If the RADIUS server supplies no VLAN information, the port is placed in its configured access VLAN after successful authentication.
- If port authentication is disabled on the port, the port is returned to its configured access VLAN.
- When the port is in the Force Authorized, Force Unauthorized or the Unauthorized state, it is placed in its configured access VLAN.

While the port is in a RADIUS server assigned VLAN, changes to the port's configured access VLAN do not take effect until the port leaves the assigned VLAN. This can occur if:

- the last authentication session on the port expires
- the link goes down
- port authentication is disabled on the port
- port authentication is disabled on the system

Multi-supplicant mode

VLAN assignment can be run in multi-supplicant mode, if the multi-supplicant mode is enabled on AT-8600, AT-8700XL, AT-8800, Rapier, Rapier, AT-8900 and AT-9900 switches. In multi-supplicant mode, the behaviour is dictated by which supplicant is authenticated first.

If the multi-supplicant mode is enabled on a port authentication port, the behaviour of the first authenticated supplicant is the same as that of a supplicant in single-supplicant mode. For all further supplicants, the **securevlan** parameter specifies the action that is taken when authenticating any supplicants after the first supplicant has authenticated. There are two possible actions:

- **securevlan=on**
  Only those supplicants with a VLAN that is the same as that of the first authenticated supplicant are authenticated. This is the default, and is the more secure action.
- **securevlan=off**
  All further authenticated supplicants are placed in the same VLAN as the first authenticated supplicant. This action is less secure.

802.1x Guest VLANs

802.1x ports can be configured with a limited access guest VLAN, which is used when no 802.1x host is currently attached to the port. This limited access VLAN is defined using the **guestvlan** parameter.

As soon as a single 802.1x packet is received on the port, it is removed from the guest VLAN, and put into its configured access VLAN in the Unauthenticated state. This effectively disables the guest VLAN on the port until the port's link goes down.

A guest VLAN can only be configured for a port that is running in single-supplicant mode.

802.1x Authentication Methods Using RADIUS

The authentication server verifies the supplicant's details, passed to it by the authenticator. This implementation of 802.1x control requires that a port acting as an authenticator must communicate with a RADIUS authentication server. The RADIUS server must be capable of receiving and deciphering EAP in RADIUS packets.

The authentication server must be connected to a port on the router or switch which does not have port authentication enabled, or is set with CONTROL=AUTHORISED.

Prior to this enhancement, the supported supplicant encryption mechanisms for communication with the RADIUS server were EAP-MD5 and EAP-OTP. With this enhancement the encryption methods supported by authenticators are EAP-MD5, EAP-OTP, EAP-TLS, EAP-TTLS, and EAP-PEAP.

Steps in the Authentication Process

Until authentication is successful, the supplicant can only access the authenticator to perform authentication message exchanges, or access services not controlled by the authenticator's controlled port.

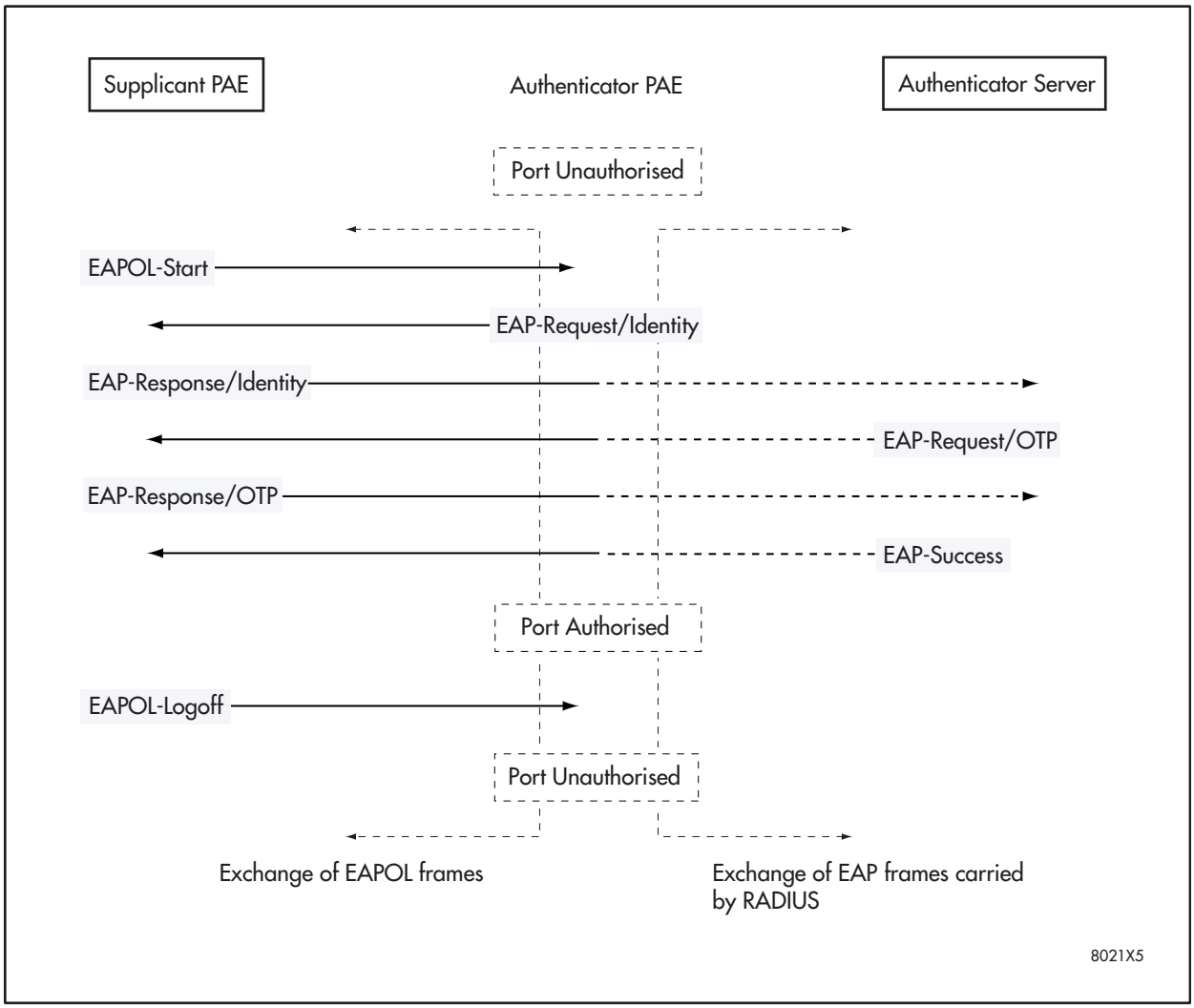
Initial 802.1x control begins with an unauthenticated supplicant and an authenticator. A port under 802.1x control acting as an authenticator is in an unauthorised state until authentication is successful.

1. Either the authenticator or the supplicant can initiate an authentication message exchange. The authenticator initiates the authentication message exchange by sending an EAPOL packet containing an encapsulated EAP-Request/Identity packet. The supplicant initiates an authentication message exchange by sending an EAPOL-Start packet, to which the authenticator responds by sending an EAPOL packet containing an encapsulated EAP-Request/Identity packet.
2. The supplicant sends an EAPOL packet containing an encapsulated EAP-Response/Identity packet to the authentication server via the authenticator, confirming its identity.
3. The authentication server selects an EAP authentication algorithm to verify the supplicant's identity, and sends an EAP-Request packet to the supplicant via the authenticator.
4. The supplicant provides its authentication credentials to the authenticator server via an EAP-Response packet.
5. The authentication server either sends an EAP-Success packet or EAP-Reject packet to the supplicant via the authenticator.
6. Upon successful authorisation of the supplicant by the authenticator server, a port under 802.1x control is in an authorised state, unless the MAC associated with the port is either physically or administratively inoperable. Also upon successful authorisation of the supplicant by the authenticator server, the supplicant is allowed full access to services offered via the controlled port. If piggybacking is enabled on the authorised authenticator port, any other device connected will also be give full access.
7. When the supplicant sends an EAPOL-Logoff message to the authenticator the port under 802.1x control is set to unauthorised.

