

SROS Command Line Interface

Reference Guide

Software Version J.03.01

© Copyright 2005 Hewlett-Packard Development Company, L.P.
The information contained herein is subject to change without notice.

This document contains proprietary information, which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without the prior written consent of Hewlett-Packard.

Publication Number

5991-2114
August 2005

Applicable Products

ProCurve Secure Router 7102dl	(J8752A)
ProCurve Secure Router 7203dl	(J8753A)

Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation. CompactFlash is a U.S. registered trademark of the CompactFlash Association. AOL Instant Messenger (AIM) is a U.S. registered trademark of American Online, Inc. Quake is a U.S. registered trademark of id Software, Inc. ICQ is a U.S. registered trademark of ICQ, Inc. pcAnywhere is a U.S. trademark of Symantec Corporation.

Disclaimer

HEWLETT-PACKARD COMPANY MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

Warranty

See the Customer Support/Warranty booklet included with the product.

A copy of the specific warranty terms applicable to your Hewlett-Packard products and replacement parts can be obtained from your HP Sales and Service Office or authorized dealer.

Table of Contents

Basic Mode Command Set	10
Enable Mode Command Set	20
Global Configuration Mode Command Set	236
DHCP Pool Command Set	394
HDLC Command Set	411
IKE Policy Command Set	462
IKE Policy Attributes Command Set	473
IKE Client Command Set	479
Crypto Map IKE Command Set	483
Crypto Map Manual Command Set	492
Radius Group Command Set	503
CA Profile Configuration Command Set	505
Certificate Configuration Command Set	516
Ethernet Interface Configuration Command Set	520
Serial Interface Configuration Command Set	570
T1 Interface Configuration Command Set	579
DSX-1 Interface Configuration Command Set	594
E1 Interface Configuration Command Set	604
G.703 Interface Configuration Command set	619
Modem Interface Configuration Command Set	626
BRI Interface Configuration Command set	630
Frame Relay Interface Config Command Set	643
Frame Relay Sub-Interface Config Command Set	663
ATM Interface Config Command Set	722
ATM Sub-Interface Config Command Set	725
ADSL Interface Config Command Set	776
BGP Configuration Command Set	780
BGP Neighbor Configuration Command Set	786
Tunnel Configuration Command Set	800
PPP Interface Configuration Command Set	851
Loopback Interface Configuration Command Set	916
Line (Console) Interface Config Command Set	943
Line (Telnet) Interface Config Command Set	954
Router (RIP) Configuration Command Set	961
Router (OSPF) Configuration Command Set	970
Quality of Service (QoS) Map Commands	984
AS Path List Command Set	992
Common Commands	995
Index	1009

REFERENCE GUIDE INTRODUCTION

This manual provides information about the commands that are available with all of the ProCurve Secure routers.

If you are new to the Operating System's Command Line Interface (CLI), take a few moments to review the information provided in the section which follows (*CLI Introduction*).

If you are already familiar with the CLI and you need information on a specific command or group of commands, proceed to *Command Descriptions* on [page 9](#) of this guide.

CLI INTRODUCTION

This portion of the Command Reference Guide is designed to introduce you to the basic concepts and strategies associated with using the Operating System's Command Line Interface (CLI).

Accessing the CLI from your PC

All products using the are initially accessed by connecting a VT100 terminal (or terminal emulator) to the **CONSOLE** port located on the rear panel of the unit using a standard DB-9 (male) to DB-9 (female) serial cable. Configure the VT100 terminal or terminal emulation software to the following settings:

- 9600 baud
- 8 data bits
- No parity
- 1 stop bit
- No flow control

Note	<i>For more details on connecting to your unit, refer to the Quick Configuration Guides and Quick Start Guides located on the Secure Router OS Documentation CD provided with your unit.</i>

Understanding Command Security Levels

The has two command security levels — **Basic** and **Enable**. Both levels support a specific set of commands. For example, all interface configuration commands are accessible only through the Enable security level. The following table contains a brief description of each level.

Level	Access by...	Prompt	With this level you can...
Basic	beginning an SROS session.	ProCurve>	• display system information • perform traceroute and ping functions • open a Telnet session

Level	Access by...	Prompt	With this level you can...
Enable	entering enable while in the Basic command security level as follows: > enable	ProCurve#	- manage the startup and running configurations - use the debug commands - enter any of the configuration modes

Note *To prevent unauthorized users from accessing the configuration functions of your product, immediately install an Enable-level password. Refer to the Quick Configuration Guides and Quick Start Guides located on the **Secure Router OS Documentation** CD provided with your unit for more information on configuring a password.*

Understanding Configuration Modes

The Secure Router OS has four configuration modes to organize the configuration commands – Global, Line, Router, and Interface. Each configuration mode supports a set of commands specific to the configurable parameters for the mode. For example, all Frame Relay configuration commands are accessible only through the Interface Configuration mode (for the virtual Frame Relay interface). The following table contains a brief description of each level.

Mode	Access by...	Sample Prompt	With this mode you can...
Global	entering config while at the Enable command security level prompt. For example: >enable # config term	ProCurve(config) #	- set the system's Enable-level password(s) - configure the system global IP parameters - configure the SNMP parameters - enter any of the other configuration modes
Line	specifying a line (console or Telnet) while at the Global Configuration mode prompt. For example: >enable #config term (config) # line console 0	ProCurve(config-con0) #	- configure the console terminal settings (databate, login password, etc.) - create Telnet logins and specify their parameters (login password, etc.)

Mode	Access by...	Sample Prompt	With this mode you can...
Router	entering router rip or router ospf while at the Global Configuration mode prompt. For example: >enable #config term (config)# router rip	ProCurve(config-rip)#	- configure RIP or OSPF parameters - suppress route updates - redistribute information from outside routing sources (protocols)
Interface	specifying an interface (T1, Ethernet, Frame Relay, ppp, etc.) while in the Global Configuration mode. For example: >enable #config term (config)# int eth 0/1	ProCurve(config-eth 0/1)# (The above prompt is for the Ethernet LAN interface located on the rear panel of the unit.)	configure parameters for the available LAN and WAN interfaces

Using CLI Shortcuts

The provides several shortcuts which help you configure your Secure Router OS product more easily. See the following table for descriptions.

Shortcut	Description
Up arrow key	To re-display a previously entered command, use the up arrow key. Continuing to press the up arrow key cycles through all commands entered starting with the most recent command.
Tab key	Pressing the <Tab> key after entering a partial (but unique) command will complete the command, display it on the command prompt line, and wait for further input.
?	The CLI contains help to guide you through the configuration process. Using the question mark, do any of the following: - Display a list of all subcommands in the current mode. For example: ProCurve(config-t1 1/1)#coding ? ami - Alternate Mark Inversion b8zs - Bipolar Eight Zero Substitution - Display a list of available commands beginning with certain letter(s). For example: ProCurve(config)#ip d? default-gateway dhcp-server domain-lookup domain-name domain-proxy - Obtain syntax help for a specific command by entering the command, a space, and then a question mark (?). The CLI displays the range of values and a brief description of the next parameter expected for that particular command. For example: ProCurve(config-eth 0/1)#mtu ? <64-1500> - MTU (bytes)

Shortcut	Description
<Ctrl> + A	Jump to the beginning of the displayed command line. This shortcut is helpful when using the no form of commands (when available). For example, pressing <Ctrl + A> at the following prompt will place the cursor directly after the #: ProCurve(config-eth 0/1) # ip address 192.33.55.6
<Ctrl> + E	Jump to the end of the displayed command line. For example, pressing <Ctrl + E> at the following prompt will place the cursor directly after the 6: ProCurve(config-eth 0/1) # ip address 192.33.55.6
<Ctrl> + U	Clears the current displayed command line. The following provides an example of the <Ctrl + U> feature: ProCurve(config-eth 0/1) # ip address 192.33.55.6 (Press <Ctrl + U> here) ProCurve(config-eth 0/1) #
<i>auto finish</i>	You need only enter enough letters to identify a command as unique. For example, entering int t1 1/1 at the Global configuration prompt provides you access to the configuration parameters for the specified T1 interface. Entering interface t1 1/1 would work as well, but is not necessary.

Performing Common CLI Functions

The following table contains descriptions of common CLI commands.

Command	Description
do	The do command provides a way to execute commands in other command sets without taking the time to exit the current and enter the desired one. The following example shows the do command used to view the Frame Relay interface configuration while currently in the T1 interface command set: ProCurve(config) # interface t1 1/1 ProCurve(config-t1 1/1) # do show interfaces fr 7
no	To undo an issued command or to disable a feature, enter no before the command. For example: no shutdown t1 1/1
copy running-config startup-config	When you are ready to save the changes made to the configuration, enter this command. This copies your changes to the unit's nonvolatile random access memory (NVRAM). Once the save is complete, the changes are retained even if the unit is shut down or suffers a power outage.
show running config	Displays the current configuration.

Command	Description
debug	Use the debug command to troubleshoot problems you may be experiencing on your network. These commands provide additional information to help you better interpret possible problems. For information on specific debug commands, refer to the section <i>Enable Mode Command Set</i> on page 20.
undebug all	To turn off any active debug commands, enter this command.

Caution *The overhead associated with the **debug** command takes up a large portion of your product's resources and at times can halt other processes. It is best to only use the **debug** command during times when the network resources are in low demand (non-peak hours, weekends, etc.).*

Understanding CLI Error Messages

The following table lists and defines some of the more common error messages given in the CLI.

Message	Helpful Hints
%Ambiguous command %Unrecognized Command	The command may not be valid in the current command mode, or you may not have entered enough correct characters for the command to be recognized. Try using the "?" command to determine your error. See <i>Using CLI Shortcuts</i> on page 6 for more information.
%Invalid or incomplete command	The command may not be valid in the current command mode, or you may not have entered all of the pertinent information required to make the command valid. Try using the "?" command to determine your error. See <i>Using CLI Shortcuts</i> on page 6 for more information.
%Invalid input detected at "^" marker	The error in command entry is located where the caret (^) mark appears. Enter a question mark at the prompt. The system will display a list of applicable commands or will give syntax information for the entry.

COMMAND DESCRIPTIONS

This portion of the guide provides a detailed listing of all available commands for the CLI (organized by command set). Each command listing contains pertinent information including the default value, a description of all sub-command parameters, functional notes for using the command, and a brief technology review. To search for a particular command alphabetically, use the [Index](#). To search for information on a group of commands within a particular command set, use the linked references given below:

Basic Mode Command Set [on page 10](#)
Enable Mode Command Set [on page 20](#)
Global Configuration Mode Command Set [on page 236](#)
DHCP Pool Command Set [on page 394](#)
HDLC Command Set [on page 411](#)
IKE Policy Command Set [on page 462](#)
IKE Policy Attributes Command Set [on page 473](#)
IKE Client Command Set [on page 479](#)
Crypto Map IKE Command Set [on page 483](#)
Crypto Map Manual Command Set [on page 492](#)
Radius Group Command Set [on page 503](#)
CA Profile Configuration Command Set [on page 505](#)
Certificate Configuration Command Set [on page 516](#)
Ethernet Interface Configuration Command Set [on page 520](#)
Serial Interface Configuration Command Set [on page 570](#)
T1 Interface Configuration Command Set [on page 579](#)
DSX-1 Interface Configuration Command Set [on page 594](#)
E1 Interface Configuration Command Set [on page 604](#)
G.703 Interface Configuration Command set [on page 619](#)
Modem Interface Configuration Command Set [on page 626](#)
BRI Interface Configuration Command set [on page 630](#)
Frame Relay Interface Config Command Set [on page 643](#)
Frame Relay Sub-Interface Config Command Set [on page 663](#)
ATM Interface Config Command Set [on page 722](#)
ATM Sub-Interface Config Command Set [on page 725](#)
ADSL Interface Config Command Set [on page 776](#)
BGP Configuration Command Set [on page 780](#)
BGP Neighbor Configuration Command Set [on page 786](#)
Tunnel Configuration Command Set [on page 800](#)
PPP Interface Configuration Command Set [on page 851](#)
Loopback Interface Configuration Command Set [on page 916](#)
Line (Console) Interface Config Command Set [on page 943](#)
Line (Telnet) Interface Config Command Set [on page 954](#)
Router (RIP) Configuration Command Set [on page 961](#)
Router (OSPF) Configuration Command Set [on page 970](#)
Quality of Service (QoS) Map Commands [on page 984](#)
Common Commands [on page 995](#)

BASIC MODE COMMAND SET

To activate the Basic mode, simply log in to the unit. After connecting the unit to a VT100 terminal (or terminal emulator) and activating a terminal session, the following prompt displays:

ProCurve>

The following command is common to multiple command sets and is covered in a centralized section of this guide. For more information, refer to the section listed below:

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

enable [on page 11](#)

logout [on page 12](#)

ping <address> [on page 13](#)

show clock [on page 15](#)

show snmp [on page 16](#)

show version [on page 17](#)

telnet <address> [on page 18](#)

traceroute <address> [on page 19](#)

enable

Use the **enable** command (at the Basic Command mode prompt) to enter the Enable Command mode. Use the **disable** command to exit the Enable Command mode.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Functional Notes

The Enable Command mode provides access to operating and configuration parameters and should be password protected to prevent unauthorized use. Use the **enable password** command (found in the Global Configuration) to specify an Enable Command mode password. If the password is set, access to the Enable Commands (and all other “privileged” commands) is only granted when the correct password is entered.

Usage Examples

The following example enters the Enable Command mode and defines an Enable Command mode password:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#enable password password
```

At the next login, the following sequence must occur:

```
ProCurve>enable  
Password: *****  
ProCurve#
```

logout

Use the **logout** command to terminate the current session and return to the login screen.

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following example shows the logout command being executed in the Basic mode:

```
ProCurve>logout
```

```
Session now available
```

```
Press RETURN to get started.
```

ping <address>

Use the **ping** command (at the Basic Command mode prompt) to verify IP network connectivity.

Syntax Description

<address>	Optional. Specifies the IP address of the system to ping. Entering the ping command with no specified address prompts the user with parameters for a more detailed ping configuration. See Functional Notes (below) for more information.

Default Values

No default value necessary for this command.

Functional Notes

The **ping** command helps diagnose basic IP network connectivity using the Packet InterNet Groper program to repeatedly bounce Internet Control Message Protocol (ICMP) Echo_Request packets off a system (using a specified IP address). The Secure Router OS allows executing a standard **ping** request to a specified IP address or provides a set of prompts to configure a more specific **ping** configuration.

The following is a list of output messages from the **ping** command:

!

Success

-

Destination Host Unreachable

\$

Invalid Host Address

X

TTL Expired in Transit

?

Unknown Host

*

Request Timed Out

The following is a list of available extended **ping** fields with descriptions:

Target IP address:

Specifies the IP address of the system to ping.

Repeat Count:

Number of ping packets to send to the system (valid range: 1 to 1000000).

Datagram Size:

Size (in bytes) of the ping packet (valid range: 1 to 1448).

Timeout in Seconds:

If a ping response is not received within the timeout period, the ping is considered unsuccessful (valid range: 1 to 5 seconds).

Extended Commands:

Specifies whether additional commands are desired for more ping configuration parameters.

Source Address (or interface):

Specifies the IP address to use as the source address in the ECHO_REQ packets.

Data Pattern:

Specify an alphanumerical string to use (the ASCII equivalent) as the data pattern in the ECHO_REQ packets.

Sweep Range of Sizes:

Varies the sizes of the ECHO_REQ packets transmitted.

Sweep Min Size:

Specifies the minimum size of the ECHO_REQ packet (valid range: 0 to 1448).

Sweep Max Size:

Specifies the maximum size of the ECHO_REQ packet (valid range: Sweep Min Size to 1448).

Sweep Interval:

Specifies the interval used to determine packet size when performing the sweep (valid range: 1 to 1448).

Verbose Output:

Specifies an extended results output.

Usage Examples

The following is an example of a successful **ping** command:

ProCurve>**ping**

Target IP address:**192.168.0.30**

Repeat count[1-1000000]:**5**

Datagram Size [1-1000000]:**100**

Timeout in seconds [1-5]:**2**

Extended Commands? [y or n]:**n**

Type CTRL+C to abort.

Legend: '!' = Success '?' = Unknown host '\$' = Invalid host address

 '*' = Request timed out '-' = Destination host unreachable

 'x' = TTL expired in transit

Pinging 192.168.0.30 with 100 bytes of data:

!!!!

Success rate is 100 percent (5/5) round-trip min/avg/max = 19/20.8/25 ms

show clock

Use the **show clock** command to display the system time and date entered using the **clock set** command.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example displays the current time and data from the system clock:

```
ProCurve>show clock
```

```
23:35:07 Tue Aug 20 2002
```

show snmp

Use the **show snmp** command to display the system Simple Network Management Protocol (SNMP) parameters and current status of SNMP communications.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is an example output using the **show snmp** command for a system with SNMP disabled and the default chassis and contact parameters:

```
ProCurve>show snmp
```

```
Chassis: Chassis ID
```

```
Contact Name:
```

```
Contact Phone:
```

```
Contact Email:
```

```
Contact Pager:
```

```
Management URL:
```

```
Management URL Label:
```

```
0 Rx SNMP packets
```

```
    0 Bad community names
```

```
    0 Bad community uses
```

```
    0 Bad versions
```

```
    0 Silent drops
```

```
    0 Proxy drops
```

```
    0 ASN parse errors
```


show version

Use the **show version** command to display the current Secure Router OS version information.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **show version** output:

>show version

```
ProCurve Secure Router 7203dl
SROS Version: J03.01
  Checksum: 4F8DCF96, built on: Tue Dec 21 08:32:18 2004
Boot ROM version J03.01
  Checksum: B133, built on: Tue Dec 21 08:32:25 2004
Copyright (c) 2004-2005, Hewlett-Packard, Co.
Platform: ProCurve Secure Router 7203dl
Serial number US449TS058
Flash: 33554432 bytes  DRAM: 268435455 bytes
```

System uptime is 0 days, 0 hours, 22 minutes, 42 seconds

Current system image file:	"CFLASH:/SROS.BIZ
Current configuration-file:	CFLASH:/startup-config"
Configured system image path:	
Primary:	"CFLASH:/SROS.BIZ"
Backup:	"NONVOL:/SROS.BIZ"
Configured configuration-file path:	
Primary:	"CFLASH:/startup-config"
Backup:	"NONVOL:/startup-config"

ProCurve#

telnet <address>

Use the **telnet** command to open a Telnet session (through the Secure Router OS) to another system on the network.

Syntax Description

<address> Specifies the IP address of the remote system.

Default Values

No default value necessary for this command.

Usage Examples

The following example opens a Telnet session with a remote system (**10.200.4.15**):

ProCurve>**telnet 10.200.4.15**

User Access Login

Password:

traceroute <address>

Use the **traceroute** command to display the IP routes a packet takes to reach the specified destination.

Syntax Description

<address> Specifies the IP address of the remote system to trace the routes to

Default Values

No default value necessary for this command.

Usage Examples

The following example performs a traceroute on the IP address 192.168.0.1:

```
ProCurve>traceroute 192.168.0.1
```

Type CTRL+C to abort.

Tracing route to 192.168.0.1 over a maximum of 30 hops

```
 1  22ms  20ms  20ms   192.168.0.65
 2  23ms  20ms  20ms   192.168.0.1
```

```
ProCurve>
```

ENABLE MODE COMMAND SET

To activate the Enable mode, enter the **enable** command at the Basic mode prompt. (If an enable password has been configured, a password prompt will display.) For example:

```
ProCurve>enable
Password: XXXXXXXX
ProCurve#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the section listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
exit [on page 1003](#)
ping <address> [on page 1004](#)
show running-config [on page 1006](#)

All other commands for this command set are described in this section in alphabetical order.

autosynch [on page 22](#)
clear commands [begin on page 24](#)
clock auto-correct-dst [on page 51](#)
clock no-auto-correct-dst [on page 52](#)
clock set <time> <day> <month> <year> [on page 53](#)
clock timezone <text> [on page 54](#)
configure [on page 56](#)
copy commands [begin on page 57](#)
debug commands [begin on page 74](#)
dir [*] [on page 114](#)
dir [cflash | flash] [*] [on page 115](#)
disable [on page 116](#)
erase [SROS.BIZ | startup-config] [on page 117](#)
erase [cflash | flash] <filename> [on page 118](#)
erase file-system cflash [on page 119](#)
events [on page 120](#)
exception report generate [on page 121](#)
logout [on page 123](#)
reload [cancel | in <delay>] [on page 124](#)
show commands [begin on page 125](#)

telnet <address> [on page 230](#)

terminal length <lines> [on page 231](#)

traceroute <address> [on page 232](#)

undebg all [on page 233](#)

wall <message> [on page 234](#)

write [*erase* | *memory* | *network* | *terminal*] [on page 235](#)

autosynch

Use the **autosynch** command to force a synchronization of the **SROS.BIZ** and **startup-config** files located in system flash and compact flash memory.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Functional Notes

The AutoSynch™ features configures the system to synchronize the **startup-config** and **SROS.BIZ** files located in the system flash memory and the compact flash card. When enabled, the system compares the two files in the two locations and replaces the files located in the system flash memory with the ones from the compact flash card (regardless of which set of files is more current). This allows the customer to maintain the version of the operating system, and the configuration for that operating system, at the desired level. To accomplish this, a synchronization check is performed on the system any time there is a change in **startup-config** or **SROS.BIZ** on the compact flash card.

The autosynch feature allows for quick installation and updates of routers by inserting a compact flash card containing the desired software (must be renamed from the desired operating system software, such as J03.01.biz to **SROS.BIZ**) and startup configuration file (must be named **startup-config**) into a router with AutoSynch™ enabled. The ProCurve Secure Routers automatically boot from the compact flash (and secondarily from internal flash). After booting, with AutoSynch™ enabled, the router will synchronize the files in system flash memory with the desired files from compact flash.

Caution *Deleting the **SROS.BIZ** and **startup-config** files from the compact flash card (using the **erase** command) deletes the files from the system flash memory as well.*

Status commands associated with the AutoSynch™ feature include **show version flash SROS.BIZ** and **show autosynch-status**.

The **show version flash SROS.BIZ** command opens the specified .biz file and returns the current Secure Router OS version information.

ProCurve>enable

ProCurve#show version flash SROS.BIZ

Version:J03.01.00

The **show autosynch-status** command displays the current AutoSynch™ configuration and the statistics for the **SROS.BIZ** and **startup-config** files (if AutoSynch™ is enabled).

ProCurve>enable

ProCurve#show autosynch-status

AutoSynch: Mode - Enabled

AutoSynch: SROS.BIZ synched

AutoSynch: startup-config synched

Usage Examples

The following example forces a synchronization of **startup-config** and **SROS.BIZ** (located in system flash and compact flash memory):

ProCurve>enable

ProCurve#autosynch

AutoSynch: SROS.BIZ synched

AutoSynch: startup-config synched

ProCurve#

clear access-list <listname>

Use the **clear access-list** command to clear all counters associated with all access lists (or a specified access list).

Syntax Description

<i><listname></i>	Optional. Specifies the name (label) of an access list

Default Values

No default value necessary for this command.

Usage Examples

The following example clears all counters for the access list labeled **MatchAll**:

```
ProCurve>enable
```

```
ProCurve#clear access-list MatchAll
```


clear arp-cache

Use the **clear arp-cache** command to remove all dynamic entries from the Address Resolution Protocol (ARP) cache table.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example removes all dynamic entries from the ARP cache:

```
ProCurve>enable
```

```
ProCurve#clear arp-cache
```

clear arp-entry *<address>*

Use the **clear arp-entry** command to remove a single entry from the Address Resolution Protocol (ARP) cache.