A successful authentication message exchange, initiated and ended by a supplicant using OTP authentication, is shown in below.

Caution: To minimise the risk of denial-of-service attacks by issuing EAPOL-Logoff messages to an Authenticator Port Access Entity (PAE) from a third party device, we recommend that 802.1x not be used in a shared media LAN.

Figure 1: Authentication Messaging Exchange Initiated by the Supplicant



Command Reference

This section describes the modifications to commands.

activate portauth port reauthenticate

Syntax

```
ACTivate PORTAuth[={8021x|MACbased}] POrt={ALL|port-name}
   REAUTHENTicate [SUPPLicantmac=macadd]
```

Description

The **portauth** parameter specifies the type of port authentication to perform. If no value is specified, 802.1x is used.

The **port** parameter specifies the port where reauthentication is to occur. If **all** is specified, reauthentication occurs on all ports enabled as 802.1x authenticator Port Access Entities (PAEs) or all ports enabled as MAC based authentication PAEs, depending on the value of the **portauth** parameter.

Examples

To reauthenticate 802.1x supplicants connected to port 1, use the command:

```
act port=8021x po=1 reauthent
```

disable portauth

Syntax

```
DISable PORTAuth[={8021x|MACbased}]
```

Description

The **portauth** parameter specifies the type of port authentication to disable. If no value is specified, 802.1x is used.

Examples

To disable 802.1x functionality on the router or switch, use the command:

```
dis porta=8021x
```

disable portauth debug port

Syntax `DISable PORTAuth[={8021x|MACbased}] DEBug={ALL|PAcket|STate} POrt={ALL|port-name}`

Description The **portauth** parameter specifies the type of port authentication port for which debugging output is disabled. If no value is specified, 802.1x is used.

The **debug** parameter specifies the type of port authentication debugging to disable. If **state** is specified, the output of state changes in the state machines for the specified port(s) and port authentication type is disabled. If **packet** is specified, which is only possible if **portauth=8021x**, the output of 802.1x EAP packet headers transmitted or received by a specific port is disabled. If **all** is specified, all debug output for the specified port(s) and authentication type is disabled.

Examples To disable 802.1x port authentication state machine debug output for port 1, use the command:

```
dis porta=8021x de=st port=1
```

disable portauth port

Syntax `DISable PORTAuth[={8021x|MACbased}] [POrt={ALL|port-name}]`

Description The **portauth** parameter specifies the type of port authentication to disable. If no value is specified, 802.1x is used.

Examples To disable 802.1x activity on port 1, use the command:

```
dis porta=8021x po=1
```

enable portauth

Syntax `ENAble PORTAuth[={8021x|MACbased}]`

Description This command enables port authentication functionality on the router or switch. To configure individual ports as authenticators and/or supplicants use the **enable portauth port** command on page 8-10.

The **portauth** parameter specifies the type of port authentication to enable. If no value is specified, 802.1x is used.

Examples To enable 802.1x functionality on the router or switch, use the command:

```
ena porta=8021x
```

enable portauth debug port

Syntax `ENAble PORTAuth[={8021x|MACbased}] DEBug={ALL|PAcket|STate} POrt={ALL|`*port-name*`}`

Description The **portauth** parameter specifies the type of port authentication port for which debugging output is enabled. If no value is specified, 802.1x is used.

The **debug** parameter specifies the type of debug information displayed. If **all** is specified all debug output for the specified port(s) and authentication type is enabled. If **packet** is specified, which is only possible if **portauth=8021x,** all 802.1x related EAP and EAPOL packet headers are echoed to the router or switch console as they are received or transmitted by the specified port. The packet contents are not displayed as they may contain sensitive username and/or password information. If **state** is specified, all state machine changes for the specified port(s) and port authentication type are echoed to the console.

Examples To enable debugging of state machine transitions for MAC Based Authentication on port 1, use the command:

```
ena porta=macbased deb=st po=1
```

enable portauth port

Syntax

```
ENAble PORTAuth[=8021x] POrt={ALL|port-name}
   TYpe=AUthenticator [CONTrol={AUTHorised|AUTO|
   UNauthorised}] [MAXReq=1..10] [MODe={MUlti|SIngle}]
   [PIGgyback={TRue|FAlse}] [QUIETperiod=0..65535]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHMax=1..10]
   [REAUTHPeriod=1..86400] [SERVERTimeout=1..60]
   [SUPPTimeout=1..60][TXperiod=1..65535]
   [GUEstvlan={VLAN-id|VLAN-name|NONe}][SECurevlan={ON|
   OFF}][VLANAssignment={ENAbled|
   DIsabled}][MIBReset={ENAbled|DIsabled}] [TRap={SUCcess|
   FAILure|BOTh|NONe}]

ENAble PORTAuth[=8021x] POrt={ALL|port-name} TYpe=BOth
   [AUTHPeriod=1..60] [CONTrol={AUTHorised|UNauthorised|
   AUTO}] [HELDPeriod=0..65535] [MAXReq=1..10]
   [MAXStart=1..10] [MODe={MUlti|SIngle}]
   [PIGgyback={TRue|FAlse}] [QUIETperiod=0..65535]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHMax=1..10]
   [REAUTHPeriod=1..86400] [SERVERTimeout=1..60]
   [STARTperiod=1..60] [SUPPTimeout=1..60]
   [TXperiod=1..65535] [USERName=login-name
   PASSword=password [METhod={OTP[ENCryption={MD4|MD5}]|
   STandard}]][GUEstvlan={VLAN-id|VLAN-name|
   NONe}][SECurevlan={ON|OFF}][VLANAssignment={ENAbled|
   DIsabled}][MIBReset={ENAbled|DIsabled}] [TRap={SUCcess|
   FAILure|BOTh|NONe}]

ENAble PORTAuth[=8021x] POrt={ALL|port-name}
   TYpe=SUpplicant [AUTHPeriod=1..60]
   [HELDPeriod=0..65535] [MAXStart=1..10]
   [STARTperiod=1..60] [USERName=login-name
   PASSword=password [METhod={OTP[ENCryption={MD4|MD5}]|
   STandard}]]

ENAble PORTAuth=MACbased POrt={ALL|port-name}
   [CONTrol={AUTHorised|UNauthorised|AUTO}]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHPeriod=1..86400]
   [QUIETperiod=0..65535] [SECurevlan={ON|
   OFF}][VLANAssignment={ENAbled|
   DIsabled}][MIBReset={ENAbled|DIsabled}] [TRap={SUCcess|
   FAILure|BOTh|NONe}]
```

Description

This command enables the port authentication functionality on ports. The type of authentication to enable can be 802.1x or MAC Based Authentication. For 802.1x, a port can be enabled as an authenticator, supplicant, or both.

The **portauth** parameter specifies the type of port authentication to enable. If no value is specified, 802.1x is used.

The **type** parameter specifies whether the port is to act as an authenticator, supplicant, or both. **type** can only be specified when **portauth=8021x**. **both** cannot be specified when the **mode** parameter is **multi**.

The **authperiod** parameter specifies the period of time in seconds that the Supplicant PAE waits for a reply after sending out an EAP-Response frame to the Authenticator PAE. The **authperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If no response is received within the specified time, a new authentication attempt may start. The valid range of integer values is 1 to 60. The default is 30.

The **control** parameter specifies the state of the controlled authenticator port. The **mode** parameter is used when the **portauth** parameter specifies **macbased,** or when the **type** parameter either specifies **authenticator,** or it specifies **both** and the port(s) are acting in an authenticator role. If **authorised** is specified, the port acts as if it already passed authentication. If **auto** is specified, the port implements normal port authentication control. If **unauthorised** is specified, the port acts as if authentication of the supplicant failed. The default is **auto**.

The **heldperiod** parameter specifies the amount of time in seconds that the Supplicant PAE should refrain from re-contacting an Authenticator PAE after an authentication attempt fails due to an invalid username/password combination. The **heldperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. At the end of the specified time, further authentication attempts are permitted. The valid range of integer values is 0 to 65535. The default is 60.

The **maxreq** parameter specifies the maximum number of times the Authenticator PAE tries to retransmit an EAP request packet to the Supplicant PAE when no response is received. The **maxreq** parameter is used when the **portauth** parameter specifies **8021x,** and the **type** parameter either specifies **authenticator,** or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 10. The default is 2.

The **maxstart** parameter specifies the maximum number of successive EAPOL-Start messages sent before the supplicant assumes no authenticator is present. The **maxstart** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant,** or it specifies **both** and the port(s) are acting in a supplicant role. The valid range of integer values is 1 to 10. The default is 3.

The **mode** parameter specifies whether a port is connected to a single supplicant or to multiple supplicants. The **mode** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter specifies **authenticator**. If **multi** is specified, the port distinguishes between multiple supplicants attached to it and requires each supplicant to authenticate itself separately. If **single** is specified, the port is authenticated by the first supplicant attached to it. The default is **single**.

The **piggyback** parameter specifies whether the piggybacking of network devices onto an authenticated supplicant is allowed. The **piggyback** parameter is used when the **portauth** parameter specifies **8021x**, the **type** parameter specifies **authenticator** or **both,** and the **mode** parameter specifies **single**. If **true** is specified, piggybacking is enabled and packets from any source are allowed to pass through the port once a supplicant has been authorised on it. If **false** is specified, piggybacking is disabled and packets from any source other than the authenticated supplicant are blocked. The default is **true**. On the AR440S, AR441S and AR450S, setting **piggyback** to **false** is only valid for Ethernet interfaces.

The **quietperiod** parameter specifies the amount of time in seconds that the Authenticator PAE should refuse additional authentication attempts after an attempt has already failed due to an invalid username/password combination supplied by the Supplicant PAE. The **quietperiod** parameter is used when the **portauth** parameter specifies **8021x,** and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. In this state, all received EAPOL packets are discarded to prevent denial-of-service attacks. At the end of the specified time, period further authentication attempts are permitted. The valid range of integer values is 0 to 65535. The default is 60.

The **reauthenabled** parameter specifies whether the Authenticator PAE requires the attached supplicants to undergo periodic reauthentication. The **reauthenabled** parameter is used when the **portauth** parameter specifies **8021x,** and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The default is **false**.

The **reauthmax** parameter specifies the maximum number of times the Authenticator PAE tries to establish contact with a Supplicant PAE when no response is received. The **reauthmax** parameter is used when the **portauth** parameter specifies **8021x,** and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. When the maximum number of attempts is reached, an EAPOL-failure message is transmitted and the Authenticator PAE resets itself before trying to contact a Supplicant PAE again. The valid range of integer values is 1 to 10. The default is 2.

The **reauthperiod** parameter specifies the time in seconds between reauthentications of the Supplicant PAE if the **reauthenabled** parameter is set to **true**. The **reauthperiod** parameter is used when the **portauth** parameter specifies **macbased**, or when the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 86400. The default is 3600.

The **servertimeout** parameter specifies the timeout period in seconds that the Authenticator PAE waits for a response from the authentication server after the Authenticator PAE has relayed an EAP response packet to it from the supplicant. The **servertimeout** parameter is used when the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 60. The default is 30.

The **startperiod** parameter specifies the time in seconds between successive attempts by the Supplicant PAE to establish contact with an Authenticator PAE when there is no response. The **startperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. Attempts to establish contact continue until the number of attempts reaches the value set by the **maxstart** parameter. When the value set by the **maxstart** parameter is reached, the Supplicant PAE assumes it is attached to a system that is not EAPOL aware and enters the authenticated state. The valid range of integer values is 1 to 60. The default is 30.

The **supptimeout** parameter specifies the timeout period in seconds to wait for a response from the Supplicant PAE after the Authenticator PAE has relayed an EAP request packet to it from the authentication server. The **supptimeout** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The Authenticator PAE retransmits the packet

to the Supplicant PAE upon timeout, up to the number of times defined by the **maxreq** parameter. The valid range of integer values is 1 to 60. The default is 30.

The **txperiod** parameter specifies the time in seconds between successive attempts by the Authenticator PAE to establish contact with a Supplicant PAE when there is no response. The **txperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 65535. The default is 30.

The **username** parameter specifies the login name to use when the port is acting in a supplicant role. The **username** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If the **username** parameter is present, it overrides the global port authentication username for the specified port(s) only. The **password** parameter must also be specified. Omitting or specifying the **username** and **password** parameters without a specific value causes the global username and password to be used during authentication attempts. The login name is not case sensitive.

The **password** parameter specifies the password to be used when the port acts in a supplicant role during the authentication process. The **password** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If the **password** parameter is present, it overrides the global port authentication password for the specified port(s) only. The **username** parameter must also be specified. Omitting or specifying the **username** and **password** parameters without a specific value causes the global username and password to be used during authentication attempts. A password may contain uppercase letters (A–Z), lowercase letters (a–z), and decimal digits (0–9), and a configurable minimum password length is enforced. The password is case sensitive. If the **method** parameter is **standard**, the password must be at least 6 characters long. If the **method** parameter is **otp**, the password must contain no less than 10 characters and no more than 63 characters. If **otp** is specified, then the password should match the OTP initialisation password used when configuring the user on the authentication server.

The **method** parameter specifies the method used to encrypt the username and password during authentication attempts. The **method** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parametereither specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If the **method** parameter is specified, then the **username** and **password** parameters must also be specified. If **standard** is specified, authentication attempts are conducted by encrypting a standard username and password using EAP-MD5. If **otp** is specified, authentication attempts use one-time passwords via EAP-OTP. The default is **standard**.

The **securevlan** parameter specifies the action that is taken when authenticating any supplicants after the first supplicant has authenticated. If **on** is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. If **off** is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN. The default is **on**.

The **guestvlan** parameter specifes any VLAN that exists on the switch. Prior to the reception of any 802.1x BPDUs on the port, the port is configured in the specified **guestvlan**. Once any EAPOL packets have been received, the port is

removed from the **guestvlan** and placed in the configured access VLAN. This parameter is only valid in Single-Supplicant mode.

The **vlanassignment** parameter specifies whether or not to use the VLAN assignment mechanism on this port. When the **vlanassignment** parameter is set to **enabled**, the device uses any VLAN information returned by the RADIUS server to place the port in a VLAN. When it is set to **disabled**, the device ignores all VLAN information returned by the RADIUS server. The default is **enabled**.

VLAN Assignment cannot be enabled on any port that is one or more of the following:

- tagged
- a mirror port
- in more than one VLAN
- an uplink for a private VLAN
- in a group with other ports for private port vlans
- in a nest vlan
- in a vlan that has a rule type other than port

The **mibreset** parameter provides a mechanism for ageing out stored information about old supplicants. When **enabled** is specified, MIB information for old supplicants is automatically destroyed. This allows new supplicants to be added. When **disabled** is specified, the MIB information remains until it is manually cleared via the **reset portauth port multimib** command. As long the MIB information remains for a supplicant, that supplicant counts towards the supplicant limit. This parameter can only be used in MAC Based Authentication and 802.1x Multi-Supplicant mode. It does not affect supplicants explicitly added through the supplicant MAC command. The default is **enabled**.

The **trap** parameter specifies the events that cause SNMP traps to be sent. When either **success** or **both** is specified, traps are sent when a supplicant is successfully authenticated and when the authentication expires. When either **failure** or **both** is specified, traps are sent when a supplicant fails authentication. When **none** is specified, no traps are sent. The default is **none**.

purge portauth port

Syntax `PURge PORTAuth[={8021x|MACbased}] POrt={ALL|`*`port-name`*`}`

Description This command purges all port authentication configurations for the specified port(s) and port authentication type.