Syntax Description

<i><address></i>	Specifies the IP address of the entry to remove.

Default Values

No default value necessary for this command.

Usage Examples

The following example removes the entry for 10.200.4.56 from the ARP cache:

```
ProCurve>enable
```

```
ProCurve#clear arp-entry 10.200.4.56
```

clear bridge <group#>

Use the **clear bridge** command to clear all counters associated with bridging (or for a specified bridge-group).

Syntax Description

<group#>	Optional. Specifies a single bridge group (1-255).

Default Values

No default value necessary for this command.

Usage Examples

The following example clears all counters for bridge group 17:

```
ProCurve>enable
```

```
ProCurve#clear bridge 17
```

clear buffers max-used

Use the **clear buffers max-used** command to clear the maximum-used statistics for buffers displayed in the **show memory heap** command.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

ProCurve>enable

ProCurve#clear buffers max-used

clear counters *<interface>*

Use the **clear counters** command to clear all interface counters (or the counters for a specified interface).

Default Values

No default value necessary for this command.

Usage Examples

The following example clears all counters associated with the Ethernet 0/1 interface:

```
ProCurve>enable
```

```
ProCurve#clear counters ethernet 0/1
```

clear crypto ike sa *<policy priority>*

Use the **clear crypto ike sa** command to clear existing IKE security associations (SAs), including active ones.

Syntax Description

<i><policy priority></i>	Optional. Clears out all existing IKE SAs associated with the designated policy priority. This number is assigned using the crypto ike policy command.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears the entire database of IKE SAs (including the active associations):

ProCurve>enable

ProCurve#clear crypto ike sa

clear crypto ipsec sa

Use the **clear crypto ipsec sa** command to clear existing IPSec security associations (SAs), including active ones.

Variations of this command include the following:

clear crypto ipsec sa

clear crypto ipsec sa entry *<ip address>* **ah** *<SPI>*

clear crypto ipsec sa entry *<ip address>* **esp** *<SPI>*

clear crypto ipsec sa map *<map name>*

clear crypto ipsec sa peer *<ip address>*

Syntax Description

entry <i><ip address></i>	Clears only the SAs related to a certain destination IP address.
ah <i><SPI></i>	Clears only a portion of the SAs by specifying the AH (authentication header) protocol and a security parameter index (SPI). You can determine the correct SPI value using the show crypto ipsec sa command.
esp <i><SPI></i>	Clears only a portion of the SAs by specifying the ESP (encapsulating security payload) protocol and a security parameter index (SPI). You can determine the correct SPI value using the show crypto ipsec sa command.
map <i><map name></i>	Clears only the SAs associated with the crypto map name given.
peer <i><ip address></i>	Clears only the SAs associated with the far-end peer IP address given.

Default Values

No default value necessary for this command.

clear dump-core

The **clear dump-core** command clears diagnostic information appended to the output of the **show version** command. This information results from an unexpected unit reboot.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears the entire database of IKE SAs (including the active associations):

```
ProCurve>enable
```

```
ProCurve#clear dump-core
```

clear event-history

Use the **clear event-history** command to clear all messages logged to the local event-history.

Warning *Messages cleared from the local event-history (using the **clear event-history** command) are no longer accessible.*

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears all local event-history messages:

ProCurve>enable

ProCurve#clear event-history

clear host [* | <hostname>]

Use the **clear host** command to clear a hostname (and associated address) from the DNS host-to-address table.

Syntax Description

*	Clears all hosts from the host table.
<hostname>	Specifies a specific host entry to remove from the host-to-address table.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears all hostnames:

```
ProCurve>enable
ProCurve#clear host *
```

clear ip bgp [* | <as-number> | <ip address>] [in | out | soft]

Use the **clear ip bgp** command to clear BGP neighbors as specified.

Syntax Description

*	Clears all BGP neighbors.
<as-number>	Clears all BGP neighbors with the specified AS number (Range: 1 to 65,535).
<ip address>	Clears the BGP neighbor with the specified IP address.
in	Causes a “soft” reset inbound with a neighbor, reprocessing routes advertised by that neighbor.
out	Causes a “soft” reset outbound with a neighbor, re-sending advertised routes to that neighbor.
soft	Causes a “soft” reset both inbound and outbound.

Default Values

No default value necessary for this command.

Functional Notes

The **clear ip bgp** command must be issued to re-initialize the BGP process between the peers matching the given arguments. Most neighbor changes, including changes to prefix-list filters, do not take effect until the **clear** command is issued. A hard reset clears the TCP connection with the specified peers, which results in clearing the table. This method of clearing is disruptive and causes peer routers to record a route flap for each route.

The **out** version of this command provides a soft reset out to occur by causing all routes to be re-sent to the specified peer(s). TCP connections are not torn down, so this method is less disruptive. Output filters/policies are re-applied before sending the update.

The **in** version of this command provides a soft reset in to occur by allowing the router to receive an updated table from a peer without tearing down the TCP connection. This method is less disruptive and does not count as a route flap. Currently, all of the peer's routes are stored permanently, even if they are filtered by a prefix list. The command causes the peer's routes to be reprocessed with any new parameters.

Usage Examples

The following example clears the information for all peers with an AS number of 101:

```
ProCurve>enable
ProCurve#clear ip bgp 101
```

clear ip dhcp-server binding [* | <ip address>]

Use the **clear ip dhcp-server binding** command to clear Dynamic Host Configuration Protocol (DHCP) server binding entries from the database.

Syntax Description

*	Clears all automatic binding entries.
<ip address>	Clears a specific binding entry. Enter the source IP address (format is A.B.C.D).

Default Values

No default value necessary for this command.

Functional Notes

A DHCP server binding represents an association between a MAC address and an IP address that was offered by the unit to a DHCP client (i.e., most often a PC). Clearing a binding allows the unit to offer that IP address again, should a request be made for one.

Usage Examples

The following example clears a DHCP server binding for the IP address **125.25.47.4**:

ProCurve>enable

ProCurve#clear ip dhcp-server binding 125.25.47.4

clear ip igmp group [*<group-address>* | *<interface>*]

Use the **clear ip igmp group** command to clear entries from the Internet Group Management Protocol (IGMP) tables. If no address or interface is specified, all non-static IGMP groups are cleared with this command.

Syntax Description

<i><group-address></i>	Optional. Specifies the multicast IP address of the multicast group.
<i><interface></i>	Optional. Designates the clearing of parameters for a specific interface (in the format type slot/port). For example: eth 0/1 .

Default Values

No default value necessary for this command.

Usage Examples

The following example shows output for the **show igmp groups** command before and after a **clear ip igmp group** command is issued. This example clears the IGMP entry that was registered dynamically by a host. Interfaces that are statically joined are not cleared:

ProCurve>enable

ProCurve#show ip igmp groups

Group Address	Interface	Uptime	Expires	Last Reporter
172.0.1.50	Loopback100	01:22:59	00:02:46	172.23.23.1
172.1.1.1	Ethernet0/1	00:00:14	00:02:45	1.1.1.2
172.1.1.1	Loopback100	01:22:59	00:02:46	172.23.23.1

ProCurve#clear ip igmp group

ProCurve#show ip igmp groups

Group Address	Interface	Uptime	Expires	Last Reporter

This version of the command clears all dynamic groups that have the specified output interface (Ethernet 0/1):

ProCurve#clear ip igmp group ethernet 0/1

This version of the command clears the specified group on all interfaces where it is dynamically registered:

ProCurve#clear ip igmp group 172.1.1.1

clear ip policy-sessions

Use the **clear ip policy-sessions** command to clear policy class sessions. You may clear all the sessions or a specific session. Refer to the **show ip policy-sessions** for a current session listing. The following lists the complete syntax for the **clear ip policy-sessions** commands:

clear ip policy-sessions

```
clear ip policy-sessions <classname> [ahp | esp | gre | icmp | tcp | udp | <protocol>] <source ip>
<source port><dest ip><dest port>
```

```
clear ip policy-sessions <classname> [ahp | esp | gre | icmp | tcp | udp | <protocol>] <source ip>
<source port><dest ip><dest port> [destination | source] <nat ip><nat port>
```

Syntax Description

<classname>	Alphanumeric descriptor for identifying the configured access policy (access policy descriptors are not case-sensitive).
<protocol>	A specific protocol (Range: 0 to 255).
<source ip>	Specifies the source IP address (format is A.B.C.D).
<source port>	Specifies the source port (in hexadecimal format for ahp, esp, and gre; decimal for all other protocols).
<dest ip>	Specifies the destination IP address (format is A.B.C.D).
<dest port>	Specifies the destination port (in hex format for ahp, esp, and gre; decimal for all other protocols).
[destination source]	For NAT sessions, this specifies whether to select a NAT source or NAT destination session.
<nat ip>	For NAT sessions, this specifies the NAT IP address (format is A.B.C.D).
<nat port>	For NAT sessions, this specifies the NAT port (in hex format for ahp, esp, and gre; decimal for all other protocols).

Default Values

No default value necessary for this command.

Functional Notes

The second half of this command, beginning with the source IP address may be copied and pasted from a row in the **show ip policy-sessions** table for easier use.

Usage Examples

The following example clears the Telnet association (TCP port 23) for policy class "pclass1" with source IP address 192.22.71.50 and destination 192.22.71.130:

ProCurve>enable

ProCurve#clear ip policy-sessions pclass1 tcp 192.22.71.50 23 192.22.71.130 23

clear ip policy-stats *<classname>* **entry** *<policy class #>*

Use the **clear ip policy-stats** command to clear statistical counters for policy classes.

Syntax Description

<i><classname></i>	Optional. Specifies the policy class to clear. If no policy class is specified, statistics are cleared for all policies.
entry	Optional. Use this optional keyword to clear statistics of a specific policy class entry
<i><policy class #></i>	Optional. Specifies the policy class entry number.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears statistical counters for all policy classes:

```
ProCurve>enable
ProCurve#clear ip policy-stats
```

The following example clears statistical counters for the policy class **MatchALL**:

```
ProCurve>enable
ProCurve#clear ip policy-stats MatchALL
```


clear ip prefix-list <listname>

Use the **clear ip prefix-list** command to clear the IP prefix list hit count shown in the **show ip prefix-list detail** command output. See *show ip prefix-list [detail | summary] <listname>* [on page 187](#).

Syntax Description

<listname>	Specifies hit count statistics of the IP prefix list to clear.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears the hit count statistics for prefix list **test**:

ProCurve>enable

ProCurve#clear ip prefix-list test

clear ip route

Use the **clear ip route** command to remove all learned routes from the IP route table. Static and connected routes are not cleared by this command.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example removes all learned routes from the route table:

```
ProCurve>enable
```

```
ProCurve#clear ip route *
```

clear lldp counters

Use the **clear lldp counters** command to reset all LLDP packet counters to 0 on all interfaces.

Syntax Description

No subcommands.

Default Values

There are no default settings for this command.

Usage Examples

The following example resets all LLDP counters:

```
ProCurve>enable
```

```
ProCurve#clear lldp counters
```

clear lldp counters interface <interface>

Use the **clear lldp counters interface** command to reset all LLDP packet counters to 0 for a specified interface.

Syntax Description

<interface>	Clears the information for the specified interface. Type clear lldp counters interface ? for a complete list of applicable interfaces.

Default Values

No default values are necessary for this command.

Usage Examples

The following example resets the counters on a PPP interface:

ProCurve>enable

ProCurve#clear lldp counters interface ppp 1

clear lldp neighbors

Use the **clear lldp neighbors** command to remove all neighbors from this unit's database. As new LLDP packets are received, the database will contain information about neighbors included in those frames.

Syntax Description

No subcommands.

Default Values

There are no default settings for this command.

Functional Notes

This command generates output indicating the names of any neighbors deleted from the database and the name of the interface on which the neighbor was learned.

Usage Examples

The following example clears LLDP neighbor Switch_1 from the Ethernet interface 0/1:

```
ProCurve>enable
```

```
ProCurve#clear lldp neighbors
```

```
LLDP: Deleted neighbor "Switch_1" on interface eth 0/1
```

```
ProCurve#
```

clear pppoe *<ppp interface #>*

Use the **clear pppoe** command to terminate the current PPPoE client session and cause the Secure Router OS to attempt to re-establish the session.

Syntax Description

<ppp interface #> PPP interface number.

Default Values

No default value necessary for this command.

Usage Examples

The following example ends the current PPPoE client session for ppp 1:

```
ProCurve>enable
```

```
ProCurve#clear pppoe 1
```

clear process cpu max

Use the **clear process cpu max** command to clear the maximum CPU usage statistic which is displayed in the **show process cpu** command output.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example resets the CPU maximum usage statistics:

```
ProCurve>enable
```

```
ProCurve#clear process cpu max
```

clear qos map

Use the **clear qos map** command to clear the statistics for all defined QoS maps or to view detailed information for maps meeting user-configured specifications.

Variations of this command include the following:

clear qos map <map name>

clear qos map <map name> <sequence number>

clear qos map interface <interface id>

Syntax Description

<map name>	Specifies the name of a defined QoS map.
<sequence number>	Specifies one of the map's defined sequence numbers.
<interface>	Specifies an interface for which to clear QoS map statistics (for just that interface). Type clear cos map ? for a complete list of applicable interfaces.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears statistics for all defined QoS maps:

ProCurve#clear qos map

The following example clears statistics for all entries in the **priority** QoS map:

ProCurve#clear qos map priority

The following example clears statistics in entry **10** of the **priority** QoS map:

ProCurve#clear qos map priority 10

The following example clears QoS statistics for a specified interface:

ProCurve#clear qos map interface frame-relay 1

Note *The **clear counters** command clears ALL interface statistics (including QoS map interface statistics).*

clear spanning-tree counters [interface <interface id>]

The **clear spanning-tree counters** command clears the following counts: BPDU transmit, BPDU receive, and number of transitions to forwarding state.

Syntax Description

interface <interface id> Optional. Specifies a single interface. Enter **clear spanning-tree counters ?** for a complete list of applicable interfaces.

Default Values

No default value necessary for this command.

Usage Examples

The following example clears the spanning tree counters for Ethernet 0/1:

ProCurve>enable

ProCurve#clear spanning-tree counters interface eth 0/1

clear spanning-tree detected-protocols [interface <interface id>]

Use the **clear spanning-tree detected-protocols** command to restart the protocol migration process.

Syntax Description

interface	Optional. Choose the Ethernet interface.
<interface id>	Optional. Enter a valid interface ID (e.g., 0/1 for Ethernet 0/1).

Default Values

No default value necessary for this command.

Functional Notes

The ProCurve Secure Router has the ability to operate using the rapid spanning-tree protocol or the legacy 802.1D version of spanning-tree. When a BPDU (bridge protocol data unit) of the legacy version is detected on an interface, the ProCurve Secure Router automatically regresses to using the 802.1D spanning-tree protocol for that interface. Issue the **clear spanning-tree detected-protocols** command to return to rapid spanning-tree operation.

Usage Examples

The following example re-initiates the protocol migration process on eth 0/2:

ProCurve>enable

ProCurve#clear spanning-tree detected-protocols interface ethernet 0/2

The following example re-initiates the protocol migration process on all interfaces:

ProCurve>enable

ProCurve#clear spanning-tree detected-protocols

clock auto-correct-dst

The **clock auto-correct-dst** command allows the automatic one-hour correction for Daylight Saving Time (DST). Use the **clock no-auto-correct-dst** command to disable this feature.

Syntax Description

No subcommands.

Default Values

By default this command is enabled.

Usage Examples

The following example allows for automatic DST correction:

```
ProCurve>enable
```

```
ProCurve#clock auto-correct-dst
```

clock no-auto-correct-dst

The **clock no-auto-correct-dst** command allows you to override the automatic one-hour correction for Daylight Saving Time (DST).

Syntax Description

No subcommands.

Default Values

No default value is necessary for this command.

Functional Notes

Many time zones include an automatic one-hour correction for daylight saving time at the appropriate time. You may override it at your location using this command.

Usage Examples

The following example overrides the one-hour offset for DST:

```
ProCurve>enable
```

```
ProCurve#clock no-auto-correct-dst
```

clock set *<time>* *<day>* *<month>* *<year>*

Use the **clock set** command to configure the system software clock. For the command to be valid, all fields must be entered. See the **Usage Example** below for an example.

Syntax Description

<i><time></i>	Sets the time (in 24-hr format) of the system software clock in the format HH:MM:SS (hours:minutes:seconds).
<i><day></i>	Sets the current day of the month (Range: 1 to 31).
<i><month></i>	Sets the current month (Range: January to December). You need only enter enough characters to make the entry unique. This entry is not case-sensitive.
<i><year></i>	Sets the current year (Range: 2000 to 2100).

Default Values

No default value necessary for this command.

Usage Examples

The following example sets the system software clock for 3:42 pm, August 22 2004:

ProCurve>enable

ProCurve#clock set 15:42:00 22 Au 2004

clock timezone <text>

The **clock timezone** command sets the unit's internal clock to the timezone of your choice. This setting is based on the difference in time (in hours) between Greenwich Mean Time (GMT) or Central Standard Time (CST) and the timezone for which you are setting up the unit. Use the **no** form of this command to disable this feature.

Syntax Description

<text>	Specifies the difference in time (in hours) between Greenwich Mean Time (GMT) or Central Standard Time (CST) and the timezone for which you are setting up the unit.

Default Values

No default value is necessary for this command.

Functional Notes

The following list shows sample cities and their timezone codes.

clock timezone +1-Amsterdam	clock timezone +8-Beijing
clock timezone +1-Belgrade	clock timezone +8-Irkutsk
clock timezone +1-Brussels	clock timezone +8-Kuala-Lumpur
clock timezone +1-Sarajevo	clock timezone +8-Perth
clock timezone +1-West-Africa	clock timezone +8-Taipei
clock timezone +10-Brisbane	clock timezone +9-Osaka
clock timezone +10-Canberra	clock timezone +9-Seoul
clock timezone +10-Guam	clock timezone +9-Yakutsk
clock timezone +10-Hobart	clock timezone +9:30-Adelaide
clock timezone +10-Vladivostok	clock timezone +9:30-Darwin
clock timezone +11	clock timezone -1-Azores
clock timezone +12-Auckland	clock timezone -1-Cape-Verde
clock timezone +12-Fiji	clock timezone -10
clock timezone +13	clock timezone -11
clock timezone +2-Athens	clock timezone -12
clock timezone +2-Bucharest	clock timezone -2

Functional Notes (cont'd)

clock timezone +2-Cairo	clock timezone -3-Brasilia
clock timezone +2-Harare	clock timezone -3-Buenos-Aires
clock timezone +2-Helsinki	clock timezone -3-Greenland
clock timezone +2-Jerusalem	clock timezone -3:30
clock timezone +3-Baghdad	clock timezone -4-Atlantic-Time
clock timezone +3-Kuwait	clock timezone -4-Caracus
clock timezone +3-Moscow	clock timezone -4-Santiago
clock timezone +3-Nairobi	clock timezone -5
clock timezone +3:30	clock timezone -5-Bogota
clock timezone +4-Abu-Dhabi	clock timezone -5-Eastern-Time
clock timezone +4-Baku	clock timezone -6-Central-America
clock timezone +4:30	clock timezone -6-Central-Time
clock timezone +5-Ekaterinburg	clock timezone -6-Mexico-City
clock timezone +5-Islamabad	clock timezone -6-Saskatchewan
clock timezone +5:30	clock timezone -7-Arizona
clock timezone +5:45	clock timezone -7-Mountain-Time
clock timezone +6-Almaty	clock timezone -8
clock timezone +6-Astana	clock timezone -9
clock timezone +6-Sri-Jay	clock timezone GMT-Casablanca
clock timezone +6:30	clock timezone GMT-Dublin
clock timezone +7-Bangkok	
clock timezone +7-Kranoyarsk	

Usage Examples

The following example sets the timezone for Santiago, Chile.

```
ProCurve>enable
ProCurve#clock timezone -4-Santiago
```

configure

Use the **configure** command to enter the Global Configuration mode or to configure the system from memory. See *Global Configuration Mode Command Set* [on page 236](#) for more information.

Syntax Description

terminal	Enters the Global Configuration mode.
memory	Configures the active system with the commands located in the default configuration file stored in flash memory.
network	Configures the system from a TFTP network host.
overwrite-network	Overwrites flash memory from a TFTP network host.

Default Values

No default value necessary for this command.