The **portauth** parameter specifies the type of port authentication to purge. If no value is specified, 802.1x is used.

Examples To remove all current 802.1x settings on the router or switch, use the command:

```
pur porta=8021x po=all
```

reset portauth port

Syntax `RESET PORTAuth[={8021x|MACbased}] POrt={ALL|port-name} [SUPPLicantmac={macadd}]`

Description This command reinitialises the port authentication functionality on the specified port(s) for the specified authentication type.

The **portauth** parameter specifies the type of port authentication to reset. If no value is specified, 802.1x is used.

Examples To reset the 802.1x PAE associated with port 1, use the command:

```
reset porta=8021x port=1
```

reset portauth port multimib

Syntax `RESET PORTAuth[={8021x|MACbased}] POrt={ALL|port-name} MULTImib`

Description This command applies when using MAC Based Authentication; or, when using 802.1x authentication, to ports that act as authenticators in a multi-supplicant configuration.

The **portauth** parameter specifies the type of port authentication to reset. If no value is specified, 802.1x is used.

Examples To remove all unused 8021x multi-supplicant MIB entries associated with port 1, use the command:

```
reset porta=8021x po=1 multi
```

To remove all unused multi-supplicant MIB entries associated with eth 0, use the command:

```
reset porta po=eth0 multi
```

set portauth port

Syntax

```
SET PORTAuth[=8021x] POrt={ALL|port-name}
   TYpe=AUthenticator [CONTrol={AUTHorised|AUTO|
   UNauthorised}] [MAXReq=1..10] [MODe={MUlti|SIngle}]
   [PIGgyback={TRue|FAlse}] [QUIETperiod=0..65535]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHMax=1..10]
   [REAUTHPeriod=1..86400] [SERVERTimeout=1..60]
   [SUPPTimeout=1..60] [TXperiod=1..65535]
   [GUEstvlan={VLAN-id|VLAN-name|NONe}][SECurevlan={ON|
   OFF}] [VLANAssignment={ENAbled|
   DIsabled}][MIBReset={ENAbled|DIsabled}]
   [TRap={SUCcess|FAILure|BOTh|NONe}]

SET PORTAuth[=8021x] POrt={ALL|port-name} TYpe=BOth
   [AUTHPeriod=1..60] [CONTrol={AUTHorised|UNauthorised|
   AUTO}] [HELDPeriod=0..65535] [MAXReq=1..10]
   [MAXStart=1..10] [MODe={MUlti|SIngle}]
   [PIGgyback={TRue|FAlse}] [QUIETperiod=0..65535]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHMax=1..10]
   [REAUTHPeriod=1..86400] [SERVERTimeout=1..60]
   [STARTperiod=1..60] [SUPPTimeout=1..60]
   [TXperiod=1..65535] [USERName=login-name
   PASSword=password [METhod={OTP[ENCryption={MD4|MD5}]|
   STandard}]] [GUEstvlan={VLAN-id|VLAN-name|
   NONe}][SECurevlan={ON|OFF}] [VLANAssignment={ENAbled|
   DIsabled}][MIBReset={ENAbled|DIsabled}]
   [TRap={SUCcess|FAILure|BOTh|NONe}]

SET PORTAuth[=8021x8021x]  POrt={ALL|port-name}
   TYpe=SUpplicant [AUTHPeriod=1..60]
   [HELDPeriod=0..65535] [MAXStart=1..10]
   [STARTperiod=1..60] [USERName=login-name
   PASSword=password [METhod={OTP[ENCryption={MD4|MD5}]|
   STandard}]]

SET PORTAuth=MACbased POrt={ALL|port-name}
   [CONTrol={AUTHorised|UNauthorised|AUTO}]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHPeriod=1..86400]
   [QUIETperiod=0..65535] [SECurevlan={ON|OFF}]
   [VLANAssignment={ENAbled|DIsabled}][MIBReset={ENAbled|
   DIsabled}]  [TRap={SUCcess|FAILure|BOTh|NONe}]
```

Description

This command sets the port authentication functionality on ports. Port authentication must already be enabled on the port for the specified **portauth** type, which can be either **8021x** or **macbased**. If **portauth** is **8021x** a port can be set as an authenticator, supplicant, or both.

The **portauth** parameter specifies the type of port authentication to set. If no value is specified, 802.1x is used.

The **type** parameter specifies whether the port is to act as an authenticator, supplicant, or both. **type** can only be specified when **portauth=8021x**. **both** cannot be specified when the **mode** parameter is **multi**.

The **authperiod** parameter specifies the period of time in seconds for which the Supplicant PAE waits for a reply after sending out an EAP-Response frame to

the Authenticator PAE. The **authperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies supplicant, or it specifies **both** and the port(s) are acting in a supplicant role. If no response is received within the specified time, a new authentication attempt may start. The valid range of integer values is 1 to 60. The default is 30.

The **control** parameter specifies the state of the controlled authenticator port. The **mode** parameter is used when the **portauth** parameter specifies **macbased**, or when the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. If **authorised** is specified, the port acts as if it already passed authentication. If **auto** is specified, the port implements normal port authentication control. If **unauthorised** is specified, the port acts as if authentication of the supplicant failed. The default is **auto**.

The **heldperiod** parameter specifies the amount of time in seconds that the Supplicant PAE should refrain from trying to re-contact an Authenticator PAE afteran authentication attempt fails due to an invalid username/password combination. The **heldperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. At the end of the specified time, further authentication attempts are permitted. The valid range of integer values is 0 to 65535. The default is 60.

The **maxreq** parameter specifies the maximum number of times the Authenticator PAE attempts to retransmit an EAP request packet to the Supplicant PAE if no response is received. The **maxreq** parameter is used when the **portauth** parameter specifies **8021x**, and the type parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 10. The default is 2.

The **maxstart** parameter specifies the maximum number of successive EAPOL-Start messages that are sent before the supplicant assumes no authenticator is present. The **maxstart** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. The valid range of integer values is 1 to 10. The default is 3.

The **mode** parameter specifies whether a port is connected to a single supplicant or to multiple supplicants. The **mode** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter specifies **authenticator**. If **multi** is specified, the port distinguishes between multiple supplicants attached to it, and requires each supplicant to authenticate itself separately. If **single** is specified, the port is authenticated by the first supplicant attached to it. The default is **single**.

The **piggyback** parameter specifies whether the piggybacking of network devices onto an authenticated supplicant is allowed. The **piggyback** parameter is used when the **portauth** parameter specifies **8021x**, the **type** parameter specifies **authenticator** or **both**, and the **mode** parameter specifies **single**. If **true** is specified, piggybacking is enabled and packets from any source are allowed to pass through the port once a supplicant has been authorised on it. If **false** is specified, piggybacking is disabled and packets from any source other than the authenticated supplicant are blocked. The default is **true**. On the AR440S, AR441S and AR450S, setting **piggyback** to **false** is only valid for Ethernet interfaces.

The **quietperiod** parameter specifies the amount of time in seconds that the Authenticator PAE should refuse additional authentication attempts after an attempt has already failed due to an invalid username/password combination supplied by the Supplicant PAE. The **quietperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. In this state, all received EAPOL packets are discarded to prevent denial-of-service attacks. At the end of the specified time, further authentication attempts are permitted. The valid range of integer values is 0 to 65535. The default is 60.

The **reauthenabled** parameter specifies whether the Authenticator PAE requires the attached supplicants to undergo periodic reauthentication.The **reauthenabled** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The default is **false**.

The **reauthmax** parameter specifies the maximum number of times the Authenticator PAE tries to establish contact with a Supplicant PAE when no response is received. The **reauthmax** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. When the maximum number of attempts is reached, an EAPOL-failure message is transmitted and the Authenticator PAE resets itself before trying to contact a Supplicant PAE again. The valid range of integer values is 1 to 10. The default is 2.

The **reauthperiod** parameter specifies the time in seconds between reauthentications of the Supplicant PAE if the reAuthEnabled parameter is set to **true**. The **reauthperiod** parameter is used when the **portauth** parameter specifies **macbased**, or when the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 86400. The default is 3600.

The **startperiod** parameter specifies the time in seconds between successive attempts by the Supplicant PAE to establish contact with an Authenticator PAE when there is no response. The **startperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. Attempts to establish contact continue until the number of attempts reaches the value set by the **maxstart** parameter. When the value set by the **maxstart** parameter is reached, the Supplicant PAE assumes it is attached to a system that is not EAPOL aware and enters the authenticated state. The valid range of integer values is 1 to 60. The default is 30.

The **supptimeout** parameter specifies the timeout period in seconds to wait for a response from the Supplicant PAE after the Authenticator PAE has relayed an EAP request packet to it from the authentication server. The **supptimeout** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The Authenticator PAE retransmits the packet to the Supplicant PAE upon timeout, up to the number of times defined by the **maxreq** parameter. The valid range of integer values is 1 to 60. The default is 30.

The **txperiod** parameter specifies the time in seconds between successive attempts by the Authenticator PAE to establish contact with a Supplicant PAE when there is no response. The **txperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 65535. The default is 30.

The **username** parameter specifies the login name to use when the port acts in a supplicant role. The **username** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If the **username** parameter is present, it overrides the global port authentication username for the specified port(s) only. The **password** parameter must also be specified. Omitting or specifying the **username** and **password** parameters without a specific value causes the global username and password to be used during authentication attempts. The login name is not case sensitive.

The **password** parameter specifies the password to be used when the port acts in a supplicant role during the authentication process. The **password** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If the **password** parameter is present, it overrides the global port authentication password for the specified port(s) only. The **username** parameter must also be specified. Omitting or specifying the **username** and **password** parameters without a specific value causes the global username and password to be used during authentication attempts. A password may contain uppercase and lowercase letters and digits (0–9), and a configurable minimum password length is enforced. The password is case sensitive. If the **method** parameter is **standard**, the password must be at least 6 characters long. If the **method** parameter is **otp**, the password must contain no less than 10 characters and no more than 63 characters. Also, if **otp** is specified, then the password should match the OTP initialisation password used when configuring the user on the authentication server.

The **method** parameter specifies the method used to encrypt the username and password during authentication attempts. The **method** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **supplicant**, or it specifies **both** and the port(s) are acting in a supplicant role. If the **method** parameter is specified, then the **username** and **password** parameters must also be specified. If **standard** is specified, authentication attempts are conducted by encrypting a standard username and password using EAP-MD5. If **otp** is specified, authentication attempts use one-time passwords via EAP-OTP. The default is **standard**.

The **securevlan** parameter specifies the action that is taken when authenticating any supplicants after the first supplicant has authenticated. If **on** is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. If **off** is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN. The default is **on**.

The **guestvlan** parameter specifes any VLAN that exists on the switch. Prior to the reception of any 802.1x BPDUs on the port, the port is configured in the specified **guestvlan**. Once any EAPOL packets have been received, the port is removed from the **guestvlan** and placed in the configured access VLAN. This parameter is only valid in Single-Supplicant mode.

The **vlanassignment** parameter specifies whether or not to use the VLAN assignment mechanism on this port. When the **vlanassignment** parameter is set to **enabled**, the device uses any VLAN information returned by the RADIUS server to place the port in a VLAN. When it is set to **disabled**, the device ignores all VLAN information returned by the RADIUS server. The default is **enabled**.

VLAN Assignment cannot be enabled on any port that is one or more of the following:

- tagged
- a mirror port
- in more than one VLAN
- an uplink for a private VLAN
- in a group with other ports for private port vlans
- in a nest vlan
- in a vlan that has a rule type other than port

The **mibreset** parameter provides a mechanism for ageing out stored information about old supplicants. When **enabled** is specified, MIB information for old supplicants is automatically destroyed. This allows new supplicants to be added. When **disabled** is specified, the MIB information remains until it is manually cleared via the **reset portauth port multimib** command. As long the MIB information remains for a supplicant, that supplicant counts towards the supplicant limit. This parameter can only be used in MAC Based Authentication and 802.1x Multi-Supplicant mode. It does not affect supplicants explicitly added through the supplicant MAC command. The default is **enabled**.

The **trap** parameter specifies the events that cause SNMP traps to be sent. When either **success** or **both** is specified, traps are sent when a supplicant is successfully authenticated, and when the authentication expires. When either **failure** or **both** is specified, traps are sent when a supplicant fails authentication. When **none** is specified, no traps are sent. The default is **none**.

Examples To modify the current **quietperiod** setting for 802.1x on port 1, use the command:

```
set porta=8021x ty=au po=1 quiet=1024
```

set portauth port supplicantmac

Syntax

```
SET PORTAuth[=8021x] POrt={ALL|port-name}
   SUPPLicantmac=macadd [CONTrol={AUTHorised|UNauthorised|
   AUTO}] [MAXReq=1..10] [QUIETperiod=0..65535]
   [REAUTHENAbled={TRue|FAlse}] [REAUTHMax=1..10]
   [REAUTHPeriod=1..86400] [SERVERTimeout=1..60]
   [SUPPTimeout=1..60] [TXperiod=1..65535]
   [SECurevlan={ON|OFF}] [VLANAssignment={ENAbled|
   DIsabled}][TRap={SUCcess|FAIlure|BOTh|NONe}]]] |
   [DEFAULT]}

SET PORTAUTH=MACbased PORT={ALL | portname}
   SUPPLicantmac=macadd {[[CONTrol={AUTHorised|
   UNauthorised|AUTO}] [REAUTHENAbled={TRue | FAlse}]
   [REAUTHPeriod=1..86400] [QUIETperiod=0..65535]
   [SECurevlan={ON|OFF}] [VLANAssignment={ENAbled|
   DIsabled}][TRap={SUCcess|FAIlure|BOTh|NONe}]]] |
   [DEFAULT]}
```

Description

This command allows the Authenticator PAE configuration for specified port(s) to be overridden for a specific supplicant. Port authentication must already be enabled on the specified port(s) for the specified **portauth** type, which can be either **8021x** or **macbased**. Any supplicant that attaches to an Authenticator PAE configured for multi-supplicant support uses the parameters set by either the **set portauth port type=authenticator** command or **set portauth port type=both** command, unless the MAC address of the supplicant matches the **supplicantmac** parameter of a previously entered **set portauth port supplicantmac** command. At least one non-standard parameter value must be entered in the command.

The **portauth** parameter specifies the type of port authentication to set. If no value is specified, **8021x** is used.

The **control** parameter specifies the state of the controlled authenticator port. The **mode** parameter is used when the **portauth** parameter specifies **macbased**, or when the **type** parameter either specifies authenticator, or it specifies **both** and the port(s) are acting in an authenticator role. If **authorised** is specified, the port acts as if it has already passed authentication. If **auto** is specified, the port implements normal port authentication control. If **unauthorised** is specified, the port acts as if authentication of the supplicant failed. The default is **auto**.

The **maxreq** parameter specifies the maximum number of times the Authenticator PAE attempts to retransmit an EAP request packet to the Supplicant PAE if no response is received. The **maxreq** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 10. The default is 2.

The **quietperiod** parameter specifies the amount of time in seconds that the Authenticator PAE should refuse additional authentication attempts, if an attempt has already failed due to an invalid username/password combination supplied by the Supplicant PAE. The **quietperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. In this state, all received EAPOL packets are discarded to prevent denial-of-service attacks. At the end of the specified time, further

authentication attempts are permitted. The valid range of integer values is 0 to 65535. The default is 60.