Usage Examples

The following example enters the Global Configuration mode from the Enable Command mode:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#
```


copy [cflash | flash] <filename> boot

Use the **copy boot** commands to copy the specified file (located in flash memory or on the compact flash card) to the Boot ROM.

Syntax Description

cflash	Specifies the memory location for the file to copy as compact flash memory.
flash	Specifies the memory location for the file to copy as flash memory.
<filename>	Specifies the name of the source file to copy.
boot	Specifies the destination memory location for the file copy as the Boot ROM.

Default Values

No default value necessary for this command.

Functional Notes

Updates to the Boot ROM are required periodically to enhance and expand the router's operation. The Boot Code can now be updated from within the Command Line Interface (CLI) (beginning with software release J03.01) using the **copy boot** command.

Usage Examples

The following example copies the file **J03_01-boot.biz** (located on the compact flash card) to the Boot ROM:

ProCurve>enable

ProCurve#copy cflash J03_01-boot.biz boot

Upgrading boot code is a critical process that cannot be interrupted. If something were to happen and the process was not able to be completed, it would render your unit inoperable. It is for this reason that during a bootcode upgrade, all other system tasks will be halted. This means packets will not be routed, and all console sessions will not respond during the upgrade process. Once the process finishes, the system will function as it did before. This process will take approximately 20 seconds. **DO NOT REMOVE COMPACT FLASH DURING THIS OPERATION!!**

Do you want to proceed? [yes/no]y

WARNING!! A bootcode upgrade has been initiated. Your session will become nonresponsive for the duration of the upgrade (approx. 20 seconds). A message will be sent when the upgrade is completed.

Bootcode upgrade process done. Your session should function normally.

Reading 318189 bytes of code, stand by...

Image is compressed, inflating.....

Verifying image

Erasing boot sector

Programming boot sector

Success!!!

copy cflash <filename> [cflash | flash] <filename>

Use the **copy cflash** command to copy file located on the compact flash card to a specified destination.

Syntax Description

<filename>	Specifies the name of the file (located on the compact flash card) to copy.
cflash	Specifies the destination memory location for the file copy as compact flash memory.
flash	Specifies the destination memory location for the file copy as flash memory.
<filename>	Specifies the filename for the copy.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the file **myfile.biz** (located on the compact flash card) to flash memory (naming the new file **newfile.biz**):

ProCurve>enable

ProCurve#copy cflash myfile.biz flash newfile.biz

The following example creates a copy of the file **myfile.biz** (located on the compact flash card), names the new file **newfile.biz**, and places the new file on the installed compact flash card:

ProCurve>enable

ProCurve#copy cflash myfile.biz cflash newfile.biz

copy cflash tftp

Use the **copy cflash tftp** command to copy any file on the compact flash card to a specified Trivial File Transfer Protocol (TFTP) server.

Syntax Description

	After entering copy cflash tftp , the Secure Router OS prompts for the following information:
<i>Address of remote host:</i>	Specifies the IP address of the TFTP server.
<i>Source filename:</i>	Specifies the name of the file (located on the compact flash card) to copy to the TFTP server.
<i>Destination filename:</i>	Specifies the filename to use when storing the copied file on the TFTP server. (The file will be placed in the default directory established by the TFTP server.)

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the file **myfile.biz** (located on the compact flash card) to the specified TFTP server:

ProCurve>enable

ProCurve#copy cflash tftp

Address of remote host?**10.200.2.4**

Source filename **myfile.biz**

Destination filename **myfile.biz**

Initiating TFTP transfer...

Received 45647 bytes.

Transfer Complete!

copy cflash xmodem

Use the **copy cflash xmodem** command to copy any file on the compact flash card (using the XMODEM protocol) to the terminal connected to the console port. XMODEM capability is provided in terminal emulation software such as HyperTerminal™.

Syntax Description

	After entering copy cflash xmodem , the Secure Router OS prompts for the following information:
<i>Source filename:</i>	Specifies the name of the file to copy from compact flash to the connected terminal.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the file **myfile.biz** (located on the compact flash card) to the connected terminal using XMODEM protocol:

ProCurve>enable

ProCurve#copy cflash xmodem

Source filename **myfile.biz**

Begin the Xmodem transfer now...

Press CTRL+X twice to cancel

CCCCC

The Secure Router OS is now ready to transmit the file on the **CONSOLE** port (using the XMODEM protocol). The next step in the process may differ depending on the type of terminal emulation software you are using. For HyperTerminal, you will now select **Transfer > Receive File** and select the destination. Once the transfer is complete, information similar to the following is displayed:

Received 231424 bytes.

Transfer complete

copy console flash <filename>

Use the **copy console** command to copy the console's input to a text file. To end copying to the text file, type **<Ctrl+D>**. The file will be saved in the SROS root directory.

Syntax Description

<filename>	Specifies the destination file (located in flash memory) for console input.

Default Values

No default is necessary for this command.

Functional Notes

The **copy console** command works much like a line editor. Prior to pressing **<Enter>**, changes can be made to the text on the line. Changes can be made using **<Delete>** and **<Backspace>** keys. The text can be traversed using the arrow keys, **<Ctrl+A>** (to go to the beginning of a line), and **<Ctrl+E>** (to go to the end of a line). To end copying to the text file, type **<Ctrl+D>**. The file will be saved in the Secure Router OS root directory. Use the **dir** command to see a list of files in the root directory.

Usage Examples

The following example copies the console input into the file **config.txt** (located in the Secure Router OS root directory):

ProCurve>enable

ProCurve#copy console flash config.txt

copy flash <filename> [cflash | flash | startup-config] <filename>

Use the **copy flash** command to copy files located in flash memory to a specified destination.

Syntax Description

<filename>	Specifies the name of the file (located on the compact flash card) to copy.
cflash	Specifies the destination memory location for the file copy as compact flash memory.
flash	Specifies the destination memory location for the file copy as flash memory.
startup-config	Replaces the primary startup-configuration file with a copy of the specified file.
<filename>	Specifies the filename for the copy.

Default Values

No default value necessary for this command.

Usage Examples

The following example creates a copy of the file **myfile.biz** (located in flash memory), names the new file **newfile.biz**, and places the new file in flash memory:

```
ProCurve>enable
ProCurve#copy flash myfile.biz flash newfile.biz
```

The following example copies the file **myfile.biz** (located in flash memory) to compact flash memory (naming the new file **newfile.biz**):

```
ProCurve>enable
ProCurve#copy flash myfile.biz flash newfile.biz
```

The following example copies the file **new_startup_config** (located in flash memory) to the primary startup-configuration:

```
ProCurve>enable
ProCurve#copy flash new_startup_config startup-config
```

copy flash tftp

Use the **copy flash tftp** command to copy a file located in flash memory to a specified Trivial File Transfer Protocol (TFTP) server.

Syntax Description

	After entering copy flash tftp , the Secure Router OS prompts for the following information:
<i>Address of remote host:</i>	Specifies the IP address of the TFTP server.
<i>Source filename:</i>	Specifies the name of the file (located in flash memory) to copy to the TFTP server.
<i>Destination filename:</i>	Specifies the filename to use when storing the copied file on the TFTP server. (The file will be placed in the default directory established by the TFTP server.)

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the software file **J03_01.biz** located in flash memory to a TFTP server:

ProCurve>enable

ProCurve#copy flash tftp

Address of remote host?**10.200.2.4**

Source filename **J03_01.biz**

Destination filename **J03_01.biz**

Initiating TFTP transfer...

Sent 769060 bytes.

Transfer Complete!

copy flash xmodem

Use the **copy flash xmodem** command to copy any file located in flash memory (using the XMODEM protocol) to the terminal connected to the console port. XMODEM capability is provided in terminal emulation software such as HyperTerminal™.

Syntax Description

	After entering copy flash xmodem , the Secure Router OS prompts for the following information:
<i>Source filename:</i>	Specifies the name of the file (located in flash memory) to copy using XMODEM.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the software file **J03_01.biz** (located in flash memory) to the connected terminal using XMODEM protocol:

ProCurve>enable

ProCurve#copy flash xmodem

Source filename **J03_01.biz**

Begin the Xmodem transfer now...

Press CTRL+X twice to cancel

CCCCC

The Secure Router OS is now ready to transmit the file on the **CONSOLE** port (using the XMODEM protocol). The next step in the process may differ depending on the type of terminal emulation software you are using. For HyperTerminal, you will now select **Transfer > Receive File** and select the destination. Once the transfer is complete, information similar to the following is displayed:

Sent 231424 bytes.

Transfer complete

copy flash <filename> interface <type> <slot/port>

Use the **copy interface** command to copy a software file to a specified interface. This command is only valid for modules that contain module-specific software that is independent of the system software.

Syntax Description

<filename>	Specifies the filename of source software file. The source file MUST be located in flash memory. All module software files should have a .biz extension.
<type>	Specifies the type of module to update. Enter the ? to display a list of all available module types.
<slot/port>	Specifies the slot and port number of the interface.

Default Values

No default is necessary for this command.

Usage Examples

The following example updates the ADSL interface with the software file **J8459A_02_01_01.biz**:

ProCurve>enable

ProCurve#copy J8459A_02_01_01.biz interface adsl 0/1

copy running-config [cflash | flash | startup-config] <filename>

Use the **copy running-config** command to create a copy of the current running-configuration and replace the current startup-configuration or save it to a specified memory location (compact flash or system flash).

Syntax Description

cflash	Specifies the destination memory location for the copied file as compact flash.
flash	Specifies the destination memory location for the copied file as system flash memory.
startup-config	Replaces the primary startup-configuration file with the current running-configuration.
<filename>	Specifies the filename of the copied file. This is only valid when copying the running-configuration file to a specified memory location.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the current running-configuration file to the startup configuration file located in flash memory:

```
ProCurve>enable
ProCurve#copy running-config startup-config
Building configuration...
NONVOL:/startup-config
Done. Success!
```

The following example copies the current running-configuration file to compact flash memory and names the file **config_01.txt**:

```
ProCurve>enable
ProCurve#copy running-config cflash config_01.txt
Percent Complete 100%

ProCurve#
```

copy running-config tftp

Use the **copy running-config tftp** command to copy the current running-configuration file to a specified Trivial File Transfer Protocol (TFTP) server.

Syntax Description

	After entering copy running-config tftp , the Secure Router OS prompts for the following information:
<i>Address of remote host:</i>	Specifies the IP address of the TFTP server.
<i>Destination filename:</i>	Specifies the filename to use when storing the copied file on the TFTP server. (The file will be placed in the default directory established by the TFTP server.)

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the current running-configuration file to a TFTP server and names the file **config_01.txt**:

ProCurve>enable

ProCurve#copy running-config tftp

Address of remote host?**10.200.2.4**

Destination filename **config_01.txt**

Initiating TFTP transfer...

Sent 3099 bytes.

Transfer Complete!

copy running-config xmodem

Use the **copy running-config xmodem** command to copy the current running-configuration file (using the XMODEM protocol) to the terminal connected to the console port. XMODEM capability is provided in terminal emulation software such as HyperTerminal™.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the current running-configuration to the connected terminal using XMODEM protocol:

```
ProCurve>enable
```

```
ProCurve#copy running-config xmodem
```

```
Begin the Xmodem transfer now...
```

```
Press CTRL+X twice to cancel
```

```
CCCCCC
```

The Secure Router OS is now ready to transmit the file on the **CONSOLE** port (using the XMODEM protocol). The next step in the process may differ depending on the type of terminal emulation software you are using. For HyperTerminal, you will now select **Transfer > Receive File** and select the destination. Once the transfer is complete, information similar to the following is displayed:

```
Sent 3704 bytes.
```

```
Transfer complete
```

copy startup-config [cflash | flash | startup-config] <filename>

Use the **copy startup-config** command to create a copy of the current startup-configuration and replace the current running-configuration or save it to a specified memory location (compact flash or system flash).

Syntax Description

cflash	Specifies the destination memory location for the copied file as compact flash.
flash	Specifies the destination memory location for the copied file as system flash memory.
startup-config	Replaces the current running-configuration file with the primary startup-configuration.
<filename>	Specifies the filename of the copied file. This is only valid when copying the startup-configuration file to a specified memory location.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the primary startup-configuration file to the current running-configuration:

```
ProCurve>enable
ProCurve#copy startup-config running-config
Opening and applying file...
```

Note	<i>Any changes made to the current running configuration of the router that have not been saved to the startup configuration file (using the write command) will be lost when the copy startup-config running-config command is entered.</i>

The following example copies the primary startup-configuration file (located in flash memory) to compact flash and names the file **config_01.txt**:

```
ProCurve>enable
ProCurve#copy startup-config cflash config_01.txt
Percent Complete 100%
```

```
ProCurve#
```

copy startup-config tftp

Use the **copy startup-config tftp** command to copy the current startup-configuration file to a specified Trivial File Transfer Protocol (TFTP) server.

Syntax Description

	After entering copy startup-config tftp , the Secure Router OS prompts for the following information:
<i>Address of remote host:</i>	Specifies the IP address of the TFTP server.
<i>Destination filename:</i>	Specifies the filename to use when storing the copied file on the TFTP server. (The file will be placed in the default directory established by the TFTP server.)

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the current startup-configuration file to a TFTP server and names the file **startup_01.txt**:

ProCurve>enable

ProCurve#copy startup-config tftp

Address of remote host?**10.200.2.4**

Destination filename **startup_01.txt**

Initiating TFTP transfer...

Sent 3099 bytes.

Transfer Complete!

copy startup-config xmodem

Use the **copy startup-config xmodem** command to copy the current startup-configuration file (using the XMODEM protocol) to the terminal connected to the console port. XMODEM capability is provided in terminal emulation software such as HyperTerminal™.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example copies the current startup-configuration to the connected terminal using XMODEM protocol:

```
ProCurve>enable
```

```
ProCurve#copy startup-config xmodem
```

```
Begin the Xmodem transfer now...
```

```
Press CTRL+X twice to cancel
```

```
CCCCCC
```

The Secure Router OS is now ready to transmit the file on the **CONSOLE** port (using the XMODEM protocol). The next step in the process may differ depending on the type of terminal emulation software you are using. For HyperTerminal, you will now select **Transfer > Receive File** and select the destination. Once the transfer is complete, information similar to the following is displayed:

```
Sent 3704 bytes.
```

```
Transfer complete
```

copy tftp <destination>

Use the **copy tftp** command to copy a file located on a network Trivial File Transfer Protocol (TFTP) server to a specified destination.

Syntax Description

<i><destination></i>	Specifies the destination of the file copied from the TFTP server. Valid destinations include: cflash (installed compact flash card), flash (flash memory), startup-config (the primary configuration file), or running-config (the current running configuration file). After entering copy tftp and specifying a destination, the Secure Router OS prompts for the following information:
<i>Address of remote host:</i>	Specifies the IP address of the TFTP server.
<i>Source filename:</i>	Specifies the Name of the file to copy from the TFTP server.
<i>Destination filename:</i>	Specifies the filename to use when storing the copied file to flash memory. (Valid only for the copy tftp cflash and copy tftp flash commands.)

Default Values

No default value necessary for this command.

Usage Examples

The following example replaces the current running-configuration file with **new_config.txt** from the TFTP server (10.200.2.4):

```
ProCurve#copy tftp running-config
Address of remote host?10.200.2.4
Source filename new_config.txt
Initiating TFTP transfer...
Received 4562 bytes.
Transfer Complete!
ProCurve#
```

The following example copies the file **J03_01.biz** from the TFTP server (10.200.2.4) and saves it compact flash memory (naming the copy **SROS.BIZ**):

```
ProCurve#copy tftp cflash
Address of remote host?10.200.2.4
Source filename J03_01.biz
Destination filename SROS.BIZ
Initiating TFTP transfer...
Received 45647 bytes.
Transfer Complete!
ProCurve#
```


copy xmodem <destination>

Use the **copy xmodem** command to copy a file (using the XMODEM protocol) to a specified destination. XMODEM capability is provided in terminal emulation software such as HyperTerminal™.

Syntax Description

<i><destination></i>	Specifies the destination of the copied file.
	Valid destinations include: cflash (installed compact flash card), flash (flash memory), startup-config (the configuration file stored in flash memory), or running-config (the current running configuration file).
	After entering copy xmodem and specifying a destination, the Secure Router OS prompts for the following information:
<i>Destination filename:</i>	Specifies the filename to use when storing the copied file to flash memory. (Valid only for the copy cflash and copy flash commands.)

Default Values

No default value necessary for this command.

Usage Examples

The following example copies a software file (**J03_01.biz**) to flash memory and labels it **SROS.BIZ**:

ProCurve#copy xmodem flash

Destination filename **SROS.BIZ**

Begin the Xmodem transfer now...

Press CTRL+X twice to cancel

CCCCC

The Secure Router OS is now ready to accept the file on the **CONSOLE** port (using the XMODEM protocol). The next step in the process may differ depending on the type of terminal emulation software you are using. For HyperTerminal, you will now select **Transfer > Send File** and browse to the file you wish to copy (**J03_01.biz**). Once the transfer is complete, information similar to the following is displayed:

Received 531424 bytes.

Transfer complete.

debug aaa

Use the **debug aaa** command to activate debug messages associated with authentication from the AAA subsystem. Debug messages are displayed (real-time) on the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the SROS are disabled.

Functional Notes

The **debug aaa** events include connection notices, login attempts, and session tracking.

Usage Examples

The following is sample output for this command:

ProCurve>enable

ProCurve#debug aaa

AAA: New Session on portal 'TELNET 0 (172.22.12.60:4867)'.

AAA: No list mapped to 'TELNET 0'. Using 'default'.

AAA: Attempting authentication (username/password).

AAA: RADIUS authentication failed.

AAA: Authentication failed.

AAA: Closing Session on portal 'TELNET 0 (172.22.12.60:4867)'.

debug access-list <listname>

Use the **debug access-list** command to activate debug messages (for a specified list) associated with access list operation. Debug messages are displayed (real-time) on the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

<listname>	Specifies a configured access list

Default Values

By default, all debug messages in the SROS are disabled.

Functional Notes

The **debug access-list** command provides debug messages to aid in troubleshooting access list issues.

Usage Examples

The following example activates debug messages for the access list labeled MatchAll:

ProCurve>enable

ProCurve#debug access-list MatchAll

debug atm events

Use the **debug atm events** command to display events on all ATM ports and all virtual circuits. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates ATM event messages:

```
ProCurve>enable
```

```
ProCurve#debug atm events
```

debug atm oam <vcd> loopback [end-to-end | segment] <LLID>

Use the **debug atm oam** command to display Operation, Administration, and Maintenance (OAM) packets for an ATM virtual circuit descriptor (VCD). Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable debug messages.

Variations of this command include the following:

debug atm oam <vcd>

debug atm oam <vcd> loopback [end-to-end | segment]

debug atm oam <vcd> loopback [end-to-end | segment] <LLID>

Syntax Description

<vcd>	Shows OAM packets for a specific VCD.
loopback	Configures an OAM loopback.
end-to-end	Configures an end-to-end OAM loopback.
segment	Configures a segment loopback.
<LLID>	Specifies 16-byte OAM loopback location ID (LLID).

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates ATM OAM debug messages for VCD 1:

ProCurve>enable

ProCurve#debug atm oam 1

debug atm packet [interface atm | vc] <ATM port | VPI/VC/ > vcd <vcd number>

Use the **debug atm packet** command to activate debug messages associated with packets on ATM ports and virtual circuits. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Variations of this command include the following:

```
debug atm packet .
debug atm packet [interface atm | vc] <port id>
debug atm packet interface atm <port id> vcd <port>
```

Syntax Description

interface atm	Shows packets on a specific ATM port and on all virtual circuits.
vc	Shows packets on a specific virtual circuit.
<ATM port>	Specifies ATM port number.
<VPI/VC/ >	Specifies virtual path identifier and virtual channel identifier (VPI/VC/).
vcd	Shows packets on specific virtual circuit descriptors (VCD).
<vcd number>	Specifies a VCD port number.

Default Values

By default, all debug messages in the SROS are disabled.

Usage Examples

The following example activates debug ATM packet debug messages on ATM port 1:

```
ProCurve>enable
ProCurve#debug atm packet interface atm 1
```

debug backup

Use the **debug backup** command to activate debug messages associated with backup operation. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug backup** command activates debug messages to aid in the troubleshooting of backup links.

Usage Examples

The following example activates debug messages for backup operation:

```
ProCurve>enable
ProCurve#debug backup
```

debug bridge

Use the **debug bridge** command to display messages associated with bridge events. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates bridge debug messages:

ProCurve#debug bridge

debug crypto [ike | ike negotiation | ike client authentication | ike client configuration | ipsec | pki]

Use the **debug crypto** command to activate debug messages associated with IKE and IPSec functions. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

ike	Displays all IKE debug messages.
ike negotiation	Displays only IKE key management debug messages (e.g., handshaking).
ike client authentication	Displays IKE client authentication messages as they occur.
ike client configuration	Displays mode-config exchanges as they take place over the IKE SA. It is enabled independently from the ike negotiation debug described previously.
ipsec	Displays all IPSec debug messages.
pki	Displays all PKI (public key infrastructure) debug messages.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates the IPSec debug messages:

```
ProCurve>enable
ProCurve#debug crypto ipsec
```

debug dialup-interfaces

Use the **debug dialup-interfaces** command to generate debug messages used to aid in troubleshooting problems with all dialup interfaces such as the modem or the BRI cards. Use the **no** version of this command to disable it.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

When enabled, these messages provide status information on incoming calls, dialing and answering progress, etc. These messages also give information on why certain calls are dropped or rejected. It is beneficial to use this command when troubleshooting backup (in addition to the **debug backup** command).

Usage Examples

The following example activates the debug messages for dialup interfaces:

```
ProCurve>enable
ProCurve#debug dialup-interfaces
```

debug dynamic-dns [verbose]

Use the **debug dynamic-dns** command to display debug messages associated with dynamic DNS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

verbose	Turns on verbose messaging.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates dynamic DNS debug messages:

```
ProCurve>enable
```

```
ProCurve#debug dynamic-dns verbose
```

debug firewall

Use the **debug firewall** command to activate debug messages associated with the Secure Router OS firewall operation. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug firewall** command activates debug messages to provide real-time information about the Secure Router OS stateful inspection firewall operation.

Usage Examples

The following example activates the debug messages for the Secure Router OS stateful inspection firewall:

```
ProCurve>enable
ProCurve#debug firewall
```

debug frame-relay [events | llc2 | lmi]

Use the **debug frame-relay** command to activate debug messages associated with the Frame Relay operation. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

events	Activates debug messages for generic Frame Relay events (such as Frame Relay interface state).
llc2	Activates debug messages for the logical link control layer.
lmi	Activates debug messages for the local management interface (such as DLCI status signaling state, etc.).

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug frame-relay** command activates debug messages to aid in the troubleshooting of Frame Relay links.

Usage Examples

The following example activates all possible debug messages associated with Frame Relay operation:

```
ProCurve>enable
ProCurve#debug frame-relay events
ProCurve#debug frame-relay llc2
ProCurve#debug frame-relay lmi
```

debug frame-relay multilink *<interface>*

Use the **debug frame-relay multilink** command to activate debug messages associated with Frame Relay multilink operation. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

<i><interface></i>	Optional. Activates debug messages for the specified interface. Type debug frame-relay multilink ? for a complete list of applicable interfaces.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates debug messages associated with multilink operation for all Frame Relay interfaces:

ProCurve>enable

ProCurve#debug frame-relay multilnk

debug hdlc [errors | verbose]

Use the **debug hdlc** command to activate debug messages associated with the HDLC interface. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

errors	Enables protocol error and statistic messages.
verbose	Enables detailed debug messages.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates detailed debug messages associated with the HDLC interface:

```
ProCurve>enable
ProCurve#debug hdlc verbose
```

debug interface < *interface* >

Use the **debug interface** command to activate debug messages associated with the specified interface. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

< <i>interface</i> >	Activates debug messages for the specified interface. Type debug interface ? for a complete list of applicable interfaces.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug interface** command activates debug messages to aid in the troubleshooting of physical interfaces.

Usage Examples

The following example activates all possible debug messages associated with the Ethernet port:

```
ProCurve>enable
ProCurve#debug interface ethernet
```


debug interface adsl events

Use the **debug interface adsl events** command to activate debug messages associated with ADSL events. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates debug messages for ADSL events:

```
ProCurve>enable
```

```
ProCurve#debug interface adsl events
```

debug ip bgp [events | in | out | keepalives | updates | updates quiet]

Use the **debug ip bgp** command to activate debug messages associated with IP BGP. Debug messages are displayed (real-time) on the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

events	Displays significant BGP events such as a neighbor state change.
in/out	Displays the same information as debug ip bgp , but limits messages to the specified direction (in or out).
keepalives	Displays BGP keepalive packets.
updates	Displays BGP updates for all neighbors.
updates quiet	Displays summary information about BGP neighbor updates. (Note: updates quiet displays a one-line summary of what update displays in 104 lines.)

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

If no arguments are given, the **debug ip bgp** command displays general BGP events such as sent/received message summaries, route processing actions, and results. Keepalive packets are not debugged with this command.

Usage Examples

The following example enables debug messages on general outbound BGP messages and events:

ProCurve#debug ip bgp out

07:42:39: BGP OUT 10.15.240.1[2]: Transmitting msg, type=UPDATE (2), len=142

debug ip dhcp-client

Use the **debug ip dhcp-client** command to activate debug messages associated with DHCP client operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug ip dhcp-client** command activates debug messages to provide information on DHCP client activity in the Secure Router OS. The Secure Router OS DHCP client capability allows interfaces to dynamically obtain an IP address from a network DHCP server.

Usage Examples

The following example activates debug messages associated with DHCP client activity:

```
ProCurve>enable
```

```
ProCurve#debug ip dhcp-client
```

debug ip dhcp-server

Use the **debug ip dhcp-server** command to activate debug messages associated with DHCP server operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug ip dhcp-server** command activates debug messages to provide information on DHCP server activity in the Secure Router OS. The Secure Router OS DHCP server capability allows the Secure Router OS to dynamically assign IP addresses to hosts on the network.

Usage Examples

The following example activates debug messages associated with DHCP server activity:

```
ProCurve>enable
```

```
ProCurve#debug ip dhcp-server
```

debug ip dns-client

Use the **debug ip dns-client** command to activate debug messages associated with DNS (domain naming system) client operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug ip dns-client** command activates debug messages to provide information on DNS client activity in the Secure Router OS. The IP DNS capability allows for DNS-based host translation (name-to-address).

Usage Examples

The following example activates debug messages associated with DNS client activity:

```
ProCurve>enable
ProCurve#debug ip dns-client
```

debug ip dns-proxy

Use the **debug ip dns-proxy** command to activate debug messages associated with DNS (domain naming system) proxy operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug ip dns-proxy** command activates debug messages to provide information on DNS proxy activity in the Secure Router OS. The IP DNS capability allows for DNS-based host translation (name-to-address).

Usage Examples

The following example activates debug messages associated with DNS proxy activity:

```
ProCurve>enable
```

```
ProCurve#debug ip dns-proxy
```

debug ip icmp [send | recv]

Use the **debug ip icmp** command to show all ICMP messages as they come into the router or are originated by the router. If an optional keyword (**send** or **recv**) is not used, all results are displayed. Use the **no** form of this command to disable the debug messages.

Syntax Description

send	Optional keyword which allows you to only display ICMP messages sent by the router.
recv	Optional keyword which allows you to only display ICMP messages received by the router.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates the **debug ip icmp** send and receive messages for the Secure Router OS:

```
ProCurve>enable
```

```
ProCurve#debug ip icmp
```

```
ICMP SEND: From (0.0.0.0) to (172.22.14.229) Type=8 Code=0 Length=72 Details:echo request
```

```
ICMP RECV: From (172.22.14.229) to (10.100.23.19) Type=0 Code=0 Length=72 Details:echo reply
```

```
ICMP SEND: From (0.0.0.0) to (172.22.14.229) Type=8 Code=0 Length=72 Details:echo request
```

```
ICMP RECV: From (172.22.14.229) to (10.100.23.19) Type=0 Code=0 Length=72 Details:echo reply
```

```
ICMP RECV: From (172.22.255.200) to (10.100.23.19) Type=11 Code=0 Length=36 Details:TTL equals 0 during transit
```

```
ICMP RECV: From (172.22.14.229) to (10.100.23.19) Type=3 Code=3 Length=36 Details:port unreachable
```

```
ICMP RECV: From (172.22.14.229) to (10.100.23.19) Type=3 Code=3 Length=36 Details:port unreachable
```

debug ip igmp *<group-address>*

Use the **debug ip igmp** command to enable debug messages for IGMP transactions (including helper activity).

Syntax Description

<group-address> Optional. IP address of a multicast group.

Default Values

No default value necessary for this command.

Usage Examples

The following example enables IGMP debug messages for the specified multicast group:

```
ProCurve>enable
```

```
ProCurve#debug ip igmp 224.1.1.1
```


debug ip mrouting

Use the **debug ip mrouting** command to activate debug messages associated with multicast table routing events. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following sample activates **ip mrouting** debug messages:

```
ProCurve>enable
```

```
ProCurve#debug ip mrouting
```

debug ip ospf

Use the **debug ip ospf** command to activate debug messages associated with OSPF routing operations. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

adj	Displays OSPF adjacency events.
database-timer	Displays OSPF database timer.
events	Displays OSPF events.
flood	Displays OSPF flooding.
hello	Displays OSPF hello events.
lsa-generation	Displays OSPF link state advertisement generation.
packet	Displays OSPF packets.
retransmission	Displays OSPF retransmission events.
spf	Displays OSPF shortest-path-first calculations.
tree	Displays OSPF database tree.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following is an example of **debug ip ospf** command results:

ProCurve>enable

ProCurve#debug ip ospf flood

```

OSPF: Update LSA: id=c0a8020d rtid=192.168.2.13 area=11.0.0.0 type=1
OSPF: Update LSA: id=0b003202 rtid=11.0.50.2 area=11.0.0.0 type=1
OSPF: Queue delayed ACK lasid=0b003202 lsartid=11.0.50.2 nbr=11.0.50.2
OSPF: Rx ACK lasid=c0a8020d lsartid=192.168.2.13 nbr=11.0.50.2
OSPF: Received LSA ACK LSA_ID=-64.-88.2.13 LSA_RT_ID=-64.-88.2.13
OSPF: Rx ACK lasid=00000000 lsartid=192.168.2.13 nbr=11.0.50.2
OSPF: Received LSA ACK LSA_ID=0.0.0.0 LSA_RT_ID=-64.-88.2.13
OSPF: Sending delayed ACK
OSPF: Update LSA: id=c0a8020d rtid=192.168.2.13 area=11.0.0.0 type=1
OSPF: Flooding out last interface
OSPF: Update LSA: id=0b003202 rtid=11.0.50.2 area=11.0.0.0 type=1

```

debug ip rip [events]

Use the **debug ip rip** command to activate debug messages associated with Routing Information Protocol (RIP) operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

events	Optional. Use this optional keyword to display only RIP protocol events.

Default Values

No default value necessary for this command.

Functional Notes

The **debug ip rip** command activates debug messages to provide information on Routing Information Protocol (RIP) activity in the Secure Router OS. RIP allows hosts and routers on a network to exchange information about routes.

Usage Examples

The following example activates debug messages associated with RIP activity:

```
ProCurve>enable
ProCurve#debug ip rip
```

debug ip routing

Use the **debug ip routing** command to activate debug messages associated with routing table events. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following sample activates **ip routing** debug messages:

```
ProCurve>enable
```

```
ProCurve#debug ip routing
```

debug ip tcp [events]

Use the **debug ip tcp events** command to activate debug messages associated with significant TCP events such as state changes, retransmissions, session aborts, etc., in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Note *These debug events are logged for packets that are sent or received from the router.
Forwarded TCP packets are not included.*

Syntax Description

events	Optional. Displays only TCP protocol events.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

In the **debug ip tcp events** information, TCB stands for TCP task control block. The numbers which sometimes appear next to TCB (e.g., **TCB5** in the following example) represent the TCP session number. This allows you to differentiate debug messages for multiple TCP sessions.

Usage Examples

The following is sample output for this command:

ProCurve>enable

ProCurve#debug ip tcp events

```
2003.02.17 07:40:56 IP.TCP EVENTS TCP: Allocating block 5
2003.02.17 07:40:56 IP.TCP EVENTS TCB5: state change: FREE->SYNRCVD
2003.02.17 07:40:56 IP.TCP EVENTS TCB5: new connection from 172.22.75.246:3473 to
10.200.2.201:23
2003.02.17 07:40:56 IP.TCP EVENTS TCB5: state change: SYNRCVD->ESTABLISHED
[172.22.75.246:3473]
2003.02.17 07:41:06 IP.TCP EVENTS TCB5: Connection aborted -- error = RESET
2003.02.17 07:41:06 IP.TCP EVENTS TCB5: De-allocating tcb
```

debug ip tcp md5

Use the **debug ip tcp md5** command to activate debug messages that detail the results of each incoming TCP packet's MD5 authentication with an internal route in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

Debug messages will only be generated for TCP ports that have MD5 authentication enabled.

Usage Examples

The following example activates the display of these debug messages:

ProCurve#debug ip tcp md5

debug ip udp

Use the **debug ip udp** command to activate debug messages associated with UDP send and receive events in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Note	<i>These debug events are logged for packets that are sent or received from the router. Forwarded UDP packets are not included.</i>

Caution	<i>The overhead associated with this command takes up a large portion of your router's resources and at times can halt other router processes. It is best to only use the command during times when the network resources are in low demand (non-peak hours, weekends, etc.).</i>

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

In the **debug ip udp** information, the message **no listener** means that there is no service listening on this UDP port (i.e., the data is discarded).

Usage Examples

The following is sample output for this command:

ProCurve>enable

ProCurve#debug ip udp

```
2003.02.17 07:38:48 IP.UDP RX: src=10.200.3.236:138, dst=10.200.255.255:138, 229 bytes, no listener
2003.02.17 07:38:48 IP.UDP RX: src=10.200.2.7:138, dst=10.200.255.255:138, 227 bytes, no listener
2003.02.17 07:38:48 IP.UDP RX: src=10.200.201.240:138, dst=10.200.255.255:138, 215 bytes, no
listener
```

debug isdn events

Use the **debug isdn events** command to activate debug messages associated with ISDN events in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Functional Notes

The **debug ip rip** command activates debug messages to provide information on Routing Information Protocol (RIP) activity in the Secure Router OS. RIP allows hosts and routers on a network to exchange information about routes.

Usage Examples

The following example activates debug messages associated with ISDN activity:

```
ProCurve>enable
```

```
ProCurve#debug isdn events
```


debug lldp [rx | tx] verbose

Use the **debug lldp** command to display debug output for all LLDP receive and transmit packets.

Syntax Description

rx	Shows information about received packets.
tx	Shows information about transmitted packets.
verbose	Shows detailed debugging information.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates all possible debug messages associated with LLDP operation:

```
ProCurve#debug lldp rx
ProCurve#debug lldp tx
ProCurve#debug lldp verbose
```

debug port-auth [general | packet [both | rx | tx] | supp-sm]

Use the **debug port-auth** command to generate debug messages used to aid in troubleshooting problems during the port authentication process. Use the **no** version of this command to disable the messages.

Syntax Description

general	Optional. Displays configuration changes to the port authentication system.
packet	Optional. Displays information for packet exchange in transmit-only, receive-only or both directions.
both	Optional. Displays packet exchange information in both receive and transmit directions.
rx	Optional. Displays packet exchange information in the receive-only direction.
tx	Optional. Displays packet exchange information in the transmit-only direction.
supp-sm	Optional. Displays information pertaining to the supplicant state machine.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates port authentication debug information on received packets:

```
>enable
#debug port-auth packet rx
```

debug ppp [authentication | errors | negotiation | verbose]

Use the **debug ppp** command to activate debug messages associated with point-to-point protocol (PPP) operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

authentication	Activates debug messages pertaining to PPP authentication (CHAP, PAP, EAP, etc.).
errors	Activates debug messages that indicate a PPP error was detected (mismatch in negotiation authentication, etc.).
negotiation	Activates debug messages associated with PPP negotiation.
verbose	Activates detailed debug messages for PPP operation.

Default Values

No default value necessary for this command.

Functional Notes

The **debug ppp** command activates debug messages to provide information on PPP activity in the system. PPP debug messages can be used to aid in troubleshooting PPP links.

Usage Examples

The following example activates debug messages associated with PPP authentication activity:

ProCurve>enable

ProCurve#debug ppp authentication

debug pppoe client

Use the **debug pppoe client** command to activate debug messages associated with point-to-point protocol over Ethernet (PPPoE) operation in the Secure Router OS. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example activates debug messages associated with PPPoE activity:

```
ProCurve>enable
```

```
ProCurve#debug pppoe client
```

debug radius

Use the **debug radius** command to enable debug messages from the RADIUS subsystem. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug radius** messages show the communication process with the remote RADIUS servers.

Usage Examples

The following is an example output for the **debug radius** command:

ProCurve>enable

ProCurve#debug radius

RADIUS AUTHENTICATION: Sending packet to 172.22.48.1 (1645).

RADIUS AUTHENTICATION: Received response from 172.22.48.1.

debug sntp

Use the **debug sntp** command to enable debug messages associated with the Simple Network Time Protocol (SNTP). All SNTP Packet Exchanges and time decisions are displayed with these debugging events enabled. Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Functional Notes

The **debug sntp** command activates debug messages to aid in troubleshooting SNTP protocol issues.

Usage Examples

The following is an example output for the **debug sntp** command:

```
ProCurve>enable
```

```
ProCurve#debug sntp
```

```
ProCurve#config term
```

```
ProCurve(config)#sntp server timeserver.localdomain
```

```
2002.12.11 15:06:37 SNTP.CLIENT sent Version 1 SNTP time request to 63.97.45.57
```

```
2002.12.11 15:06:37 SNTP.CLIENT received SNTP reply packet from 63.97.45.57
```

```
2002.12.11 15:06:37 SNTP.CLIENT setting time to 12-11-2002 15:06:02
```

```
2002.12.11 15:06:37 SNTP.CLIENT waiting for 86400 seconds for the next poll interval
```

debug spanning-tree [config | events | general | root]

Use the **debug spanning-tree** command to enable the display of spanning-tree debug messages.

Syntax Description

config	Enables the display of spanning-tree debug messages when configuration changes occur.
events	Enables the display of debug messages when spanning-tree protocol events occur.
general	Enables the display of general spanning-tree debug messages.
root	Enables the display of debug messages related to the spanning-tree root.

Default Values

No default value necessary for this command.

Usage Examples

The following example enables the display of general spanning-tree debug messages:

ProCurve>enable

ProCurve#debug spanning-tree general

debug spanning-tree bpdv [receive | transmit | all]

Use the **debug spanning-tree bpdv** command to display BPDU (bridge protocol data unit) debug messages. When enabled, a debug message is displayed for each BPDU packet that is transmitted or received by the unit.

Syntax Description

receive	Displays debug messages for BPDU packets received by the unit.
transmit	Displays debug messages for BPDU packets transmitted by the unit.
all	Displays debug messages for BPDU packets that are transmitted and received by the unit.

Default Values

No default value necessary for this command.

Usage Examples

The following example displays debug messages for BPDU packets that are transmitted and received by the unit:

```
ProCurve>enable
```

```
ProCurve#debug spanning-tree bpdv all
```


debug system

Use the **debug system** command to enable debug messages associated with system events (i.e., login, logouts, etc.). Debug messages are displayed (real-time) to the terminal (or Telnet) screen. Use the **no** form of this command to disable the debug messages.

Syntax Description

No subcommands.

Default Values

By default, all debug messages in the Secure Router OS are disabled.

Usage Examples

The following example activates debug messages associated with system information:

```
ProCurve>enable
ProCurve#debug system
```

dir [*]

Use the **dir** command to display a directory list of all files on the system in flash memory or all files on the system in flash memory matching the specified pattern.

Syntax Description

*	Optional. When a wildcard is specified, only files located in flash memory matching the listed pattern are displayed.

When no wildcard is specified, the entire contents of flash memory is displayed.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from the **dir** command specifying a list of all .biz files:

```
ProCurve>enable
ProCurve#dir *.biz
4206603 J03_01.biz
4206603 SROS.biz
4039977J02_02A.biz
24208408 bytes used, 4915176 available, 29123584 total
```

dir [cflash | flash] [*]

Use the **dir** command to display a directory list of all files on the system in flash memory or on the installed compact flash card or all files on the system in flash memory or on the installed compact flash card matching the specified pattern.

Syntax Description

cflash	Specifies files located on the installed compact flash card.
flash	Specifies files located on the system in flash memory.
*	Optional. When a wildcard is specified, only files located in the specified location matching the listed pattern are displayed.

When no wildcard is specified, the entire contents of flash memory is displayed.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from the **dir** command specifying a list of all .biz files found on the installed compact flash card:

ProCurve>enable

ProCurve#dir cflash *.biz

```
4206603 HP7203A-08-00-23b-HP1-E.biz
284007 HP7203B-boot-08-01-01-HP.biz
4234845 HP7203A-08-01-01-HP-E.biz
 284238 HP7203B-boot-08-01-02-HPatp.biz
4038590 HP7203A-08-01-02-HPatp-E.biz
 285416 J01_01_02-boot.biz
4039977 J01_01_02.biz
4043024 J01_01_03.biz
2649600 ericcode.biz
24208408 bytes used, 4915176 available, 29123584 total
```

disable

Use the **disable** command to exit the Enable Command mode and enter the Basic Command mode.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example exits the Enable Command mode and enters the Basic Command mode:

ProCurve#disable

>

erase [SROS.BIZ | startup-config]

Use the **erase** command to erase the specified file.

Syntax Description

SROS.BIZ	Erases the SROS.BIZ file stored in flash memory. The SROS.BIZ file is the current software. If the autosynch feature is enabled (using the global configuration autosynch-mode command), the SROS.BIZ file is removed from both flash and compact flash.
startup-config	Erases the startup configuration file stored in flash memory. If the autosynch feature is enabled (using the autosynch command), the startup-config file is removed from both flash and compact flash.

Default Values

No default value necessary for this command.

Usage Examples

The following example erases the startup configuration file stored in flash memory:

```
ProCurve>enable
```

```
ProCurve#erase startup-config
```

If a new startup-configuration file is not specified before power-cycling the unit, the Secure Router OS will initialize using a default configuration.

erase [cflash | flash] <filename>

Use the **erase cflash** or **erase flash** command to erase the specified file from the system flash memory or an installed compact flash card.

Syntax Description

cflash	Specifies the location of the file to erase as the installed compact flash card.
flash	Specifies the location of the file to erase as the system flash memory.
<filename>	Specifies the name of the file to erase.

Default Values

No default value necessary for this command.

Usage Examples

The following example erases the file **myfile.biz** stored in flash memory:

ProCurve>enable

ProCurve#erase flash myfile.biz

erase file-system cflash

Use the **erase file-system cflash** command to erase all files on the installed compact flash card.

Note *Erasing the file system is equivalent to formatting the compact flash card.*

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example erases all files located on the installed compact flash card:

ProCurve>enable

ProCurve#erase file-system cflash

events

Use the **events** command to enable event reporting to the current CLI session. Use the **no** form of this command to disable all event reporting to the current CLI session.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Usage Examples

The following example enables event reporting:

```
ProCurve>enable
ProCurve#events
```


exception report generate

Use the **exception report generate** command to immediately generate an exception report that contains a snap shot of all current system processes.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example immediately generates an exception report:

ProCurve>enable

ProCurve#exception report generate

Exception report generated.

ProCurve#show file flash exception-report-20050726071500

Using 47428 bytes

```
=====
VERSION
=====
```

```
ProCurve Secure Router 7102dl
SROS Version: J03.01.00
  Checksum: 5D5AE64E, built on: Mon Jun 20 13:31:52 2005
Boot ROM version J03.01.00
  Checksum: B1BC, built on: Mon Jul 18 13:11:02 2005
Copyright (c) 2005-2005, Hewlett-Packard, Co.
Platform: ProCurve Secure Router 7102dl
Serial number US449TR053
Flash: 33554432 bytes  DRAM: 134217727 bytes
```

System uptime is 0 days, 0 hours, 14 minutes, 40 seconds

Current system image: "CFLASH:/SROS.BIZ"
Current configuration-file: "CFLASH:/startup-config"

Configured system image path:

Primary: "CFLASH:/SROS.BIZ"

Backup: "NONVOL:/SROS.BIZ"

Configured configuration-file path:

Primary: "CFLASH:/startup-config"

Backup: "NONVOL:/startup-config"

```
=====
CORE DUMP
=====
```

```
=====
BUFFER USERS
=====
```

Number of users: 9

Rank	User	Count
1	fixedsize	128
2	0x00873a50	128
3	0x00162530	84
4	0x00863e5c	64
5	0x0051c1e8	43
6	0x0086cfa8	16
7	0x00226cf0	14
8	0x00144990	1
9	0x0051f408	1
10	0x00000000	0
11	0x00000000	0
12	0x00000000	0
13	0x00000000	0
14	0x00000000	0
15	0x00000000	0

```
=====
EVENT HISTORY
=====
```

CurrentTime	ActiveQueue	Event
68169518	FrontPanel	0x002294b4
68169510	PacketRouting	0x005b9f68
68169500	PacketRouting	0x005b9f68
68169498	IP Events	0x00148164

----- More -----

logout

Use the **logout** command to terminate the current session and return to the login screen.

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following example shows the logout command being executed in Enable mode:

```
ProCurve>enable
ProCurve#logout
```

```
Session now available
Press RETURN to get started.
```

reload [cancel | in <delay>]

Use the **reload** command to preform a manual reload of the Secure Router OS.

Caution *Performing a **reload** disrupts data traffic.*

Syntax Description

cancel	Optional. Use the cancel keyword to deactivate a pending reload command.
in	Optional. Use the in keyword to specify a delay period the Secure Router OS will wait before reloading.
<delay>	Specifies the delay period in minutes (mmm) or hours and minutes (hh:mm).

Default Values

No default value necessary for this command.

Usage Examples

The following example reloads the Secure Router OS software in 3 hours and 27 minutes:

```
ProCurve>enable
ProCurve#reload in 03:27
```

The following example reloads the Secure Router OS software in 15 minutes:

```
ProCurve>enable
ProCurve#reload in 15
```

The following example terminates a pending reload command:

```
ProCurve>enable
ProCurve#reload cancel
```

show access-lists <listname>

Use the **show access-lists** command to display all configured access lists in the system (or a specific list).

Syntax Description

<listname>	Optional. Specify a particular access list to display.

Default Values

No default value necessary for this command.

Functional Notes

The **show access-lists** command displays all configured access lists in the system. All entries in the access list are displayed, and a counter indicating the number of packets matching the entry is listed.

Usage Examples

The following is a sample output from the **show access-lists** command:

ProCurve>enable

ProCurve#show access-lists

Standard access list MatchAll

permit host 10.3.50.6 (0 matches)

permit 10.200.5.0 wildcard bits 0.0.0.255 (0 matches)

extended access list UnTrusted

deny icmp 10.5.60.0 wildcard bits 0.0.0.255 any source-quench (0 matches)

deny tcp any (0 matches)

show arp [realtime]

Use the **show arp** command to display the Address Resolution Protocol (ARP) table.

Syntax Description

realtime	Displays full-screen output in real time. See the Functional Notes below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output of the **show arp** command:

ProCurve>enable

ProCurve#show arp

ADDRESS	TTL (min)	MAC ADDRESS	LAST UPDATED (min)	INTERFACE
192.168.30.36	13	00:E0:7D:88:1A:B9	4260	eth 0/1
192.168.30.253	17	02:60:8C:DD:0A:CE	4264	eth 0/1

show atm [pvc | traffic] interface atm <interface>

Use the **show atm** command to display information specific to the ATM interface.

Variations of this command include the following:

show atm pvc

show atm [pvc | traffic] interfaces atm <interface>

Syntax Description

pvc	Shows ATM PVC information.
traffic	Shows ATM traffic information.
<i><sub-interface number></i>	For ATM PVC information, enter the sub-interface (x.x) number.
<i><atm port interface></i>	For ATM port traffic information, enter the port ATM number 1-1023.
<i><atm vcl interface></i>	For ATM VCL traffic information, enter the ATM VCL number 1-1023.1-65536.

Default Values

No default is necessary for this command.

Usage Examples

The following is sample output from this command:

ProCurve>enable

ProCurve#show atm pvc interface atm 1.1

		Encap		Peak	Avg/Min	Burst		
Name	VPI	VCI	Type	SC	Kbps	Kbps	Cells	Status
atm 1.1	0	200	SNAP	N/A	0	0	0	Active

show autosynch-status

Use the **show autosynch-status** command to display the current autosynch configuration and the statistics for the **SROS.BIZ** and **startup-config** files (if autosynch is enabled).

Syntax Description

No subcommands.

Default Value

No default is necessary for this command.

Usage Examples

The following is a sample output from the **show autosynch-status** command (with AutoSynch™ disabled):

```
ProCurve>enable
```

```
ProCurve#show autosynch-status
```

```
AutoSynch: Mode - Disabled
```

```
AutoSynch: SROS.BIZ not synched
```

```
AutoSynch: startup-config not synched
```

The following is a sample output from the **show autosynch-status** command (with AutoSynch™ enabled):