The **reauthenabled** parameter specifies whether the Authenticator PAE requires the Supplicant PAE to undergo periodic reauthentication. The **reauthenabled** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The default is **false**.

The **reauthperiod** parameter specifies the time between reauthentications of the Supplicant PAE if the **reauthenabled** parameter is set to **true**. The **reauthperiod** parameter is used when the **portauth** parameter specifies **macbased**, or when the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 86400. The default is 3600.

The **supptimeout** parameter specifies the timeout period in seconds to wait for a response from the Supplicant PAE after the Authenticator PAE has relayed an EAP request packet to it from the authentication server. The **supptimeout** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The Authenticator PAE retransmits the packet to the Supplicant PAE on timeout, up to the number of times defined by the **maxreq** parameter. The valid range of integer values is 1 to 60. The default is 30.

The **txperiod** parameter specifies the time in seconds between successive attempts by the Authenticator PAE to establish contact with a Supplicant PAE when there is no response. The **txperiod** parameter is used when the **portauth** parameter specifies **8021x**, and the **type** parameter either specifies **authenticator**, or it specifies **both** and the port(s) are acting in an authenticator role. The valid range of integer values is 1 to 65535. The default is 30.

The **securevlan** parameter specifies the action that is taken when authenticating any supplicants after the first supplicant has authenticated. If **on** is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. If **off** is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN. The default is **on**.

The **vlanassignment** parameter specifies whether or not to use the VLAN assignment mechanism on this port. When the **vlanassignment** parameter is set to **enabled**, the device uses any VLAN information returned by the RADIUS server to place the port in a VLAN. When it is set to **disabled**, the device ignores all VLAN information returned by the RADIUS server. The default is **enabled**.

VLAN Assignment cannot be enabled on any port that is one or more of the following:

- tagged
- a mirror port
- in more than one VLAN
- an uplink for a private VLAN
- in a group with other ports for private port vlans
- in a nest vlan
- in a vlan that has a rule type other than port

The **trap** parameter specifies the events that cause SNMP traps to be sent. When either **success** or **both** is specified, traps are sent when a supplicant is successfully authenticated, and when the authentication expires. When either **failure** or **both** is specified, traps are sent when a supplicant fails authentication. When **none** is specified, no traps are sent. The default is **none**.

Examples To modify the current **quietperiod** setting for 802.1x on Port 1 of a multi-supplicant system with a supplicant MAC address of `22-22-22-22-22-22`, use the command:

```
set porta=8021x po=1 suppl=22-22-22-22-22-22 quiet=1024
```

To modify the current **quietperiod** setting on eth 0 of a multi-supplicant system with a supplicant MAC address of `22-22-22-22-22-22`, use the command:

```
set porta po=eth0 suppl=22-22-22-22-22-22 quiet=1024
```

set portauth username

Syntax `SET PORTAuth[=8021x] USERName=`*`login-name`* `PASSword=`*`password`*
`[METhod={OTP[ENCryption={MD4|MD5}]|STandard}]`

Description The **portauth** parameter specifies the type of port authentication to which the settings apply. This command only applies to 802.1x, and defaults to 802.1x if no value is specified.

Examples To set the global port authentication username and password for 802.1x on the router or switch, use the command:

```
set porta usern=manager pass=friend
```

show portauth

Syntax SHow PORTAuth**[={8021x|MACbased}]**

Description The **portauth** parameter specifies the type of port authentication that is displayed. If no value is specified, 802.1x is used.

Figure 8-1: Example output from the **show portauth=8021x** command

```
802.1x System
-----------------------------------------------------
  SystemAuthControl..... ENABLED
  Global Username..... portAuthPortAuth
  Global Password..... portAuthPortAuth
  Global Encryption Method..... OTP
  Global Encryption Type..... MD5
  Number of Multi Supplicants.. 4 (limit 480)


  Port      PAE Capabilities      Protocol Version
-----------------------------------------------------
  port1     Supplicant            1
  port2     Authenticator         1
  port3     Both                  1
  port4     None                  1
  port5     None                  1
  port6     None                  1
  port7     None                  1
  port8     None                  1
```

Table 8-1: Parameters in the output of the **show portauth=8021x** command

Parameter	Meaning
SystemAuthControl	Whether 802.1x port authentication is Enabled or Disabled.
Number of Supplicants	The number of 802.1x supplicants currently being handled by the system. The limit value in brackets indicates the maximum number of multi supplicants that the system can handle.

Figure 8-2: Example output from the **show portauth=macbased** command

```
MAC Based Authentication System
-------------------------------------------------
  SystemAuthControl............. ENABLED
  Number of Supplicants.........4 (limit 480)


  Port                        PAE Status
-------------------------------------------------
  port1                       Enabled
  port2                       Enabled
  port3                       Enabled
  port4                       None
  port5                       None
  port6                       None
  port7                       None
  port8                       None
```

Table 8-2: Parameters in the output of the **show portauth=macbased** command

Parameter	Meaning
SystemAuthControl	Whether MAC Based Authentication is Enabled or Disabled.
PAE Status	Whether MAC Based Authentication is Enabled or Disabled for a particular port.
Number of Supplicant	The number of MAC Based Authentication supplicants currently being handled by the system. The limit value in brackets indicates the maximum number of multi supplicants that the system can handle.

Examples To show the 802.1x capabilities of all ports on the device, use the command:

```
sh porta=802x
```

show portauth counter

Syntax `SHow PORTAuth[=8021x] COUnter POrt={ALL|`*`port-name`*`}`

Description The **portauth** parameter specifies the type of port authentication for which counters are displayed. This command only applies to 802.1x, and defaults to 802.1x if no value is specified.

show portauth port

Syntax SHow PORTAuth**[={8021x|MACbased}]** POrt={ALL|*port-name*}

Description The **portauth** parameter specifies the type of port authentication for which the port's configuration is displayed. If no value is specified, 802.1x is used.

Figure 8-3: Example output from the **show portauth port=8021x** command

```
802.1x Configuration
-----------------------------------------------------------
Interface: port1
  PAE Type............... Supplicant
          heldPeriod..................... 60
          authPeriod..................... 30
          startPeriod.................... 30
          maxStart....................... 3
          Supplicant PAE State.......... AUTHENTICATED

Interface: port2
  PAE Type............... Authenticator
          Authenticator PAE State....... CONNECTING
          Port Status.................... unauthorised
          Backend Authenticator State... IDLE
          AuthControlPortControl........ Auto
          quietPeriod.................... 60
          txPeriod....................... 30
          suppTimeout.................... 30
          serverTimeout.................. 30
          maxReq......................... 2
          reAuthMax...................... 2
          reAuthPeriod................... 3600
          reAuthEnabled.................. False
          piggyBack...................... True
          keyTransmissionEnabled........ False (not supported)
          adminControlledDirections..... Both (not supported)
          guestVlan...................... VLAN2
          trap........................... None
          vlanAssignment................. Enabled

Interface: port3
  PAE Type............... Both

  Multi-Supplicant Authenticator
  Number of Multi-Supplicants........... 2
    Default Settings
          AuthControlPortControl........ Auto
          quietPeriod.................... 60
          txPeriod....................... 30
          suppTimeout.................... 30
          serverTimeout.................. 30
          maxReq......................... 2
          reAuthMax...................... 2
          reAuthPeriod................... 3600
          reAuthEnabled.................. False
          secureVlan..................... On
          trap .......................... None
          mibReset ...................... Enabled
          vlanAssignment ................ Enabled
```

Figure 8-3: Example output from the **show portauth port=8021x** command (Continued)

```
Attached Supplicant(s)
      MAC Address..................... 12-34-56-78-90-12
          Authenticator PAE State....... INITIALISE
          Port Status................... authorised
          Backend Authenticator State... IDLE
          AuthControlPortControl........ Auto
          quietPeriod................... 60
          txPeriod...................... 30
          suppTimeout................... 30
          serverTimeout................. 30
          maxReq........................ 2
          reAuthMax..................... 2
          reAuthPeriod.................. 600
          reAuthEnabled................. False
          keyTransmissionEnabled........ False (not supported)
          operControlledDirections...... False (not supported)
          secureVlan.................... On
          trap ......................... None
          mibReset ..................... Enabled
          vlanAssignment ............... Enabled

Attached Supplicant(s)
      MAC Address..................... ff-ee-dd-cc-bb-aa
          Authenticator PAE State....... INITIALISE
          Port Status................... authorised
          Backend Authenticator State... IDLE
          AuthControlPortControl........ Auto
          quietPeriod................... 60
          txPeriod...................... 30
          suppTimeout................... 30
          serverTimeout................. 30
          maxReq........................ 2
          reAuthMax..................... 2
          reAuthPeriod.................. 500
          reAuthEnabled................. False
          keyTransmissionEnabled........ False (not supported)
          operControlledDirections...... False (not supported)
          secureVlan.................... On
          trap ......................... None
          mibReset ..................... Enabled
          vlanAssignment ............... Enabled

  Supplicant
          heldPeriod.................... 60
          authPeriod.................... 30
          startPeriod................... 30
          maxStart...................... 3
          Supplicant PAE State.......... DISCONNECTED
```

Table 8-3: Parameters in the output of the **show portauth=8021x port** command

Parameter	Meaning
Authenticator PAE State	The current state of the Authenticator PAE: Initialise Disconnected Connecting Authenticating Authenticated Aborting Held Force authenticated Force unauthorised
guestVlan	Displays the Guest VLAN-ID if one has been specified. NONE is displayed otherwise.
secureVlan	The action taken when authenticating any supplicants after the first supplicant has authenticated. If On is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. If Off is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN.
trap	Displays the types of events that cause SNMP traps to be sent. If Success or Both is specified, traps are sent when a supplicant is successfully authenticated and when the authentication expires. If Failure or Both is specified, traps are sent when a supplicant fails authentication. If None is specified, no traps are sent.
mibReset	Whether MIB information stored for supplicants will age out.
vlanAssignment	Whether the VLAN assignment mechanism is enabled or disabled.
Number of Supplicants	The number of 802.1x supplicants currently being handled by the port.

Figure 8-4: Example output from the **show portauth=macbased port** command

```
MAC Based Authentication Configuration
--------------------------------------------------------------------------------
Interface: port1
    PAE Status..................... Enabled
    Number of Supplicants ........ 2
    Default Settings
            AuthControlPortControl........ Auto
            quietPeriod .................. 60
            reAuthPeriod.................. 3600
            reAuthEnabled................. False
            secureVlan.................... On
            trap ......................... None
            mibReset ..................... Enabled
            vlanAssignment ............... Enabled

    Attached Supplicant(s)
        MAC Address..................... 12-34-56-78-90-12
            Authenticator PAE State....... AUTHENTICATED
            Port Status................... authorised
            Backend Authenticator State... IDLE
            AuthControlPortControl........ Auto
            quietPeriod................... 60
            reAuthPeriod.................. 3600
            reAuthEnabled................. False
            secureVlan.................... On
            trap ......................... None
            mibReset ..................... Enabled
            vlanAssignment ............... Enabled

        MAC Address..................... ff-ee-dd-cc-bb-aa
            Authenticator PAE State....... AUTHENTICATED
            Port Status................... authorised
            Backend Authenticator State... IDLE
            AuthControlPortControl........ Auto
            quietPeriod................... 60
            reAuthPeriod.................. 600
            reAuthEnabled................. False
            secureVlan.................... On
            trap ......................... None
            mibReset ..................... Enabled
            vlanAssignment ............... Enabled
```

Table 8-4: Parameters displayed in the output of the **show portauth=macbased port** command.

Parameter	Meaning
PAE Status	Whether MAC Based Authentication is Enabled or Disabled on the port.
Number of Supplicants	The number of MAC Based Authentication supplicants currently being handled by the port.
AuthControlPortControl	Whether the port's authorisation status is being administratively controlled: Force Authorised Auto Force Unauthorised.
quietPeriod	The delay period in seconds before the Authenticator PAE tries to acquire a Supplicant PAE.
reAuthPeriod	The time in seconds between reauthentication of the supplicant.
reAuthEnabled	Whether regular reauthentication takes place on this port.
Mac Address	An Ethernet six-octet MAC address, specifying the hardware address of the connected supplicant.
Authenticator PAE State	The current state of the Authenticator PAE: Initialise Disconnected Authenticating Authenticated Aborting Held ForceAuth ForceUnauth
Port Status	The current state of the controlled port: Authorised Unauthorised.
Backend Authentication State	The current state of the Backend Authentication: Request Success Fail Timeout Idle Initialise
secureVlan	Determines the action taken when authenticating any supplicants, after the first supplicant has authenticated. When On is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. When Off is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN.

Table 8-4: Parameters displayed in the output of the **show portauth=macbased port** command.

Parameter	Meaning
trap	Displays the types of events that will cause SNMP traps to be sent. When Success or Both is specified, traps are sent when a supplicant is successfully authenticated and when the authentication expires. When Failure or Both is specified, traps are sent when a supplicant fails authentication. When None is specified, no traps are sent.
mibReset	Determines whether MIB information stored for supplicants will age out.
vlanAssignment	Whether the VLAN assignment mechanism is enabled or disabled
secureVlan	Determines the action taken when authenticating any supplicants after the first supplicant has authenticated. When On is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. When Off is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN.
Number of Supplicants	Specifies the number of MAC Based Authentication supplicants currently being handled by the port.

Examples To show the current 802.1x configuration settings for port 1, use the command:

```
sh porta=8021x po=1
```

show portauth port multisupplicant

Syntax SHow PORTAuth**[={8021x|MACbased}]** MULTIsupplicant POrt={ALL|*port-name*} **[SUPPlicantmac=macadd]**

where:

- macadd is an Ethernet six-octet MAC address, expressed as six pairs of hexadecimal digits delimited by hyphens.

Description The **portauth** parameter specifies the type of port authentication that will have its port's configuration displayed. If no value is specified, 802.1x is used.

The **supplicantmac** parameter specifies the hardware address of the connected supplicant to display, in standard MAC format, for example 11-22-33-44-55-66.

Figure 8-5: Example output from the **show portauth=8021x port multisupplicant** command

```
802.1x Multi-Supplicant Configuration
---------------------------------------------------------------
Interface: port1
  PAE Type..............Authenticator
  Multi-Supplicant Authenticator
    Number of Multi Supplicants......2
    Default Settings
       AuthControlPortControl........forceAuthorised
       quietPeriod...................60
       txPeriod......................30
       suppTimeout...................30
       serverTimeout.................30
       maxReq........................2
       reAuthMax.....................2
       reAuthPeriod..................120
       reAuthEnabled.................True
       secureVlan..................... On
       trap ........................ None
       mibReset ..................... Enabled
       vlanAssignment ............... Enabled

    Attached Supplicant(s)
      MAC Address...................ba-09-87-65-43-21
       Authenticator PAE State.......INITIALISE
       Port Status...................authorised
       Backend Authenticator State...INITIALISE
       AuthControlPortControl........forceAuthorised
       quietPeriod...................60
       txPeriod......................30
       suppTimeout...................30
       serverTimeout.................30
       maxReq........................2
       reAuthMax.....................2
       reAuthPeriod..................120
       reAuthEnabled.................True
       keyTransmissionEnabled........False (not supported)
       operControlledDirections......False (not supported)
       secureVlan..................... On
       trap ........................ None
       mibReset ..................... Enabled
       vlanAssignment ............... Enabled
```