```
ProCurve>enable
```

```
ProCurve#show autosynch-status
```

```
AutoSynch: Mode - Enabled
```

```
AutoSynch: SROS.BIZ synched
```

```
AutoSynch: startup-config synched
```


show backup interfaces

Use the **show backup interfaces** command to display all configured backup interfaces and the associated parameters for each.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example enters the Enable command mode and uses the **show** command to display backup interface information:

ProCurve>enable

ProCurve#show backup interfaces

Backup interfaces...

fr 1.16 backup interface:

Backup state: idle

Backup protocol: PPP

Call mode: originate

Auto-backup: enabled

Auto-restore: enabled

Priority: 50

Backup delay: 10 seconds

Restore delay: 10 seconds

Connect timeout: 60 seconds

Redial retries: unlimited

Redial delay: 10 seconds

Backup enabled all day on the following days:

Sunday Monday Tuesday Wednesday Thursday Friday Saturday

Backup phone number list:

Number	Call Type	min/max DS0s	Backup I/F
5551212	analog	1/1	ppp 2

show bridge *<interface>* *<slot/port>* *<bridge group #>*

Use the **show bridge** command to display a list of all configured bridge groups (including individual members of each group). Enter an interface or a bridge number to display the corresponding list.

Syntax Description

<interface> *<slot/port>* Optional. Displays all bridge groups associated with the specific interface. Type the **show bridge ?** command to display a list of applicable interfaces.

<bridgegroup#> Optional. Displays information for a specific bridge group.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show bridge** command:

ProCurve>enable

ProCurve#show bridge

Total of 300 station blocks 295 free

Address	Action	Interface	Age	Rx Count	Tx Count
00:04:51:57:4D:5A	forward	eth 0/1	0	7133392	7042770
00:04:5A:57:4F:2A	forward	eth 0/1	0	402365	311642
00:10:A4:B3:A2:72	forward	eth 0/1	4	2	0
00:12:79:00:8F:98	forward	eth 0/1	0	412367	231
00:E0:81:10:FF:CE	forward	fr 1.17	0	1502106	1486963

show buffers [realtime]

Use the **show buffers** command to display the statistics for the buffer pools on the network server.

Syntax Description

realtime	Displays full-screen output in real time. See the Functional Notes below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output from the **show buffers** command:

ProCurve>enable

ProCurve#show buffers

Buffer handles: 119 of 2000 used.

Pool	Size	Total	Used	Available	Max. Used
0	1800	1894	119	1775	122
1	2048	64	0	64	0
2	4096	32	0	32	0
3	8192	4	0	4	0
4	16384	2	0	2	0
5	32768	2	0	2	0
6	65536	2	0	2	0

show buffers users [realtime]

Use the **show buffers users** command to display a list of the top users of packet buffers. Typically, this command will only be used as a debug tool.

Syntax Description

realtime	Displays full-screen output in real time. See the Functional Notes below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample from the **show buffers users** command:

ProCurve>enable

ProCurve#show buffers users

Number of users: 7

Rank	User	Count
1	0x0052f4f8	59
2	0x0051a4fc	32
3	0x00528564	8
4	0x0053c1c8	7
5	fixedsize	5
6	0x001d8298	2
7	0x0010d970	1
8	0x00000000	0
9	0x00000000	0
10	0x00000000	0
11	0x00000000	0
12	0x00000000	0
13	0x00000000	0
14	0x00000000	0
15	0x00000000	0

show cflash *<filename>*

Use the **show cflash** command to display a list of all files currently stored in compact flash memory or details about a specified file.

Syntax Description

<i><filename></i>	Optional. Displays details for the specified file located in compact flash memory. Enter a wildcard (such as *.biz) to display the details for all files matching the entered pattern.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **show cflash** output:

```
ProCurve>enable
ProCurve#show cflash
4043024 J03_01.BIZ
285188 J03_01-boot.biz
3154 startup-config
4043024 SROS.BIZ
8374390 bytes used, 119545738 available, 127920128 total
```

show clock [detail]

Use the **show clock** command to display the system time and date entered using the **clock set** command. See *clock set* <time> <day> <month> <year> [on page 53](#) for more information.

Syntax Description

detail	Optional. Use this optional keyword to display more detailed clock information, including the time source.

Default Values

No default value necessary for this command.

Usage Examples

The following example displays the current time and data from the system clock:

```
>show clock
```

```
23:35:07 Tue Aug 20 2002
```

show configuration

Use the **show configuration** command to display a text printout of the startup configuration file stored in flash memory.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output of the **show configuration** command:

```
ProCurve>enable
ProCurve#show configuration
!
!
no enable password
!
ip subnet-zero
ip classless
ip routing
!
event-history on
no logging forwarding
logging forwarding priority-level info
no logging email
!
ip policy-timeout tcp all-ports 600
ip policy-timeout udp all-ports 60
ip policy-timeout icmp 60
!
!
interface eth 0/1
speed auto
no ip address
shutdown
!
ip access-list standard Outbound
permit host 10.3.50.6
permit 10.200.5.0 0.0.0.255
```

```
!  
ip access-list extended UnTrusted  
  deny icmp 10.5.60.0 0.0.0.255 any source-quench  
  deny tcp any any  
!  
no ip snmp agent  
!  
!  
!  
line con 0  
  no login  
!  
line telnet 0  
  login  
line telnet 1  
  login  
line telnet 2  
  login  
line telnet 3  
  login  
line telnet 4  
  login  
!
```


show connections

Use the **show connections** command to display information (including TDM group assignments) for all active connections.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from the **show connections** command:

ProCurve>enable

ProCurve#show connections

Displaying all connections....

Conn ID	From	To
1	atm 1	adsl 1/1
2	ppp 1	t1 2/1, tdm-group 1
3	ppp 1	t1 2/2, tdm-group 1
4	ppp 3	e1 3/1, tdm-group 1
5	ppp 3	e1 3/2, tdm-group 1
6	ppp 3	e1 3/3, tdm-group 1

show crypto ca [certificates | crls | profiles]

Use the **show crypto ca** command to display information regarding certificates and profiles.

Syntax Description

certificates	Displays information on all certificates.
crls	Displays a summary of all certificate revocation lists (CRLs) for each CA.
profiles	Displays information on all configured CA profiles.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample from the **show crypto ca certificates** command:

ProCurve>enable

ProCurve#show crypto ca certificates

CA Certificate

Status: Available

Certificate Serial Number: 012d

Subject Name: /C=FI/O=SSH Communications Security/OU=Web test/CN=Test CA 1

Issuer: /C=FI/O=SSH Communications Security/OU=Web test/CN=Test CA 1

CRL Dist. Pt: /C=FI/O=SSH Communications Security/OU=Web test/CN=Test CA 1

Start date is Jan 9 16:25:15 2003 GMT

End date is Dec 31 23:59:59 2003 GMT

Key Usage:

Non-Repudiation

Key Encipherment

Data Encipherment

CRL Signature

Encipherment Only

show crypto ike

Use the **show crypto ike** command to display information regarding the IKE configuration.

Variations of this command include the following:

show crypto ike client configuration pool

show crypto ike client configuration pool *<poolname>*

show crypto ike policy

show crypto ike policy *<policy priority>*

show crypto ike remote-id *<remote-id>*

show crypto ike sa

Syntax Description

client configuration pool	Displays the list of all configured IKE client configuration pools.
<i><poolname></i>	Displays detailed information regarding the specified IKE client configuration pool.
policy	Displays information on all IKE policies. Indicates if client configuration is enabled for the IKE policies and displays the pool names.
<i><policy priority></i>	Displays detailed information on the specified IKE policy. This number is assigned using the crypto ike policy command. Refer to <i>crypto ike</i> on page 260 for more information.
remote-id <i><remote-id></i>	Displays information on all IKE information regarding the remote-id. The remote-id value is specified using the crypto ike remote-id command (refer to <i>crypto ike remote-id</i> on page 264).
sa	Displays the configuration of active IKE security associations.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample from the **show crypto ike policy** command:

ProCurve>enable

ProCurve#show crypto ike policy

Crypto IKE Policy 100

 Main mode

 Using System Local ID Address

 Peers:

 63.105.15.129

 initiate main

 respond anymode

 Attributes:

 10

 Encryption: 3DES

 Hash: SHA

 Authentication: Pre-share

 Group: 1

 Lifetime: 900 seconds

show crypto ipsec

Use the **show crypto ipsec** command to display information regarding the IPsec configuration.

Variations of this command include the following:

show crypto ipsec sa

show crypto ipsec sa address *<ip address>*

show crypto ipsec sa map *<mapname>*

show crypto ipsec transform-set

show crypto ipsec transform-set *<transform-set name>*

Syntax Description

sa	Displays all IPsec security associations.
sa address <i><ip address></i>	Displays all IPsec security associations associated with the designated peer IP address.
sa map <i><mapname></i>	Displays all IPsec security associations associated with the designated crypto map name.
transform-set	Displays all defined transform sets.
<i><transform-set name></i>	Displays information for a specific transform set.

Default Values

No default value necessary for this command.

show crypto map

Use the **show crypto map** command to display information regarding crypto map settings.

Variations of this command include the following:

show crypto map

show crypto map interface ethernet *<slot/port>*

show crypto map interface frame-relay *<port number>*

show crypto map interface loopback *<port number>*

show crypto map interface ppp *<port number>*

show crypto map *<map name>*

show crypto map *<map name>* *<map number>*

Syntax Description

interface	Displays the map settings for the specified interface. Valid interfaces include: Ethernet, Frame Relay, Frame Relay sublinks, loopback, PPP, or VLAN.
<i><slot/port></i>	For Ethernet interfaces, designate the slot and port number.
<i><port number></i>	For Frame Relay, loopback, and PPP ports, enter the port number (Range: 1 to 1024). For Frame Relay sublinks, the syntax is <i><port#.sublink#></i> (Range (sublinks): 1 to 1007).
<i><map name></i>	Enter a specific crypto map name.
<i><map number></i>	Enter a specific crypto map number.

Default Values

No default value necessary for this command.

show debugging

Use the **show debugging** command to display a list of all activated debug message categories.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show debugging** command:

ProCurve>enable

ProCurve#show debugging

```
debug access-list MatchAll
debug firewall
debug ip rip
debug frame-relay events
debug frame-relay llc2
debug frame-relay lmi
```

show dialin interfaces

Use the **show dialin interfaces** command to display information regarding remote console dialin.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from the **show dialin interfaces** command:

```
ProCurve>enable
```

```
ProCurve#show dialin interfaces
```

```
Dialin interfaces...
```

```
modem 1/3 dialin interface:
```

```
  Connection Status: Connected
```

```
Caller id info: name-John Smith number-5551212 time-14:23:10 2/17/2003
```


show dynamic-dns

Use the **show dynamic-dns** command to show information related to the dynamic DNS configuration.

Syntax Description

No subcommands.

Default Values

No default is necessary for this command.

Usage Examples

The following is sample output from this command:

ProCurve#show dynamic-dns

eth 0/1:

Hostname: host

Is Updated: no

Last Registered IP: 10.15.221.33

Last Update Time: 00:00:00 Thu Jan 01 1970

show event-history

Use the **show event-history** command to display all entries in the current local event-history log.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The event history provides useful information regarding the status of the system and individual port states. Use the event history as a troubleshooting tool when identifying system issues. The following is a sample event history log.

ProCurve>enable

ProCurve#show event-history

Using 526 bytes

2002.07.12 15:34:01 T1.t1 1/1 Yellow

2002.07.12 15:34:01 INTERFACE_STATUS.t1 1/1 changed state to down.

2002.07.12 15:34:02 T1.t1 1/1 No Alarms

2002.07.12 15:34:02 INTERFACE_STATUS.t1 1/1 changed state to up.

2002.07.12 15:34:03 INTERFACE_STATUS.eth 0/1 changed state to up.

2002.07.12 15:34:10 OPERATING_SYSTEM Warm Start

2002.07.12 15:34:12 PPP.NEGOTIATION LCP up

2002.07.12 15:34:12 PPP.NEGOTIATION IPCP up

show file [cflash | flash] <filename>

Use the **show file** command to display a specified file (located in either compact flash or flash memory) to the terminal screen.

Syntax Description

cflash	Specifies a file located in compact flash memory.
flash	Specifies a file located in flash memory.
<filename>	Specify the filename of the file located in the specified memory location. Wildcard entries (such as *.biz) are not valid for the show file command.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **show cflash** output:

```
ProCurve>enable
ProCurve#show file cflash startup-config
Using 2354 bytes
!
!
hostname "HP_West"
enable password password
!
ip subnet-zero
ip classless
ip routing
!
event-history on
no logging forwarding
no logging email
logging email priority-level info
!
username "admin" password "password"
!
!
no ip firewall alg h323
ip firewall alg sip
--MORE-
```

show flash <filename>

Use the **show flash** command to display a list of all files currently stored in flash memory or details about a specified file.

Syntax Description

<filename>	Optional. Displays details for a specified file located in flash memory. Enter a wildcard (such as *.biz) to display the details for all files matching the entered pattern.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **show flash** output:

ProCurve>enable

ProCurve#show flash

Files:

245669 030100boot.biz

1141553 new.biz

821 startup-config

1638 startup-config.old

1175679 SROS.biz

821 startup-config.bak

2572304 bytes used 4129776 available 6702080 total

show frame-relay

Use the **show frame-relay** command to display configuration and status parameters for configured virtual Frame Relay interfaces.

Variations of this command include the following:

show frame-relay lmi

show frame-relay pvc

show frame-relay pvc interface frame-relay *<interface>*

show frame-relay pvc interface frame-relay *<interface>* **realtime**

show frame-relay pvc realtime

Syntax Description

lmi	Displays Link Management Interface (LMI) statistics for each virtual Frame Relay interface.
pvc	Displays Permanent Virtual Circuit (PVC) configuration and statistics for all virtual Frame Relay interfaces (or a specified interface).
interface	Displays configuration and statistics for a specified Frame Relay interface.
frame-relay	Optional. Displays Frame Relay PVC statistics for a specific Frame Relay interface.
<i><interface></i>	Specifies the virtual Frame Relay interface (for example fr 1).
realtime	Displays full-screen output in realtime. See the Functional Notes below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following are sample outputs from various **show frame-relay** commands:

ProCurve>enable

ProCurve#show frame-relay lmi

LMI statistics for interface FR 1 LMI TYPE = ANSI
 Num Status Enq. Sent 79 Num Status Msgs Rcvd 71
 Num Update Status Rcvd 12 Num Status Timeouts 5

ProCurve>enable

ProCurve#show frame-relay pvc

Frame Relay Virtual Circuit Statistics for interface FR 1

	Active	Inactive	Deleted	Static
local	2	0	0	2

DLCI = 16 DLCI USAGE = LOCAL PVC STATUS = ACTIVE INTERFACE = FR 1.16
 MTU: 1500

input pkts: 355	output pkts: 529	in bytes: 23013
out bytes: 115399	dropped pkts: 13	in FECN pkts: 0
in BECN pkts: 0	in DE pkts: 0	out DE pkts: 0
pvc create time: 00:00:00:12		last time pvc status changed: 00:00:13:18

DLCI = 20 DLCI USAGE = LOCAL PVC STATUS = ACTIVE INTERFACE = FR 1.20
 MTU: 1500

input pkts: 0	output pkts: 44	in bytes: 0
out bytes: 22384	dropped pkts: 11	in FECN pkts: 0
in BECN pkts: 0	in DE pkts: 0	out DE pkts: 0
pvc create time: 00:00:01:25		last time pvc status changed: 00:00:13:18

show frame-relay fragment [interface frame-relay <port.sublink>]

Use the **show frame-relay fragment** command to display FRF.12 statistics for Frame Relay sublinks enabling FRF.12 fragmentation.

Syntax Description

interface frame-relay <port.sublink>	Optional. Displays detailed FRF.12 statistics for the specified Frame Relay sublink (if FRF.12 is enabled on that sublink).

Default Values

No default value necessary for this command.

Usage Examples

The following are sample outputs from various **show frame-relay fragment** commands:

ProCurve>enable

ProCurve#show frame-relay fragment

interface	dlci	frag_size	rx_frag	tx_frag	dropped_frag
fr 1.1	17	100	46	48	0
fr 1.2	18	200	42	21	0

ProCurve>enable

ProCurve#show frame-relay fragment frame-relay 1.1

DLCI = 17 FRAGMENT SIZE = 100

rx frag. pkts	46	tx frag. pkts	48
rx frag. bytes	4598	tx frag. bytes	4724
rx non-frag. pkts	18	tx non-frag. pkts	28
rx non-frag. bytes	1228	tx non-frag. bytes	1960
rx assembled pkts	23	tx pre-fragment pkts	34
rx assembled bytes	5478	tx pre-fragment bytes	6324
dropped reassembling pkts	0	dropped fragmenting pkts	0
rx out-of-sequence fragments	0		
rx unexpected beginning fragment	0		

show frame-relay multilink <interface> detailed

Use the **show frame-relay multilink** command to display information associated with the Frame Relay multilink interface.

Syntax Description

<interface>	Optional. Specifies the display of information for a specific interface. Enter the show frame-relay multilink ? command for a complete list of interfaces.
detailed	Optional. Use this optional keyword to display more detailed information.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from this command:

```
ProCurve>enable
ProCurve#show frame-relay multilink
Bundle: frame-relay 1 is DOWN; class A bundle
Near-end BID: MFR1; Far-end BID: unknown
```


show hosts [verbose]

Use the **show hosts** command to display information such as the domain name, name lookup service, a list of name server hosts, and the cached list of host names and addresses on the network to which you can connect.

Syntax Description

verbose	Enables detailed messaging.

Default Values

No default value necessary for this command.

Functional Notes

The list below describes the fields contained in the host table:

- **Flags:** Indicate whether the entry is permanent (P) or temporary (T) and if the entry is OK or expired (EXP).
- **Age:** Indicates the age of the entry.
- **Type:** Shows the protocol type.
- **Address:** Displays the IP address for the entry.

Usage Examples

The following example is sample output from the **show hosts** command:

ProCurve>enable

ProCurve#show hosts

Name/address lookup uses domain name service

DNS Proxy is disabled

Default domain is not set

Name servers are 1.1.1.1 2.2.2.2

Host	Flags	Age	Type	Address
Example1	(P OK)	--	IP	1.1.1.1
Example2	(P OK)	--	IP	2.2.2.2

show interfaces <interface>

Use the **show interfaces** command to display configuration parameters and current statistics for all interfaces (or a specified interface).

Syntax Description

<interface>	Optional. Specific interface to display. Type show interfaces ? for a complete list of valid interfaces.
performance-statistics	Optional. Displays the current 15-minute interval, the current 24-hour totals, and all 96 stored intervals.
performance-statistics total-24-hour	Optional. Displays the current 24-hour totals and the past seven 24-hour intervals.
performance-statistics <x-y>	Shows the current 15-minute interval, the current 24-hour totals, and all intervals from x through y. This command is basically the same thing as the performance-statistics command with the added function of allowing you to specify a particular interval (or range of intervals) to display rather than displaying all 96. <i>Note: If you want to display the 24th interval, enter (for example) show interface t1/1 performance-statistics 24-24. Entering show interface t1 1/1 performance-statistics 24 results in displaying the 24-hour statistics. Any number other than 24 (between 1 and 96) results in the correct display of the selected interval (e.g., show interface t1 1/1 performance-statistics 4 shows the fourth interval).</i>
realtime	Displays full-screen output in real time. See the <i>Functional Notes</i> below for more information.
version	Optional. Displays current version information (e.g., model and list number, software version, etc.) for the T1 interface.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following are samples from various **show interfaces** commands:

ProCurve>enable

ProCurve#show interfaces t1 1/1

t1 1/1 is UP

T1 coding is B8ZS framing is ESF

Clock source is line FDL type is ANSI

Line build-out is 0dB

No remote loopbacks No network loopbacks

DS0 Status: 123456789012345678901234

NNNNNNNNNNNNNNNNNNNNNNNNNNNN

Line Status: -- No Alarms --

Current Performance Statistics:

0 Errored Seconds 0 Bursty Errored Seconds

0 Severely Errored Seconds 0 Severely Errored Frame Seconds

0 Unavailable Seconds 0 Path Code Violations

0 Line Code Violations 0 Controlled Slip Seconds

0 Line Errored Seconds 0 Degraded Minutes

ProCurve#show interfaces modem 1/2

modem 1/2 is UP

Line status: on-hook

Caller ID will be used to route incoming calls

0 packets input 0 bytes 0 no buffer

0 runts 0 giants 0 throttles

0 input errors 0 CRC 0 frame

0 abort 0 ignored 0 overruns

0 packets output 0 bytes 0 underruns

0 input clock glitches 0 output clock glitches

0 carrier lost 0 cts lost

ProCurve#show interfaces eth 0/1

Ip address is 10.200.1.50

Netmask is 255.255.0.0

MTU is 1500

Fastcaching is Enabled

RIP Authentication is Disabled

RIP Tx uses global version value

RIP Rx uses global version value

ProCurve#show interfaces fr 1

TDM group 10 line protocol is UP

Encapsulation FRAME-RELAY (fr 1)

463 packets input 25488 bytes 0 no buffer

0 runts 0 giants 0 throttles

0 input errors 0 CRC 0 frame
0 abort 0 ignored 0 overruns
864 packets output 239993 bytes 0 underruns
0 input clock glitches 0 output clock glitches
0 carrier lost 0 cts lost

Line Status: -- No Alarms --

Current Performance Statistics:

0 Errored Seconds 0 Bursty Errored Seconds
0 Severely Errored Seconds 0 Severely Errored Frame Seconds
0 Unavailable Seconds 0 Path Code Violations
0 Line Code Violations 0 Controlled Slip Seconds
0 Line Errored Seconds 0 Degraded Minutes

ProCurve#show interfaces fr 1.100*

fr 1.100 is Active

Ip address is 63.97.45.57, mask is 255.255.255.248

Interface-dlci is 100

MTU is 1500 bytes, BW is 96000 Kbit (limited)

Average utilization is 53%

*Note: If the user has configured a **Bc** and **Be** value on the virtual circuit, the bandwidth (**BW**) displayed is the sum of those values (Bc + Be). If not, the value for **BW** is the speed of the interface. The **Average utilization** displayed is the average utilization of the displayed bandwidth. If the bandwidth number is the Bc + Be value, the **(limited)** text appears (as shown above).

show interfaces adsl <slot/port> information [atuc | atur | bit-allocation]

Use the **show interfaces adsl** command to display information related to the ADSL port.

Syntax Description

<slot/port>	Enter interface slot and port number.
atuc	Show ADSL interface remote information.
atur	Show ADSL local information.
bit-allocation	Show ADSL DMT bit-allocation table.

Default Values

No default is necessary for this command.

Usage Examples

The following example shows sample output for this command:

ProCurve#show interfaces adsl 0/1 information

adsl 0/1 line information

adsl 0/1 Local Line Information

Vendor Id:

Serial Number:

Firmware Version:

ADSL Capabilities G.DMT, G.LITE, ADSL2, ADSL2+

adsl 0/1 Remote Line Information

Vendor Id: 00000000

Serial Number: 00000000

Firmware Version: 0

ADSL Capabilities G.DMT, G.LITE, ADSL2, ADSL2+

show ip access-lists *<listname>*

Use the **show ip access-lists** command to display all configured IP access lists in the system.

Syntax Description

<i><listname></i>	Optional. Specify a particular access list to display.

Default Values

No default value necessary for this command.

Functional Notes

The **show ip access-lists** command displays all configured IP access lists in the system. All entries in the access list are displayed, and a counter indicating the number of packets matching the entry is listed.

Usage Examples

The following is a sample output from the **show ip access-lists** command:

ProCurve>enable

ProCurve#show ip access-lists

Standard IP access list MatchAll

 permit host 10.3.50.6 (0 matches)

 permit 10.200.5.0 wildcard bits 0.0.0.255 (0 matches)

Extended IP access list UnTrusted

 deny icmp 10.5.60.0 wildcard bits 0.0.0.255 any source-quench (0 matches)

 deny tcp any any (0 matches)

show ip arp [realtime]

Use the **show ip arp** command to display the Address Resolution Protocol (ARP) table.

Syntax Description

realtime	Displays full-screen output in real time. See the <i>Functional Notes</i> below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output of the **show ip arp** command:

ProCurve>enable

ProCurve#show ip arp

ADDRESS	TTL (min)	MAC ADDRESS	LAST UPDATED (min)
192.168.30.36	13	00:E0:7D:88:1A:B9	4260
192.168.30.253	17	02:60:8C:DD:0A:CE	4264

show ip as-path-list [<listname>]

Use the **show ip as-path-list** command to display any AS path lists that have been configured in the router, along with any permit and deny clauses in each list.

Syntax Description

<listname>	Optional. Specifies that the command display only the list matching the specified AS path listname. If not specified, all AS path lists are displayed.

Default Values

By default, this command displays all AS path lists.

Usage Examples

In the following example, all AS path lists defined in the router are displayed.

ProCurve>enable

ProCurve#show ip as-path-list

ip as-path-list AsPathList1:

 permit 100

 permit 200

 permit 300

 deny 6500

ip as-path-list AsPathList2:

 permit 400

 permit 500

In the following example, only the AS Path List with the name **AsPathList2** is displayed.

ProCurve>enable

ProCurve#show ip as-path-list AsPathList2

ip as-path-list AsPathList2:

 permit 400

 permit 500

show ip bgp community [*<community number>* . . . *<community number>* | internet | no-export | local-as | no-advertise] [exact]

Use the **show ip bgp community** command to display only those routes learned via BGP that match the community numbers specified in the command. If no communities are specified, all BGP routes are shown.

Syntax Description

<i><community-number></i>	Optional. Displays routes that contain this value in their community attribute. This is a numeric value that can be an integer from 1 to 4,294,967,295 or string in the form "aa:nn", where the value of "aa" is the AS number and the value of "nn" is the community number. Multiple community-number parameters can be present in the command.
internet	Optional. Displays routes that contain this value in their community attribute. This represents the well-known reserved community number for the INTERNET community.
local-as	Optional. Displays routes that contain this value in their community attribute. This represents the well-known reserved community number for NO_EXPORT_SUBCONFED. Routes containing this attribute should not be advertised to external BGP peers.
no-export	Optional. Displays routes containing this value in the community attribute. This represents the well-known reserved community number for NO_EXPORT. Routes containing this attribute should not be advertised to BGP peers outside a confederation boundary.
no-advertise	Optional. Displays routes containing this value in the community attribute. This represents the well-known reserved community number for NO_ADVERTISE. Routes containing this attribute should not be advertised to any BGP peer.
exact	Optional. Displays BGP routes with the community numbers specified and <i>only</i> those specified.

Default Values

By default, this command displays all BGP routes.

Usage Examples

In the following example, all BGP routes are displayed whose community numbers match those listed in the **show ip bgp community** command.

ProCurve>enable

ProCurve#show ip bgp community local-as 10:405

BGP local router ID is 10.22.131.241, local AS is 302.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Path
10.22.152.20/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.24/29	10.22.131.10	304		302 300 1 3 4 5 i
10.22.152.36/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.52/30	10.22.131.10	304		302 300 1 3 4 i
11.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
12.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
13.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
14.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
20.0.0.0/30	10.22.131.10	304		302 300 1 3 4 5 i
21.0.0.0/30	10.22.131.10	304		302 300 1 3 4 5 i

Total RIB entries = 10

Information displayed includes: the ID of this router and its Autonomous System (AS) number; the destination Network address of the route learned; the Next Hop address to that network; the Metric; the Local Preference value (set using the **set local-preference** command); and the AS Path to the destination network.

The following is a sample output for the show-ip bgp community command with an exact match specified: BGP routes with the community numbers specified and *only* those specified are shown

ProCurve>enable

ProCurve#show ip bgp community 1001 2001 3001 exact

BGP local router ID is 192.168.9.1, local AS is 252.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	NextHop	Metric	LocPrf	Path
* 192.168.11.0/24	10.22.27.251			249 251 i
* 192.168.12.0/24	10.22.27.251			249 251 i
*> 192.168.32.0/24	10.22.27.249			249 i
*> 192.168.33.0/24	10.22.27.249			249 i

Total RIB entries = 4

show ip bgp community-list <community-list-name> [exact]

Use the **show ip bgp community-list** command to display BGP routes that are permitted by the specified community list.

Syntax Description

<community-list-name> Specifies the name of the community list whose routes you wish to see.

exact Optional. Specifying this option restricts the routes displayed to only those whose community lists exactly match those specified in the named community list. If this parameter is omitted, all routes matching any part of the specified community list will be displayed.

Default Values

No default value necessary for this command.

Usage Examples

In the following example, all BGP routes are displayed whose community numbers match those defined in the community list named CList1.

ProCurve>enable

ProCurve#show ip bgp community-list CList1

BGP local router ID is 10.22.131.241, local AS is 302.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Path
10.22.152.20/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.24/29	10.22.131.10	304		302 300 1 3 4 5 i
10.22.152.36/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.52/30	10.22.131.10	304		302 300 1 3 4 i
11.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
12.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
13.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
14.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
20.0.0.0/30	10.22.131.10	304		302 300 1 3 4 5 i
21.0.0.0/30	10.22.131.10	304		302 300 1 3 4 5 i

Total RIB entries = 10

Information displayed includes: the ID of this router and its Autonomous System number, the destination Network address of the route learned, the Next Hop address to that network, the Metric, the Local Preference value (set using the "set local-preference *" command), and the Autonomous System Path to the destination network.

show ip bgp regexp <expression>

Use the **show ip bgp regexp** command to display a summary of the BGP route table that includes routes whose autonomous system (AS) path matches the specified expression.

Syntax Description

<expression>	Displays routes whose autonomous system (AS) path matches the regular expression specified.

Default Values

No default value necessary for this command.

Functional Notes

Entries that are not filtered by prefix lists are marked with an asterisk (*) to show they are valid. Entries that are deemed the best path to advertised route are marked with a caret (>).

Usage Examples

The following sample output of the **show ip bgp regexp _303_** command shows all of the entries in the BGP database that contain "303" in the AS path.

ProCurve#show ip bgp regexp _303_

BGP local router ID is 192.168.3.1, local AS is 304.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

	Network	NextHop	Metric	LocPrf	Path
	10.22.130.8/29	10.22.132.9			303 304 302 i
*	i10.22.130.240/28	0.22.132.1		100	303 300 i
*	10.22.130.240/28	10.22.132.9			303 300 i
	10.22.131.0/29	10.22.132.9			303 304 302 i
	10.22.131.8/29	10.22.132.9			303 304 302 i
*	i10.22.131.16/29	10.22.132.1	0	100	303 i
*	10.22.131.16/29	10.22.132.9	0		303 i
*	i10.22.131.240/28	10.22.132.1		100	303 300 i
*	10.22.131.240/28	10.22.132.9			303 300 i
*	10.22.132.0/29	10.22.131.1	0		302 303 i
*	10.22.132.0/29	10.22.131.9	0		302 303 i
*	i10.22.132.0/29	10.22.132.1	0	100	303 i
*>	10.22.132.0/29	10.22.132.9	0		303 i
*	10.22.132.8/29	10.22.131.1	0		302 303 i
*	10.22.132.8/29	10.22.131.9	0		302 303 i

```

* 10.22.132.8/29    10.22.132.9    0          303 i
* i10.22.132.240/28 10.22.132.1    0          100    303 i
*> 10.22.132.240/28 10.22.132.9    0          303 i
    10.22.134.0/29    10.22.132.9          303 304 i
    10.22.134.8/29    10.22.132.9          303 304 i
    10.22.134.16/29   10.22.132.9          303 304 i
    10.22.134.24/29   10.22.132.9          303 304 i
    10.22.134.32/29   10.22.132.9          303 304 i
    10.22.134.40/29   10.22.132.9          303 304 i
    10.22.134.48/29   10.22.132.9          303 304 i
    10.22.134.56/29   10.22.132.9          303 304 i
    10.22.134.64/29   10.22.132.9          303 304 i
    10.22.134.80/29   10.22.132.9          303 304 i
    10.22.135.0/29    10.22.132.9          303 304 305 i
    10.22.135.8/29    10.22.132.9          303 304 305 i

```

Total RIB entries = 30

show ip bgp summary

Use the **show ip bgp summary** command to display a summary of the BGP route table.

Syntax Description

summary	Displays a summary of the status for all BGP.

Default Values

No default value necessary for this command.

Functional Notes

Entries that are not filtered by prefix lists are marked with an asterisk (*) to show they are valid. Entries that are deemed the best path to advertised route are marked with a caret (>).

Usage Examples

The following sample output of the **show ip bgp summary** command shows a summarized list of the configured BGP neighbors as well as their status and statistics.

ProCurve#show ip bgp summary

BGP router identifier 192.168.3.1, local AS number 304

8 network entries, 5 paths, and 23 BGP path attribute entries

Neighbor	V	AS	MsgRcvd	MsgSent	InQ	OutQ	Up/Down	State/PfxRcd
10.22.131.1	4	302	95	104	0	0	01:30:06	9
10.22.131.9	4	302	97	105	0	0	01:30:07	21
10.22.132.9	4	303	200	179	0	0	02:43:09	21
10.22.134.1	4	304	166	178	0	0	02:43:15	3
10.22.134.10	4	304	174	179	0	0	02:43:24	7
10.22.134.26	4	304	172	174	0	0	02:41:43	10
10.22.134.34	4	304	164	174	0	0	02:41:40	4

show ip bgp neighbors

Use the **show ip bgp neighbors** command to display information for the specified neighbor. Variations of this command include the following:

show ip bgp neighbors

show ip bgp neighbors *<ip address>*

show ip bgp neighbors *<ip address>* **advertised-routes**

show ip bgp neighbors *<ip address>* **received-routes**

show ip bgp neighbors *<ip address>* **routes**

Syntax Description

<i><ip address></i>	Displays information for the specified neighbor. If no IP address is entered, information for all neighbors is displayed.
advertised-routes	Displays all routes being advertised to the specified neighbor. Command output is the same as for show ip bgp except filtered to only the BGP routes being advertised to the specified neighbor.
received-routes	Displays all routes (accepted and rejected) advertised by the specified neighbor. Routes may be rejected by inbound filters such as prefix list filters.
routes	Displays all accepted received routes advertised by the specified neighbor. Routes displayed have passed inbound filtering. This command output is the same as show ip bgp except the output is filtered to those learned from the specified neighbor.

Default Values

No default value necessary for this command.

Functional Notes

Entries that are not filtered by prefix lists are marked with an asterisk (*) to show they are valid. Entries that are deemed the best path to advertised route are marked with a caret (>).

Usage Examples

The following are output variations of the **show ip bgp neighbors** command:

ProCurve>enable

ProCurve#show ip bgp neighbors

BGP neighbor is 10.15.43.17, remote AS 100, external link
Configured hold time is 180, keepalive interval is 60 seconds
Default minimum time between advertisement runs is 30 seconds
Connections established 6; dropped 5
Last reset: Interface went down
Connection ID: 15
BGP version 4, remote router ID 8.1.1.1
BGP state is Established, for 01:55:05
Negotiated hold time is 180, keepalive interval is 60 seconds
Message statistics:
InQ depth is 0, OutQ depth is 0
Local host: 10.15.43.18, Local port: 179

	Sent	Rcvd
Opens:	1	1
Notifications:	0	0
Updates:	0	8
Keepalives:	116	116
Unknown:	0	0
Total:	117	125

Foreign host: 10.15.43.17, foreign port: 1048

Flags: passive open

ProCurve#show ip bgp neighbors 10.15.43.34 advertised-routes

BGP local router ID is 10.0.0.1, local AS is 101.
Status codes: * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

Network	NextHop	Metric Path
*> 1.0.0.0/8	10.15.43.17	1 100 i
*> 2.0.0.0/9	10.15.43.17	1 100 i

ProCurve#show ip bgp neighbors 10.15.43.17 received-routes

BGP local router ID is 10.0.0.1, local AS is 101.

Status codes: * valid, > best, i - internal

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	NextHop	Metric Path
*> 1.0.0.0/8	10.15.43.17	1 100 i
*> 2.0.0.0/9	10.15.43.17	1 100 i

ProCurve#show ip bgp neighbors 10.15.43.17 routes

BGP local router ID is 10.0.0.1, local AS is 101.

Status codes: * valid, > best, i - internal

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	NextHop	Metric Path
*> 1.0.0.0/8	10.15.43.17	1 100 i
*> 2.0.0.0/9	10.15.43.17	1 100

show ip cache

Use the **show ip cache** command to display the fast cache table.

Syntax Description

No subcommands.

Default Values

No default necessary for this command.

Usage Examples

The following example shows sample output from the **show ip cache** command:

ProCurve>enable

ProCurve#show ip cache

DESTINATION	INTERFACE	NEXT HOP	USE COUNT	MAC ADDRESS
10.17.6.52	Loopback	127.0.0.1	231	
172.22.77.80	eth 0/1	10.17.254.254	0	00:12:79:11:BA:32
10.17.255.255	Loopback	127.0.0.1	16	

show ip community-list [<community-list-name>]

Use the **show ip community-list** command to display any or all defined community lists in the router configuration.

Syntax Description

<community-list-name> Optional. Specifies the name of the community list you wish to display. If this parameter is omitted, all defined community lists will be displayed.

Default Values

No default value necessary for this command.

Usage Examples

The following example shows two community lists, one of which permits all routes containing community number 10:67, and another which permits routes containing community number 10:68 and the internet community number, but denies routes containing community number 10:45.

ProCurve#show ip community-list

```
ip community-list CommList1:
  permit 10:67
ip community-list CommList2:
  permit 10:68 internet
  deny 10:45
```

show ip dhcp-client lease *<interface>*

Use the **show ip dhcp-client lease** command to display all Dynamic Host Client Protocol (DHCP) lease information for interfaces that have dynamically assigned IP addresses.

Syntax Description

<interface>	Optional. Displays the information for the specified interface. Type show ip dhcp-client lease ? for a complete list of applicable interfaces.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show dhcp-client lease** command:

ProCurve>enable

ProCurve#show dhcp-client lease

Interface: ethernet 0/1

Temp IP address: 10.100.23.64 Mask: 0.0.0.0

DHCP Lease server: 10.100.23.207 State: Bound (3)

Lease: 120 seconds

Temp default gateway address: 0.0.0.0

Client-ID: N/A

show ip dhcp-server binding *<client ip address>*

Use the **show ip dhcp-server binding** command to display the Dynamic Host Client Protocol (DHCP) server client table with associated information.

Syntax Description

<client ip address> Optional. Specifies a particular client IP address.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show ip dhcp-server binding** command:

ProCurve>enable

ProCurve#show ip dhcp-server binding

IP Address	Client Id	Lease Expiration	Client Name
10.100.23.64	01:00:12:79:00:8f:b3	Aug 15 2002 11:02 AM	Router

show ip igmp groups <group-address>

Use the **show ip igmp groups** command to display the multicast groups that have been registered by directly connected receivers using IGMP. If no group address is specified, all groups are shown with this command.

Syntax Description

<group-address>	Optional. Displays IP address of a multicast group.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from this command:

ProCurve>enable

ProCurve#show ip igmp groups

IGMP Connected Group Membership

Group Address	Interface	Uptime	Expires	Last Reporter
172.0.1.50	Loopback100	00:42:57	00:02:50	172.23.23.1
172.1.1.1	Ethernet0/1	00:05:26	00:02:51	1.1.1.2
172.1.1.1	Loopback100	00:42:57	00:02:51	172.23.23.1

show ip igmp interface <interface>

Use the **show ip igmp interface** command to display multicast-related information per-interface. If no interface is specified, this command shows information for all interfaces.

Syntax Description

<interface>	Optional. Designates the display of information for a specific interface (in the format type slot/port). Enter the show ip igmp interface ? command for a complete list of interfaces.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from this command:

ProCurve>enable

ProCurve#show ip igmp interface

eth 0/1 is UP

Ip Address is 10.22.120.47, netmask is 255.255.255.0

IGMP is enabled on interface

Current IGMP version is 2

IGMP query interval is 60 seconds

IGMP querier timeout is 120 seconds

IGMP max query response time is 10 seconds

Last member query count is 2

Last member query response interval is 1000 ms

IGMP activity: 548 joins, 0 leaves

IGMP querying router is 0.0.0.0

IGMP helper address is disabled

show ip interfaces [<interface> | brief]

Use the **show ip interfaces** command to display the status information for all IP interfaces (or a specific interface).

Note *To view secondary IP addresses, use the **show running-config** command.*

Syntax Description

<interface>	Optional. Displays status information for a specific interface. If no interface is entered, status information for all interfaces is displayed. Type show ip interfaces ? for a complete list of applicable interfaces.
brief	Use this optional keyword to display an abbreviated version of interface statistics for all IP interfaces.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output of the **show ip interfaces** command:

ProCurve>enable

ProCurve#show ip interfaces

```
eth 0/1 is UP, line protocol is UP
Ip address is 10.10.10.1
Netmask is 255.255.255.0
MTU is 1500
Fastcaching is Enabled
RIP Authentication is Disabled
RIP Tx uses global version value
RIP Rx uses global version value
```


show ip mroute [<group-address> | <interface>] [summary]

Use the **show ip mroute** command to display IP multicasting routing table information.

Syntax Description

<group-address>	Optional. Displays IP address of a multicast group.
<interface>	Optional. Designates the display of parameters for a specific interface (in the format type slot/port). For example: eth 0/1.
summary	Optional. Displays a single-line summary for each entry in the IP multicast routing table.

Default Values

No default value necessary for this command.

Usage Examples

The following is sample output from this command:

ProCurve>enable

ProCurve#show ip mroute

IP Multicast Routing Table

Timers: Uptime/Expires

```
(*, 239.2.170.3), 01:03:19/00:00:00
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    fr 1.20, Forward, 01:03:19/00:01:48
(*, 224.1.1.1), 00:00:01/00:02:58, RP 0.0.0.0, flags: DCL
Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list: <-- because lo 100 ifc is joined
    Loopback100, Forward/Dense, 00:00:01/00:00:00

(*, 17.0.1.50), 00:00:01/00:02:58, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Loopback100, Forward/Dense, 00:00:01/00:00:00
```

show ip ospf

Use the **show ip ospf** command to display general information regarding OSPF processes.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show ip ospf** command:

ProCurve>enable

ProCurve#show ip ospf

```
Summary of OSPF Process with ID: 192.2.72.101
Supports only single Type Of Service routes (TOS 0)
SPF delay timer: 5 seconds, Hold time between SPF's: 10 seconds
LSA interval: 240 seconds
Number of external LSAs: 0, Checksum Sum: 0x0
Number of areas: 0, normal: 0, stub: 0, NSSA: 0
```

show ip ospf database

Use the **show ip ospf database** command to display information from the OSPF database regarding a specific router. There are several variations of this command which you can use to obtain information about different OSPF link state advertisements. The variations are shown below:

show ip ospf <area-id> database

show ip ospf <area-id> database adv-router <ip address>

show ip ospf <area-id> database database-summary

show ip ospf <area-id> database external <link-state-id>

show ip ospf <area-id> database external <link-state-id> adv-router <ip address>

show ip ospf <area-id> database network <link-state-id>

show ip ospf <area-id> database network <link-state-id> adv-router <ip address>

show ip ospf <area-id> database router <link-state-id>

show ip ospf <area-id> database router <link-state-id> adv-router <ip address>

show ip ospf <area-id> database summary <link-state-id>

show ip ospf <area-id> database summary <link-state-id> adv-router <ip address>

Syntax Description

<area id>	Optional. Displays area ID number associated with the OSPF address range. This range is defined in the network router configuration command used to define the particular area. See <i>network <ip address> <wildcard> area <area id></i> on page 977 for more information.
<link-state-id>	Optional. Identifies the portion of the Internet environment that is being described by the advertisement. The value needed in this field is tied to the advertisement's LS type.
<ip address>	Enter in the form <A.B.C.D>.

Default Values

No default value necessary for this command.

Functional Notes

The link state ID differs depending on whether the link state advertisement in question describes a network or a router.

If describing a network, this ID is one of the following:

- The network's IP address. This is true for type 3 summary link advertisements and in autonomous system external link advertisements.
- An address obtained from the link state ID. If the network link advertisement's link state ID is masked with the network's subnet mask, this will yield the network's IP address.

If describing a router, this ID is always the router's OSPF router ID.

Usage Examples

ProCurve>enable

ProCurve#show ip ospf database

OSPF router with ID: 0.0.0.0

show ip ospf interface *<interface type>* *<interface number>*

Use the **show ip ospf interface** command to display OSPF information for a specific interface.

Syntax Description

<i><interface type></i>	Optional. Specifies the interface type. Type show ip ospf interface ? for a complete list of applicable interfaces.
<i><interface number></i>	Optional. Specifies the interface number.

Default Values

No default value necessary for this command.

Usage Examples

The following example shows OSPF information for the ppp 1 interface.

ProCurve>enable

ProCurve#show ip ospf interface ppp 1

show ip ospf neighbor *<interface type>* *<interface number>* *<neighbor id>*
[detail]

Use the **show ip ospf neighbor** command to display OSPF neighbor information for a specific interface.

Syntax Description

<i><interface type></i>	Optional. Specifies the interface type (i.e., eth , ppp , etc.).
<i><interface number></i>	Optional. Specifies the interface number.
<i><neighbor id></i>	Optional. Specifies a specific neighbor's router ID.
detail	Optional. Enter this keyword to display details on all neighbors.

Default Values

No default value necessary for this command.

Usage Examples

ProCurve>enable

ProCurve#show ip ospf neighbor

show ip ospf summary-address

Use the **show ip ospf summary-address** command to display a list of all summary address redistribution information for the system.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

ProCurve>enable

ProCurve#show ip ospf summary-address

show ip policy-class <polycyname>

Use the **show ip policy-class** command to display a list of currently configured access policies. See *ip policy-class <polycyname> max-sessions <number>* [on page 330](#) for information on configuring access policies.

Syntax Description

<polycyname>	Optional. Enter a specific policy class name to display information for a single policy.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show ip policy-class** command:

ProCurve>enable

ProCurve#show ip policy-class

ip policy-class max-sessions 0

Policy-class "Trusted":

0 current sessions (6000 max)

Entry 1 - allow list MatchAll

show ip policy-sessions <polycyname>

Use the **show ip policy-sessions** command to display a list of current policy class associations. See *ip policy-class <polycyname> max-sessions <number>* [on page 330](#) for information on configuring access policies.

Syntax Description

<polycyname>	Optional. Enter a specific policy class name to display information for a single policy.

Default Values

No default value necessary for this command.

Usage Examples

ProCurve>enable

ProCurve#show ip policy-sessions

show ip policy-stats *<polycyname>*

Use the **show ip policy-stats** command to display a list of current policy class statistics. See *ip policy-class* *<polycyname> max-sessions* *<number>* [on page 330](#) for information on configuring access policies.

Syntax Description

<i><polycyname></i>	Optional. Enter a specific policy class name to display information for a single policy.

Default Values

No default value necessary for this command.

Usage Examples

ProCurve>enable

ProCurve#show ip policy-stats

show ip prefix-list [detail | summary] <listname>

Use the **show ip prefix-list** command to display BGP prefix list information.

Syntax Description

detail	Shows a listing of the prefix list rules and their hit counts.
summary	Shows information about the entire prefix list.
<listname>	Specifies to display information for a particular prefix list.

Default Values

No default values are necessary for this command.

Functional Notes

If the **show ip prefix-list** command is issued with no arguments, a listing of the prefix-list rules but no hit count statistics is displayed.

Usage Examples

The following example displays information about the prefix list **test**.

ProCurve#show ip prefix-list test