Figure 8-5: Example output from the **show portauth=8021x port multisupplicant** command (Continued)

```
Attached Supplicant(s)
  MAC Address.....................12-34-56-78-90-ab
   Authenticator PAE State.......INITIALISE
   Port Status...................authorised
   Backend Authenticator State...INITIALISE
   AuthControlPortControl........forceAuthorised
   quietPeriod...................60
   txPeriod......................30
   suppTimeout...................30
   serverTimeout.................30
   maxReq........................2
   reAuthMax.....................3
   reAuthPeriod..................60
   reAuthEnabled.................True
   keyTransmissionEnabled........False (not supported)
   operControlledDirections......False (not supported
   secureVlan..................... On
   trap ........................... None
   mibReset ...................... Enabled
   vlanAssignment ................ Enabled
```

Table 8-5: Parameters in the output of the **show portauth=8021x port multisupplicant** command

Parameter	Meaning
secureVlan	The action taken when authenticating any supplicants after the first supplicant has authenticated. When On is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. When Off is specified, subsequent supplicants are allowed on the port, as long as they authenticated to a valid VLAN.
trap	Specifies which type of events will cause SNMP traps to be sent. When Success or Both is specified, traps are sent when a supplicant is successfully authenticated and when the authentication expires. When Failure or Both, traps are sent when a supplicant fails authentication. When set to None, no traps are sent.
mibReset	Whether MIB information stored for supplicants will age out.
vlanAssignment	Specifies whether to use the VLAN assignment mechanism on this port; either Enabled or Disabled
Number of Supplicants	Specifies the number of 802.1x supplicants currently being handled by the port.

Figure 8-6: Example output from the **show portauth=macbased port multisupplicant** command

```
MAC Based Authentication Configuration
--------------------------------------------------------------------------------
Interface: port1
    PAE Status..................... Enabled
    Number of Supplicants ........ 1
    Default Settings
            AuthControlPortControl........ Auto
            quietPeriod .................. 60
            reAuthPeriod.................. 3600
            reAuthEnabled................. False
            secureVlan.................... On
            trap ......................... None
            mibReset ..................... Enabled
            vlanAssignment ............... Enabled

    Attached Supplicant(s)
        MAC Address..................... 12-34-56-78-90-12
            Authenticator PAE State....... AUTHENTICATED
            Port Status................... authorised
            Backend Authenticator State... IDLE
            AuthControlPortControl........ Auto
            quietPeriod................... 60
            reAuthPeriod.................. 3600
            reAuthEnabled................. False
            secureVlan.................... On
            trap ......................... None
            mibReset ..................... Enabled
            vlanAssignment ............... Enabled
```

Table 8-6: Parameters in the output of the **show portauth=macbased port multisupplicant** command

Parameter	Meaning
PAE Status	Whether the port has been enabled for MAC Based Authentication; either Enabled or Disabled.
Number of Supplicants	The number of MAC Based Authentication supplicants currently being handled by the port.
AuthControlPortControl	Whether the port's authorisation status is being administratively controlled; either Force Authorised, Auto, or Force Unauthorised.
quietPeriod	The delay period in seconds before the Authenticator PAE tries to acquire a Supplicant PAE.
reAuthPeriod	The time in seconds between reauthentication of the supplicant.
reAuthEnabled	Whether regular reauthentication takes place on this port.
Mac Address	An Ethernet six-octet MAC address, specifying the hardware address of the connected supplicant.

Table 8-6: Parameters in the output of the **show portauth=macbased port multisupplicant** command (Continued)

Parameter	Meaning
Authenticator PAE State	The current state of the Authenticator PAE: Initialise Disconnected Authenticating Authenticated Aborting Held ForceAuth ForceUnauth
Port Status	Whether the current state of the controlled port is Authorised or Unauthorised.
Backend Authentication State	The current state of the Backend Authentication: Request Success Fail Timeout Idle Initialize
secureVlan	Determines the action taken when authenticating any supplicants after the first supplicant has authenticated. When On is specified, subsequent supplicants must authenticate to the same VLAN as that of the first authenticated host. When Offis specified, subsequent supplicants are allowed on the port, as long as they have authenticated to a valid VLAN.
trap	Specifies which type of events will cause SNMP traps to be sent. When Success or Both is specified, traps are sent when a supplicant is successfully authenticated and when the authentication expires. When Failure or Both is specified, traps are sent when a supplicant fails authentication. When set to None, no traps are sent.
mibReset	Determines whether MIB information stored for supplicants will age out.
vlanAssignment	Specifies whether to use the VLAN assignment mechanism on this port; either Enabled or Disabled
Number of Supplicants	Specifies the number of MAC Based Authenication supplicants currently being handled by the port.

show portauth timer

Syntax SHow PORTAuth**[={8021x|MACbased}]** TIMer POrt={ALL|*port-name*}

Description The **portauth** parameter specifies the type of port authentication for which port timer information is displayed. If no value is specified, 802.1x is used.

Figure 8-7: Example output from the **show portauth=macbased timer** command

```
MAC Based Authentication Timers
-------------------------------------------------------------------------------
Interface: port3

Supplicant             quietWhile        reAuthWhen
-------------------------------------------------
12-34-56-78-90-12      00000             00000
ff-ee-dd-cc-bb-aa      00000             00000

Interface: port4

Supplicant             quietWhile        reAuthWhen
-------------------------------------------------
12-34-56-12-34-56      00000             00000
aa-bb-cc-dd-ee-ff      00000             00000
```

Table 8-7: New parameters in the output of the **show portauth=macbased timer** command

Parameter	Meaning
PAE Status	Whether the port has been enabled for MAC Based Authentication; either Enabled or Disabled.
Attached Supplicant	An Ethernet six-octet MAC address, specifying the hardware address of the connected supplicant.
quietWhile	A timer used by the Authenticator state machine to define periods of time during which it does not try to authenticate a supplicant. The initial value of this timer is quietPeriod.
reAuthWhen	A timer used by the Reauthentication Timer state machine to determine when reauthentication of the supplicant takes place. The initial value of this timer is ReAuthPeriod.

Examples To show the 802.1x timers for port 1, use the command:

```
sh porta=8021x tim po=1
```

show switch port

Syntax SHow SWItch POrt[={*port-list*|ALL}]

Figure 8-8: Example output from the **show switch port** command

```
Switch Port Information
------------------------------------------------------------
 Port ............................ 1
   Description ................... To intranet hub, port 4
   Status ........................ ENABLED
   Link State .................... Up
   UpTime ........................ 00:10:49
   Configured speed/duplex ....... Autonegotiate
   Actual speed/duplex ........... 100 Mbps, full duplex
   Automatic MDI/MDI-X ........... Enabled
   Configured MDI/MDI-X........... MDI-X
   Actual MDI/MDI-X............... MDI
   Broadcast rate limit .......... 128Kbps
   Multicast rate limit .......... -
   DLF rate limit ................ -
   Flow control(s) ............... Disabled
   Send tagged pkts for VLAN(s) .. vlan2 (2)
                                   vlan3 (3)
   Port-based VLAN ............... accounting (4)
   Dynamically assigned VLAN...... vlan3 (3)
   Ingress Filtering ............. OFF
------------------------------------------------------------
```

Table 8-8: New parameters in the output of the **show switch port** command

Parameter	Meaning
Dynamically assigned VLAN	The name of the VLAN to which the port is assigned if it has been dynamically assigned by VLAN.

show vlan

Syntax SHow VLAN[={*vlan-name*|1..4094|ALL}]

Table 8-9: New parameters in the output of the **show vlan** command

Parameter	Meaning
Configured	Specifies which ports are configured for the specified VLAN if the VLAN has ports that are either assigned to another VLAN, or configured for another VLAN but assigned to this VLAN by Dynamic VLAN Assignment.
Actual	Specifies which ports are are actually in the specified VLAN if the VLAN has ports that are either assigned to another VLAN, or configured for another VLAN but assigned to this VLAN by Dynamic VLAN Assignment.