```
ip prefix-list test: 4 entries
  seq 5 permit 0.0.0.0/0 ge 8 le 8
  seq 10 deny 0.0.0.0/0 ge 9 le 9
  seq 15 permit 0.0.0.0/0 ge 10 le 10
  seq 20 deny 0.0.0.0/0 ge 11
```

show ip protocols

Use the **show ip protocols** command to display IP routing protocol parameters and statistics.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show ip protocols** command:

ProCurve>enable

ProCurve#show ip protocols

Sending updates every 30 seconds, next due in 8 seconds

Invalid after 180 seconds, hold down time is 120 seconds

Redistributing: rip

Default version control: send version 2, receive version 2

Interface	Send Ver.	Rec Ver.

eth 0/1	2	2

ppp 1	2	2

Routing for networks:

1.1.1.0/24

show ip route [connected | ospf | rip | static | table | bgp | summary | summary realtime | <ip address> <subnet>]

Use the **show ip route** command to display the contents of the IP route table.

Syntax Description

connected	Optional. Displays only the IP routes for directly connected networks.
ospf	Optional. Displays only the IP routes associated with OSPF.
rip	Optional. Displays only the IP routes that were dynamically learned through RIP.
static	Optional. Displays only the IP routes that were statically entered.
table	Optional. Displays a condensed version of the IP route table.
bgp	Optional. Displays only the IP routes associated with BGP.
summary	Optional. Displays a summary of all IP route information.
summary realtime	Optional. Displays full-screen output in realtime. See the <i>Functional Notes</i> below for more information.
<ip address><subnet> Displays only the IP routes to destinations within the given address and subnet.	

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output from the **show ip route** command:

ProCurve>enable

ProCurve#show ip route rip

Codes: C - connected S - static R - RIP O - OSPF IA - OSPF inter area
 N1 - OSPF NSSA external type 1 N2 - OSPF NSSA external type 2
 E1 - OSPF external type 1 E2 - OSPF external type 2

Gateway of last resort is 10.15.227.29 to network 0.0.0.0

```
R 10.15.223.0/30 [120/1] via 10.15.227.41, ppp 3
R 10.15.223.2/32 [120/1] via 10.15.227.41, ppp 3
```

```
R 10.15.226.0/28 [120/1] via 10.15.227.41, ppp 3
R 10.15.226.16/28 [120/1] via 10.15.227.41, ppp 3
R 10.15.226.48/28 [120/1] via 10.15.227.29, ppp 1
R 10.15.226.96/28 [120/1] via 10.15.227.29, ppp 1
```

The following example shows how to display IP routes learned via BGP. The values in brackets after a BGP route entry represent the entry's administrative distance and metric:

ProCurve>enable

ProCurve#show ip route bgp

Codes: C - connected, S - static, R - RIP, O - OSPF, B - BGP

IA - OSPF inter area, N1 - OSPF NSSA external type 1

N2 - OSPF NSSA external type 2, E1 - OSPF external type 1

E2 - OSPF external type 2

Gateway of last resort is 10.15.43.17 to network 0.0.0.0

```
B 1.0.0.0/8 [30/0] via 10.15.43.17, fr 1.17
B 2.0.0.0/9 [30/0] via 10.15.43.17, fr 1.17
B 2.128.0.0/10 [30/0] via 10.15.43.17, fr 1.17
B 2.192.0.0/11 [30/0] via 10.15.43.17, fr 1.17
B 2.224.0.0/12 [30/0] via 10.15.43.17, fr 1.17
B 2.240.0.0/13 [30/0] via 10.15.43.17, fr 1.17
B 2.248.0.0/14 [30/0] via 10.15.43.17, fr 1.17
```

show ip traffic [realtime]

Use the **show ip traffic** command to display all IP traffic statistics.

Syntax Description

realtime	Displays full-screen output in real time. See the <i>Functional Notes</i> below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output from the **show ip traffic** command:

ProCurve>enable

ProCurve#show ip traffic

IP statistics:

Routing discards: 0

Rcvd: 15873 total, 7617 delivered

0 header errors, 0 address errors

0 unknown protocol, 0 discards

0 checksum errors, 0 bad hop counts

Sent: 8281 generated, 4459 forwarded

0 no routes, 0 discards

Frag: 0 reassemble required, 0 reassembled, 0 couldn't reassemble

0 created, 0 fragmented, 0 couldn't fragment

UDP statistics:

Rcvd: 3822 total, 0 checksum errors, 0 no port

Sent: 3822 total

TCP statistics:

Retrans Timeout Algorithm: 0

Min retrans timeout (ms): 0

Max retrans timeout (ms): 0

Max TCP Connections: 0

0 active opens, 64 passive opens, 0 failed attempts

5 establish resets, 1 establish current

3795 segments received, 4459 segments sent, 26 segments retransmitted

show lldp

Use the **show lldp** command to display LLDP timer configuration.

Syntax Description

No subcommands.

Default Values

No default values are necessary for this command.

Usage Examples

The following example shows a sample LLDP timer configuration:

ProCurve#show lldp

Global LLDP information:

Sending LLDP packets every 30 seconds

Sending TTL of 120 seconds

show lldp device <system name>

Use the **show lldp device** command to display specific neighbor information about a given neighbor.

Syntax Description

<system name>	Specifies the system name of the neighbor to display.

Default Values

No default values are necessary for this command.

Functional Notes

If there is more than one neighbor with the same system name, all neighbors with that system name will be displayed.

Usage Examples

The following example shows specific information about a neighbor for the system name **Router**:

ProCurve#show lldp device Router

Chassis ID: 00:12:79:02:DD:2A (MAC Address)

System Name: Router

Device Port: eth 0/1 (Locally Assigned)

Holdtime: 30

Platform: 3305

Software: Version: 08.00.22.sw1.D, Date: Mon Nov 01 10:28:55 2004

Capabilities: Bridge, Router

Enabled Capabilities: Router

Local Port: eth 0/2

Management Addresses:

Address Type: IP version 4, Address: 10.23.10.10

Interface Type: Interface Index, Interface Id: 2

show lldp interface <interface>

Use the **show lldp interface** command to display LLDP configuration and statistics for interfaces on this device.

Syntax Description

<interface>	Displays the information for the specified interface. Type show lldp interface ? for a complete list of applicable interfaces.

Default Values

No default values are necessary for this command.

Usage Examples

The following example shows LLDP configuration and statistics for the Ethernet 0/1 interface:

ProCurve#show lldp interface ethernet 0/1

eth 0/1 (TX/RX)

0 packets input

0 input errors

0 TLV errors, 0 TLVs Discarded

0 packets discarded

8799 packets output

0 neighbor ageouts

ProCurve#

show lldp neighbors [interface <interface type> <interface>] [detail]

Use the **show lldp neighbors interface** command to display information about neighbors of this device learned about via LLDP.

Syntax Description

interface <interface type>	Displays a summary of all neighbors learned about through interfaces of the specified type (e.g., eth).
<interface>	Displays a summary of all neighbors learned about through a specified interface (e.g., eth 0/1). Type show lldp neighbors interface ? for a complete list of applicable interfaces.
detail	Optional. Shows detailed neighbor information for the specified interface or interface type.

Default Values

No default values necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following example shows detailed information about a device's neighbors:

ProCurve#show lldp neighbors interface eth 0/2 detail

Chassis ID: 00:12:79:02:DD:2A (MAC Address)

System Name: Router

Device Port: eth 0/1 (Locally Assigned)

Holdtime: 38

Platform: 3305

Software: Version: 08.00.22.sw1.D, Date: Mon Nov 01 10:28:55 2004

Capabilities: Bridge, Router

Enabled Capabilities: Router

Local Port: eth 0/2

Management Addresses:

Address Type: IP version 4, Address: 10.23.10.10

Interface Type: Interface Index, Interface Id: 2

show lldp neighbors realtime

Use the **show lldp neighbors realtime** command to display information about neighbors of this device learned about via LLDP.

Syntax Description

realtime	Displays full-screen output in real time. See the <i>Functional Notes</i> below for more information.

Default Values

No default values necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following example shows detailed information about a device's neighbors in real time:

ProCurve#show lldp neighbors realtime

Chassis ID: 00:12:79:02:DD:2A (MAC Address)

System Name: Router

Device Port: eth 0/1 (Locally Assigned)

Holdtime: 38

Platform: 3305

Software: Version: 08.00.22.sw1.D, Date: Mon Nov 01 10:28:55 2004

Capabilities: Bridge, Router

Enabled Capabilities: Router

Local Port: eth 0/2

Management Addresses:

Address Type: IP version 4, Address: 10.23.10.10

Interface Type: Interface Index, Interface Id: 2

...

show lldp neighbors statistics

Use the **show lldp neighbors statistics** command to display statistics about LLDP neighbor table actions.

Syntax Description

No subcommands.

Default Values

There are no default values necessary for this command.

Functional Notes

This command shows information about the changes in this device's neighbor table. The information displayed indicates the last time a neighbor was added to or removed from the table as well as the number of times neighbors were inserted into or deleted from the table.

Usage Examples

The following example shows sample output for this command:

ProCurve#show lldp neighbors statistics

System Last Change Time	Inserts	Deletes	Drops	Age outs
10-15-2004 14:24:56	55	3	1	1

System Last Change Time - Shows the time at which the most recent change occurred in the neighbor table.

Inserts - Shows the number of times neighbors have been added to the table.

Deletes - Shows how many times neighbors have been deleted from the table because an interface was shut down.

Drops - Shows how many times the insertion of a new neighbor into the table failed because the table was full.

Age outs - Shows how many times neighbors have been removed from the table because no new updates were received from that neighbor before its time-to-live timer expired.

show memory [heap | realtime]

Use the **show memory** command to display statistics regarding memory including memory allocation and buffer use statistics. Shows how memory is in use (broken down by memory size) and how much memory is free.

Syntax Description

heap	Shows how much memory is in use (broken down by memory block size) and how much memory is free.
realtime	Displays full-screen output in real time. See the Functional Notes below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output from the **show memory heap** command:

```
ProCurve>enable
```

```
ProCurve#show memory heap
```

Memory Heap:

HeapFree: 2935792

HeapSize: 8522736

Block Managers:

Mgr	Size	Used	Free	Max-Used
0	0	58	0	58
1	16	1263	10	1273
2	48	1225	2	1227
3	112	432	2	434
4	240	140	3	143
5	496	72	2	74
6	1008	76	1	26
7	2032	25	1	26
8	4080	2	1	3
9	8176	31	1	32
10	16368	8	0	8
11	32752	5	1	6
12	65520	3	0	30
13	131056	0	0	0

show modules

Use the **show modules** command to display a list of the modules currently installed in the system.

Syntax Description

No subcommands.

Default Value

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show modules** command:

ProCurve>enable

ProCurve#show modules

Slot	Port	Type	Part Number	Software Version
1	1	SERIAL WAN		1
1	2	MODEM DBU		1
2	1-2	E1/DSX WAN		1
2	3	MODEM DBU		1
3	—	Empty	—	—

show output-startup

Use the **show output-startup** command to display startup configuration output line-by-line. This output can be copied into a text file and then used as a configuration editing tool.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show output-startup** command:

```
ProCurve>enable
```

```
ProCurve#show output-startup
```

```
!
```

```
ProCurve#!
```

```
ProCurve#hostname "UNIT_2"
```

```
UNIT_2#no enable password
```

```
UNIT_2#!
```

```
UNIT_2#ip subnet-zero
```

```
UNIT_2#ip classless
```

```
UNIT_2#ip routing
```

```
UNIT_2#!
```

```
UNIT_2#event-history on
```

```
UNIT_2#no logging forwarding
```

```
UNIT_2#logging forwarding priority-level info
```

```
UNIT_2#no logging email
```

```
etc....
```

show port-auth supplicant [interface <interface id> | summary]

Use the **show port-auth** command to display supplicant information pertaining to port authentication. The supplicant is the port that will receive services from the port authenticator.

Syntax Description

interface <interface id>	Optional. Shows port authorization supplicant information related to a specific interface. Type show port-auth supplicant interface ? for a complete list of applicable interfaces.
summary	Optional. Shows only basic information about each applicable interface.

Default Values

No default value necessary for this command.

Usage Examples

The following example displays supplicant information for Ethernet interface 0/2:

ProCurve>enable

ProCurve#show port-auth supplicant interface eth 0/2

Interface: eth 0/2

Local Supplicant mode is enabled

Username: User1

Password: securePass

Authorization Status: yes

Supp State Machine: CONNECTED

show pppoe

Use the **show pppoe** command to display all pppoe settings and associated parameters.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example enters the Enable Command mode and uses the **show** command to display pppoe information:

```
ProCurve>enable
```

```
ProCurve#show pppoe
```

```
ppp 1
```

```
  Outgoing Interface: eth 0/1
```

```
  Outgoing Interface MAC Address: 00:12:79:00:85:20
```

```
  Access-Concentrator Name Requested: FIRST VALID
```

```
  Access-Concentrator Name Received: 13021109813703-LRVLGSROS20W_IFITL
```

```
  Access-Concentrator MAC Address: 00:10:67:00:1D:B8
```

```
  Session Id: 64508
```

```
  Service Name Requested: ANY
```

```
  Service Name Available:
```

```
  PPPoE Client State: Bound (3)
```

```
  Redial retries: unlimited
```

```
  Redial delay: 10 seconds
```

```
Backup enabled all day on the following days:
```

```
  Sunday Monday Tuesday Wednesday Thursday Friday Saturday
```

```
Backup phone number list:
```

Number	Call Type	min/max DS0s	Backup I/F
5551212	analog	1/1	ppp 2

show processes [cpu | cpu realtime | history | queue | stack]

Use the **show processes** command to display process statistic information.

Syntax Description

cpu	Displays informations about processes that are currently active.
cpu realtime	Displays full-screen CPU output in real time. See the <i>Functional Notes</i> below for more information.
history	Displays the process switch history.
queue	Displays process queue utilization.
stack	Displays process stack usage.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample output from the **show processes cpu** command:

ProCurve>enable

ProCurve#show processes cpu

processes cpu

System load: 7.07% Min: 0.00% Max 85.89%

Context switch load: 0.21%

Task	Task	Invoked	Exec	Time	Runtime	Load %
D	Name	PRI STAT	(count)	(usec)	(usec)	(1sec)
0	Idle	0 W	129689	1971	927923	92.79
1	FrontPanel	249 W	9658	165	3202	0.32
3	Stack Usage	11 W	485	305	325	0.03
4	Q Test 1	10 W	50	4	0	0.00
5	Q Test 2	11 W	50	6	0	0.00
10	Clock	20 W	1443	24	55	0.01
11	PacketRouting	250 W	31656	10	3871	0.39
12	Thread Pool	50 W	161	159	0	0.00
13	IKE	10 W	2	341	0	0.00
14	RouteTableTick	50 W	49	874	874	0.09

....etc.

show qos map

The **show qos map** command outputs information about the QoS map. This information differs based on how a particular map entry is defined.

Variations of this command include the following:

show qos map

show qos map *<map name>*

show qos map *<map name>* *<sequence number>*

show qos map interface *<interface ID>*

Syntax Description

<i><map name></i>	Enter the name of a defined QoS map.
<i><sequence number></i>	Enter one of the map's defined sequence numbers.
<i><interface ID></i>	Specify an interface to display QoS map information for just that interface (e.g., Frame Relay, PPP, or ATM). Enter the show qos map interface ? command for a complete list of interfaces.

Default Values

No defaults necessary for this command.

Usage Example

ProCurve#show qos map

qos map priority

map entry 10

match IP packets with a precedence value of 6

priority bandwidth: 400 (kilobits/sec) burst: default

packets matched by map: 125520

map entry 20

match ACL icmp

packets matched by map: 99

map entry 30

match RTP packets on even destination ports between 16000 and 17000

packets matched by map: 0

map entry 50

match ACL tcp

packets matched by map: 4326

map entry 60

```
match IP packets with a dscp value of 2
set dscp value to 6
packets matched by map: 0
map entry 70
match NetBEUI frames being bridged by the router
priority bandwidth: 150 (kilobits/sec) burst: default
packets matched by map: 0
```

```
qos map tcp_map
map entry 10
  match ACL tcp
  priority bandwidth: 10 (kilobits/sec) burst: default
  set precedence value to 5
  packets matched by map: 0
map entry 20
  match IP packets with a precedence value of 3
  priority bandwidth: 50 (kilobits/sec) burst: default
  packets matched by map: 0
```

The following example shows the “priority” qos map and all entries in that map:

ProCurve#show qos map priority

```
qos map priority
map entry 10
  match IP packets with a precedence value of 6
  priority bandwidth: 400 (kilobits/sec) burst: default
  packets matched by map: 125520
map entry 20
  match ACL icmp
  packets matched by map: 99
map entry 30
  match RTP packets on even destination ports between 16000 and 17000
  packets matched by map: 0

map entry 50
  match ACL tcp
  packets matched by map: 4326
map entry 60
  match IP packets with a dscp value of 2
  set dscp value to 6
  packets matched by map: 0
map entry 70
  match NetBEUI frames being bridged by the router
  priority bandwidth: 150 (kilobits/sec) burst: default
  packets matched by map: 0
```

The following example shows a particular QoS map entry (in this case map entry 10):

ProCurve#show qos map priority 10

```
qos map priority
map entry 10
  match IP packets with a precedence value of 6
  priority bandwidth: 400 (kilobits/sec) burst: default
  packets matched by map: 125520
```

The following examples show QoS map interface stats associated with the map defined for an interface:

ProCurve#show qos map interface frame-relay 1

```
fr 1
```

```
qos-policy out: priority
```

```
map entry 10
```

```
  match IP packets with a precedence value of 6
  budget 145/10000 bytes (current/max)
  priority bandwidth: 400 (kilobits/sec)
  packets matched on interface: 27289
  packets dropped: 98231
```

```
map entry 20
```

```
not configured for rate limiting
```

```
map entry 30
```

```
not configured for rate limiting
```

```
map entry 50
```

```
not configured for rate limiting
```

```
map entry 60
```

```
not configured for rate limiting
```

```
map entry 70
```

```
  match NetBEUI frames being bridged by the router
  budget 3750/3750 bytes (current/max)
  priority bandwidth: 150 (kilobits/sec)
  packets matched on interface: 0
  packets dropped: 0
```


show queue <interface>

Use the **show queue** command to display conversation information associated with an interface queue. This command shows summary and per-conversation information.

Syntax Description

<interface>	Displays the queueing information for the specified interface. Type the show queue ? command to display a list of valid interfaces.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show queue** command:

ProCurve>enable

ProCurve#show queue fr 1

Queueing method: weighted fair

Output queue: 18/25/200/64/1027 (size/highest/max total/threshold/drops)

Conversations 2/4/256 (active/max active/max total)

(depth/weight/highest/discards) 12/256/33/0

Conversation 10, linktype: ip, length: 67

source: 10.100.23.11, destination: 10.200.2.125, id: 0x0000, ttl: 47,

TOS: 0 prot: 17 (udp), source port 99, destination port 99

(depth/weight/highest/discards) 6/256/25/0

Conversation 23, linktype: ip, length: 258

source: 10.100.23.11, destination: 10.200.2.125, id: 0x0000, ttl: 47,

TOS: 0 prot: 6 (tcp), source port 16, destination port 16

show queuing [fair]

Use the **show queuing** command to display information associated with configured queuing methods.

Syntax Description

fair	Optional keyword used to display only information on the weighted fair queuing configuration.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show queuing** command:

ProCurve>enable

ProCurve#show queuing

Interface	Discard threshold	Conversation subqueues
fr 1	64	256
fr 2	64	256
ppp 1	64	256

show radius statistics

Use the **show radius statistics** command to display various statistics from the RADIUS subsystem. These statistics include number of packets sent, number of invalid responses, number of timeouts, average packet delay, and maximum packet delay. Statistics are shown for both authentication and accounting packets.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is an example output using the **show radius statistics** command:

ProCurve#show radius statistics

	Auth.	Acct.
Number of packets sent:	3	0
Number of invalid responses:	0	0
Number of timeouts:	0	0
Average delay:	2 ms	0 ms
Maximum delay:	3 ms	0 ms

show running-config interface tunnel <id> verbose

show route-map [<name>]

Use the **show route-map** command to display any route maps that have been configured in the router. It displays any match and set clauses associated with the route map, as well as the number of incoming routes that have matched each route map.

Syntax Description

<name>	Optional. Displays only the route map matching the specified name.

Default Values

By default, this command displays all defined route maps.

Usage Examples

In the example below, all route maps in the router are displayed.

ProCurve>enable

ProCurve#show route-map

route-map RouteMap1, permit, sequence 10

Match clauses:

community (community-list filter): CommList1

Set clauses:

local-preference 250

BGP Filtering matches: 75 routes

route-map RouteMap1, permit, sequence 20

Match clauses:

community (community-list filter): CommList2

Set clauses:

local-preference 350

BGP Filtering matches: 87 routes

route-map RouteMap2, permit, sequence 10

Match clauses:

ip address (access-lists): 192.168.1.1

Set clauses:

metric 100

BGP Filtering matches: 10 routes

route-map RouteMap2, permit, sequence 20

Match clauses:

ip address (access-lists): 192.168.2.1

Set clauses:

metric 200

BGP Filtering matches: 12 routes

In the example below, only the route map with the name "RouteMap2" is displayed.

ProCurve>enable

ProCurve#show route-map RouteMap2

route-map RouteMap2, permit, sequence 10

Match clauses:

ip address (access-lists): 192.168.1.1

Set clauses:

metric 100

BGP Filtering matches: 10 routes

route-map RouteMap2, permit, sequence 20

Match clauses:

ip address (access-lists): 192.168.2.1

Set clauses:

metric 200

BGP Filtering matches: 12 routes

show running-config

Use the **show running-config** command to display a text print of all the non-default parameters contained in the current running configuration file. Use the **verbose** keyword to display a text print of the entire configuration (including parameters in their default state). Specific portions of the running-config may be displayed, based on the command entered.

Variations of this command include the following:

```
show running-config
show running-config verbose
show running-config access-lists
show running-config access-lists verbose
show running-config checksum
show running-config interface <interface type> <interface id>
show running-config interface <interface type> <interface id> verbose
show running-config policy-class
show running-config policy-class verbose
show running-config qos-map
show running-config qos-map verbose
```

Syntax Description

access-lists	Displays the current running configuration for all configured IP access lists.
checksum	Optional. Displays the encrypted Message Digest 5 (MD5) version of the running configuration.
interface <interface type>	Displays the current running configuration for a particular interface. Type show running-config interface ? for a list of valid interfaces.
<interface id>	Specifies any valid slot/port interface (e.g., 0/1).
policy-class	Displays the current running configuration for all configured policy classes.
qos-map	Displays the current running configuration for all configured QoS maps.
verbose	Optional. Displays the entire running configuration to the terminal screen (versus only the non-default values).

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show running-config** command:

```
ProCurve>enable
ProCurve#show running-config
Building configuration...
!
```

```
no enable password
!
ip subnet-zero
ip classless
ip routing
!
event-history on
no logging forwarding
logging forwarding priority-level info
no logging email
!
ip policy-timeout tcp all-ports 600
ip policy-timeout udp all-ports 60
ip policy-timeout icmp 60
!
interface eth 0/1.....
```

show snmp

Use the **show snmp** command to display the system Simple Network Management Protocol (SNMP) parameters and current status of SNMP communications.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is an example output using the **show snmp** command for a system with SNMP disabled and the default Chassis and Contact parameters:

> **show snmp**

Chassis: Chassis ID

Contact Name:

Contact Phone:

Contact Email:

Contact Pager:

Management URL:

Management URL Label:

0 Rx SNMP packets

0 Bad community names

0 Bad community uses

0 Bad versions

0 Silent drops

0 Proxy drops

0 ASN parse errors

show sntp

Use the **show sntp** command to display the system Simple Network Time Protocol (SNTP) parameters and current status of SNTP communications.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

```
> show sntp
```

show spanning-tree <bridgegroup#> [realtime]

Use the **show spanning-tree** command to display the status of the spanning-tree protocol.

Syntax Description

<bridgegroup#>	Optional. Display spanning-tree for a specific bridge group.
realtime	Shows full screen spanning-tree protocol status real time.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is an example output using the **show spanning-tree** command:

ProCurve>enable

ProCurve#show spanning-tree

Spanning Tree enabled protocol ieee

Root ID Priority 32768

Address 00:12:79:00:88:41

We are the root of the spanning tree

Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32768

Address 00:12:79:00:88:41

Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Aging Time 300

Interface	Role	Sts	Cost	Prio.Nbr	Type
-----	-----	-----	-----	-----	-----
eth 0/2	Desg	FWD	19	128.2	P2p

show startup-config

Use the **show startup-config** command to display a text printout of the startup configuration file stored in flash memory.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output of the **show startup-config** command:

```
ProCurve>enable
ProCurve#show startup-config
!
!
no enable password
!
ip subnet-zero
ip classless
ip routing
!
event-history on
no logging forwarding
logging forwarding priority-level info
no logging email
!
ip policy-timeout tcp all-ports 600
ip policy-timeout udp all-ports 60
ip policy-timeout icmp 60
!
!
!
interface eth 0/1
speed auto
no ip address
shutdown
!
interface bri 1/2
shutdown
```

```
!  
!  
ip access-list standard MatchAll  
  permit host 10.3.50.6  
  permit 10.200.5.0 0.0.0.255  
!  
!  
ip access-list extended UnTrusted  
  deny  icmp 10.5.60.0 0.0.0.255 any source-quench  
  deny  tcp any any  
!  
no ip snmp agent  
!  
!  
!
```

show startup-config checksum

Use the **show startup-config checksum** command to display the MD5 checksum of the unit's startup configuration.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Functional Notes

This command is used in conjunction with the show running-config checksum command to determine whether the configuration has changed since the last time it was saved.

Usage Examples

The following example displays the MD5 checksum of the unit's startup configuration:

```
ProCurve#show startup-config checksum
```

```
10404D5DAB3FE35E307B6A79AC6AC8C0
```

```
ProCurve#
```

```
ProCurve#show running-config checksum
```

```
10404D5DAB3FE35E307B6A79AC6AC8C0
```

```
ProCurve#
```

show tcp info [realtime] <control block>

Use the **show tcp info** command to display TCP control block information in the Secure Router OS. This information is for troubleshooting and debug purposes only. For more detailed information, you can optionally specify a particular TCP control block. When a particular TCP control block is specified, the system provides additional information regarding crypto map settings that the **show tcp info** command does not display.

Syntax Description

realtime	Displays full-screen output in real time. See the Functional Notes below for more information.
<control block>	Optional. Specify a particular TCP control block for more detailed information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample from the **show tcp info** command:

ProCurve>enable

ProCurve#show tcp info

TCP TCB Entries

ID	STATE	LSTATE	OSTATE	TYPE	FLAGS	RPORT	LPORT	SWIN	SRT	INTERFACE
0	FREE	FREE	FREE	SRVR	0	0	0	0	0	NONE
1	LISTEN	FREE	FREE	CONN	0	0	21	0	0	NONE
2	LISTEN	FREE	FREE	CONN	0	0	80	0	0	NONE
3	LISTEN	FREE	FREE	CONN	0	0	23	0	0	NONE
4	LISTEN	FREE	FREE	CONN	0	0	5761	0	0	NONE
5	FREE	FREE	FREE	SRVR	0	0	0	0	0	NONE
.										
.										
31	FREE	FREE	FREE	SRVR	0	0	0	0	0	NONE

show tech [terminal]

Use the **show tech** command to create a file **showtech.txt** located in flash memory that contains pertinent system information collected from various **show** commands. Instead of writing to a file, display the created file on the terminal screen using the **terminal** option.

Syntax Description

terminal	Optional. The terminal option displays the showtech.txt file output to the terminal screen real time.

Default Value

No default value necessary for this command.

Functional Notes

The **show tech** command runs a system file that creates a **showtech.txt** file in flash memory that contains terminal displays from the following show commands:

show version
show modules
show flash
show cflash
show running-config verbose
show interfaces
show atm pvc
show dial-backup interfaces
show frame-relay lmi
show frame-relay pvc
show ip bgp neighbors
show ip bgp summary
show ip ospf neighbor
show ip ospf summary-address
show ip mroute
show ip bridge
show spanning-tree
show ip interfaces
show connections
show arp
show ip traffic
show tcp info
show ip protocols
show ip route
show ip access-lists
show event-history
show output-startup

show processes cpu
show buffers
show buffers users
show memory heap
show debugging

Usage Examples

The following creates a **showtech.txt** file and displays it to the terminal screen:

```
ProCurve>enable  
ProCurve#show tech
```

Opening and applying file.....
Done.

```
ProCurve#show file flash showtech.txt  
Using 57693 bytes
```

```
ProCurve#!  
ProCurve#!  
ProCurve#!  
ProCurve# show version  
ProCurve Secure Router 7102dl  
SROS Version: J03.01.01  
Checksum: 5C8D29BE, built on: Mon Jul 25 16:14:46 2005  
Boot ROM version J03.01.01  
Checksum: 49C7, built on: Mon Jul 25 16:15:52 2005  
Copyright (c) 2005-2005, Hewlett-Packard, Co.  
Platform: ProCurve Secure Router 7102dl  
Serial number US449TR019  
Flash: 33554432 bytes DRAM: 134217727 bytes
```

System uptime is 0 days, 0 hours, 14 minutes, 40 seconds

```
Current system image:      "CFLASH:/SROS.BIZ"  
Current configuration-file: "CFLASH:/startup-config"  
Configured system image path:  
Primary:                  "CFLASH:/SROS.BIZ"  
Configured configuration-file path:  
Primary:                  "CFLASH:/startup-config"  
Backup:                   "NONVOL:/startup-config"
```

```
ProCurve#!  
ProCurve#!
```


ProCurve# show modules

Slot	Port	Type	Part Number	Software Version
1	1-2	T1/T1 WAN	Not Available	1
2	1-2	E1/E1 WAN	Not Available	1
2	3	BRI U DBU	Not Available	1
3	-	Empty	-	-

ProCurve#!

ProCurve#!

ProCurve# show flash

287413 J01_02B-boot.biz

3775 startup-config

5166 startup-config.bak

..... etc.

show udp info [realtime] <session id>

Use the **show udp info** command to display User Datagram Protocol (UDP) session information.

Syntax Description

realtime	Displays full-screen output in real time. See the <i>Functional Notes</i> below for more information.
<session id>	Specifies ID of session to display (Range: 0 to 31).

Default Values

No default necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following example shows sample output from the **show udp info** command:

ProCurve>enable

ProCurve#show udp info

UDP Session Entries

ID	Local Port	IP Address	Socket
---	-----	-----	-----
2	520	0.0.0.0	1
3	0	0.0.0.0	4
4	161	0.0.0.0	5
5	8	127.0.0.1	7
6	10	0.0.0.0	11
7	6	127.0.0.1	16
8	4	127.0.0.1	17
9	14	127.0.0.1	18
10	12	127.0.0.1	19

show users [realtime]

Use the **show users** command to display the name (if any) and state of users authenticated by the system. Displayed information includes:

- Connection location (for remote connections this includes TCP information)
- Username of authenticated user
- Current state of the login (in process or logged in)
- Current enabled state
- Time the user has been idle on the connection

Syntax Description

realtime	Displays full-screen output in real time. See the <i>Functional Notes</i> below for more information.

Default Values

No default value necessary for this command.

Functional Notes

Use the **realtime** argument for this command to display full-screen output in real time. Information is continuously updated on the console until you either freeze the data (by pressing the **F** key) or exit realtime mode (by pressing **Ctrl-C**). If there is not enough room on the screen for all available data, the information will truncate at the bottom of the screen. In order to maximize the amount of data displayed, increase the terminal length (using the **#terminal length** command).

Usage Examples

The following is a sample of **show users** output:

ProCurve>enable

ProCurve#show users

```
- CONSOLE 0 'user' logged in and enabled
  Idle for 00:00:00
- TELNET 0 (172.22.12.60:3998) 'password-only' logged in (not enabled)
  Idle for 00:00:14
- FTP (172.22.12.60:3999) 'user' logged in (not enabled)
  Idle for 00:00:03
```

show version

Use the **show version** command to display the current Secure Router OS version information.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **show version** output:

ProCurve>enable

ProCurve#show version

ProCurve Secure Router 7203dl

SROS Version: J03.01

Checksum: 4F8DCF96, built on: Tue Dec 21 08:32:18 2004

Boot ROM version J03.01

Checksum: B133, built on: Tue Dec 21 08:32:25 2004

Copyright (c) 2004-2005, Hewlett-Packard, Co.

Platform: ProCurve Secure Router 7203dl

Serial number US449TS058

Flash: 33554432 bytes DRAM: 268435455 bytes

System uptime is 0 days, 0 hours, 22 minutes, 42 seconds

Current system image file: "CFLASH:/SROS.BIZ

Current configuration-file: CFLASH:/startup-config"

Configured system image path:

Primary: "CFLASH:/SROS.BIZ"

Backup: "NONVOL:/SROS.BIZ"

Configured configuration-file path:

Primary: "CFLASH:/startup-config"

Backup: "NONVOL:/startup-config"

ProCurve#

show version [cflash | flash] <filename>

Use the **show version cflash** and **show version flash** commands to display the current Secure Router OS version information for a specified .biz file. When this command is entered, the system opens the .biz file specified and returns the current Secure Router OS version information.

Syntax Description

cflash	Specifies a .biz file located in the compact flash memory.
flash	Specifies a .biz file located in flash memory.
<filename>	Specifies the exact filename of the .biz file for the system to determine the version information.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **show version cflash** output:

ProCurve>enable

ProCurve#show version cflash SROS.BIZ

Version: 03.01.00

telnet <address>

Use the **telnet** command to open a Telnet session (through the Secure Router OS) to another system on the network.

Syntax Description

<address>	Specifies the IP address of the remote system.

Default Values

No default value necessary for this command.

Usage Examples

The following example opens a Telnet session to a remote system (**10.200.4.15**):

```
ProCurve>enable
```

```
ProCurve#telnet 10.200.4.15
```

User Access Login:

Password:

terminal length <lines>

The **terminal length** command sets the number of rows (lines) for a terminal session. Use the **no** form of this command to disable this feature. Use the **no** form of this command to return to the default terminal length.

Syntax Description

<lines>	Number of rows (lines) for the terminal session (Range: 0 to 480).

Note	<i>Setting the terminal length to zero disables paging.</i>

Default Values

The default setting for this command is 24 rows.

Usage Examples

The following example sets the number of rows to 30.

ProCurve>enable

ProCurve#terminal length 30

traceroute <address>

Use the **traceroute** command to display the IP routes a packet takes to reach the specified destination.

Syntax Description

<address> Optional. Specifies the IP address of the remote system to trace the routes to.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample **traceroute** output:

ProCurve>enable

ProCurve#traceroute 192.168.0.1

Type CTRL+C to abort.

Tracing route to 192.168.0.1 over a maximum of 30 hops

```
 1  22ms  20ms  20ms   192.168.0.65
 2  23ms  20ms  20ms   192.168.0.1
```

ProCurve#

undebug all

Use the **undebug all** command to disable all activated debug messages.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example disabled all activated debug messages:

```
ProCurve>enable
ProCurve#undebug all
```

wall <*message*>

Use the **wall** command to send messages to all users currently logged in to the Secure Router OS unit.

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following example sends the message "Reboot in 5 minutes if no objections" to the CLI screen of everyone currently connected:

```
ProCurve>enable
```

```
ProCurve#wall Reboot in 5 minutes if no objections
```

write [erase | memory | network | terminal]

Use the **write** command to save the running configuration to the unit's flash memory or a TFTP server. Also use the **write** command to clear flash memory or to display the running configuration on the terminal screen. Entering the **write** command with no other arguments copies your configuration changes to the unit's flash memory. Once the save is complete, the changes are retained even if the unit is shut down or suffers a power outage.

Syntax Description

erase	Optional. Erase the configuration files saved to the unit's flash memory.
memory	Optional. Save the current configuration to flash memory. See <i>copy running-config [cflash flash startup-config] <filename></i> on page 66 for more information.
network	Optional. Save the current configuration to the network TFTP server. See <i>copy tftp <destination></i> on page 72 for more information.
terminal	Optional. Display the current configuration on the terminal screen.

Default Values

No default value necessary for this command.

Usage Examples

The following example saves the current configuration to the unit's flash memory:

ProCurve>enable

ProCurve#write memory

GLOBAL CONFIGURATION MODE COMMAND SET

To activate the Global Configuration mode, enter the **configuration** command at the Enable security mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

aaa commands [begin on page 238](#)
arp <ip address> <mac address> *arpa* [on page 246](#)
autosynch-mode [on page 247](#)
banner [exec | login | motd] <character> <message> <character> [on page 249](#)
boot config [cflash | flash] <filename> [cflash | flash] <backup filename> [on page 250](#)
boot system [cflash | flash] <filename> [cflash | flash] [no-backup | <backup filename>] [on page 251](#)
bridge <group#> *protocol ieee* [on page 252](#)
crypto commands [begin on page 253](#)
enable password [md5] <password> [on page 269](#)
event-history [on page 270](#)
event-history priority [error | fatal | info | notice | warning] [on page 271](#)
exception report [file-name <filename>] [on page 273](#)
ftp authentication <listname> [on page 274](#)
hostname <name> [on page 275](#)
interface commands [begin on page 276](#)
ip commands [begin on page 285](#)
line [console | telnet] <line-number> <ending number> [on page 345](#)
lldp [minimum-transmit-interval l reinitialization-delay l transmit-interval l ttl-multiplier] <numeric value> [on page 346](#)
logging commands [begin on page 347](#)
mac address-table aging-time <aging time> [on page 359](#)

mac address-table static <mac address> bridge <group#> interface [atm | ethernet | frame-relay | hdlc | ppp] <interface id> [on page 360](#)

modem countrycode [<countrycode>] [on page 361](#)

qos map <mapname><sequence number> [on page 363](#)

radius-server [on page 364](#)

radius-server host [on page 366](#)

router ospf [on page 369](#)

router rip [on page 370](#)

snmp-server commands [begin on page 372](#)

sntp server <address or hostname> version <1-3> [on page 383](#)

spanning-tree commands [begin on page 384](#)

username <username> password <password> [on page 393](#)

aaa authentication [banner <delimiter> | fail-message <delimiter> | password-prompt | username-prompt]

Use the **aaa authentication** command to control various features of the AAA subsystem authentication process. For more detailed information on AAA functionality, refer to the *Technology Review* section of the command *aaa* on [page 243](#).

Syntax Description

banner <delimiter>	Sets the banner shown before user authentication is attempted. The banner can be multiple lines. Enter a delimiter (such as :) to begin recording the typed text message used for the banner. The message must end with the same delimiter to indicate that the message is complete. The text delimiters are not displayed to the screen during operation.
fail-message	Sets the message shown if user authentication fails. The message can be multiple lines. Enter a delimiter (such as :) to begin recording the typed text message displayed after a failed authentication attempt. The message must end with the same delimiter to indicate that the message is complete. The text delimiters are not displayed to the screen during operation.
password-prompt	Sets the prompt for the user's password. The prompt is a single line. Enclose the string in quotation marks.
username-prompt	Sets the prompt for the user's name. The prompt is a single line. Enclose the string in quotation marks.

Default Values

banner	User Access Verification
fail-message	Authentication Failed
password-prompt	Password:
username-prompt	Username:

Usage Examples

The following example defines a banner of "Welcome to the ProCurve Secure Router":

ProCurve(config)#aaa authentication banner :

Enter TEXT message. End with the character `:`.

Welcome to the ProCurve Secure Router.:

The following example defines an authentication failed message of "Authentication Failed. Contact IT for further assistance.":

ProCurve(config)#aaa authentication fail-message :

Enter TEXT message. End with the character `:`.

Authentication Failed. Contact IT for further assistance.:

The following example defines a password prompt of "PW:":

```
ProCurve(config)#aaa authentication password-prompt "PW:"  
ProCurve(config)#
```

The following example defines a Username prompt of "User:":

```
ProCurve(config)#aaa authentication username-prompt "User:"  
ProCurve(config)#
```

aaa authentication enable default [none | line | enable | groupname]

Use the **aaa authentication enable default** command to create (or change) the list of methods used for privileged mode access authentication. For more detailed information on AAA functionality, refer to the *Technology Review* section of the command *aaa* on [page 243](#).

Syntax Description

none	Specifies that access is automatically granted.
line	Specifies to use the line password.
enable	Specifies to use the enable password.
groupname	Specifies to use the group of remote servers. The group name radius uses all defined RADIUS servers.

Default Values

If there is no default list configured, the default behavior is to use the enable password for the unit. If there is no password configured, consoles are allowed in (prevents a lock-out).

Functional Notes

A user is authenticated by trying the list of methods from first to last until a method succeeds or fails. If a method is unable to complete, the next method is tried. The group falls through if the servers in the remote group could not be found.

Note that enable access is a password-only process. The local user database cannot be used and the username given to any remote RADIUS server is **\$enab15\$**. The only list name allowed is **default**.

Usage Examples

ProCurve(config)#aaa authentication enable default line

aaa authentication login <listname> [none | line | enable | local | group]

Use the **aaa authentication login** to create (or change) a list of methods for user authentication. For more detailed information on AAA functionality, refer to the *Technology Review* section of the command *aaa on page 243*.

Syntax Description

<listname>	Specifies the name of the list.
none	Access automatically granted.
line	Use the line password (Telnet 0-4 or console 0-1).
enable	Use the enable password.
local	Use the local user database.
group	Use a group of remote RADIUS servers.

Default Values

The login list named **default** is the default list used to authenticate users when no other list is assigned to the line.

Functional Notes

A user is authenticated by trying the list of methods from first to last until a method succeeds or fails. If a method is unable to complete, the next method is tried. The local user database falls through to the next method if the username does not appear in the database. The group falls through if the servers in the remote group could not be found. See the command *radius-server on page 364* for information on defining RADIUS server groups.

Usage Examples

```
ProCurve(config)#aaa authentication login myList local group myGroup line
ProCurve(config)#aaa authentication login default local
```

aaa group server radius <listname>

Use the **aaa group server radius** command to group pre-defined RADIUS servers into named lists. For more detailed information on AAA functionality, refer to the *Technology Review* section of the command *aaa* on [page 243](#).

Syntax Description

<listname>	Specifies the name of the list.

Default Values

No default value necessary for this command.

Functional Notes

Use the **radius-server** command to specify RADIUS servers before adding them to a group. This command enters a for adding individual servers to the named group. See *Radius Group Command Set* on [page 503](#) for more information.

The default group cannot be changed and includes all RADIUS servers in the order they were specified by the **radius-server** commands.

Usage Examples

The following example creates the named list **myServers** and enters the Radius Group:

```
ProCurve(config)#aaa group server radius myServers
ProCurve(config-sg-radius)#
```

aaa on

Use the **aaa on** command to activate the AAA subsystem. Use the **no** form of this command to deactivate AAA.

Syntax Description

No subcommands.

Default Values

By default, AAA is not activated.

Functional Notes

By default, the AAA subsystem is turned off and authentication follows the line technique (local, line, etc.). Once activated, the AAA lists override the methods specified in the line command.

Technology Review

AAA stands for authentication, authorization, and accounting. The Secure Router OS AAA subsystem currently supports authentication. Authentication is the means by which a user is granted access to the device (router). For instance, a username/password is authenticated before the user can use the CLI. VPN clients can also verify username/password before getting access through the device.

There are several methods that can be used to authenticate a user:

NONE	Instant access
LINE-PASSWORD	Use the line password (telnet 0-4 or console 0-1)
ENABLE-PASSWORD	Use the enable password
LOCAL-USERS	Use the local user database
GROUP <groupname>	Use a group of remote RADIUS servers

The AAA system allows the user to create a named list of these methods to try in order (in case one fails, it falls to the next one). This named list is then attached to a portal (telnet 0-4 or console 0-1). When a user telnets in or accesses the terminal, the AAA system uses the methods from the named list to authenticate the user.

The AAA system must be turned on to be active. By default it is off. Use the **aaa on** command to activate the AAA system.

If a portal is not explicitly assigned a named list, the name **default** is automatically assigned to it. The user can customize the **default** list just like any other list. If no **default** list is configured, the following default behavior applies (defaults are based on portal):

- Instant access (NONE) is assigned to the CONSOLE using the **default** list (when the list has not been configured).
- The local user database is used for TELNETS using the **default** list (when the list has not been configured).

- No access is granted for FTP access using the **default** list (when the list has not been configured).

Methods fail (and therefore cause the system to proceed to the next configured method) under circumstances such as the following:

- LINE and ENABLE passwords fall through if there is no LINE or ENABLE password configured.
- LOCAL USERS fall through if the given user is not in the database.
- RADIUS servers fall through if the given server(s) cannot be contacted on the network.

Example

For a default list defined with the order [LINE, ENABLE, LOCAL, and GROUP **mygroup**], the following statements are true:

- If there is no LINE password, the list falls through to the ENABLE password.
- If there is no ENABLE password, the AAA system prompts the user for a username and password for the local user database.
- If the given user is not in the local list, the username and password are handed to the remote servers defined in **mygroup**.
- A failure at any point (password not matching) denies access.

If the AAA process falls through the list completely, system behavior is based on portal:

- CONSOLE access is granted if the process falls completely through (this prevents a lock-out condition).
- TELNET and FTP are denied access.

Usage Examples

The following example activates the AAA subsystem:

```
ProCurve(config)#aaa on
```

aaa processes <threads>

Use the **aaa processes** command to set the number of threads available to the AAA subsystem. Use the **no** form of this command to return to the default setting. For more detailed information on AAA functionality, refer to the *Technology Review* section of the command *aaa* on [page 243](#).

Syntax Description

<threads>	Specifies the number of threads available to the AAA subsystem. Range: 1 to 64.

Default Values

By default, this is set to 1 process.

Functional Notes

Increasing this number may speed up simultaneous authentication at the cost of system resources (e.g., memory).

Usage Examples

The following example specifies five available threads for the AAA subsystem:

ProCurve(config)#**aaa processes 5**

arp <ip address> <mac address> arpa

Use this command to enter static entries into the address resolution protocol (ARP) table.

Syntax Description

arpa	Sets the standard address resolution protocol for this interface.
<ip address>	Specifies the IP address.
<mac address>	Specifies the MAC address.

Default Values

The default for this command is **arpa**.

Usage Examples

The following example enables standard ARP for the VLAN interface:

```
ProCurve(config)#interface vlan 1
```

```
ProCurve(config-interface-vlan 1)#arp 196.173.22.253 00:12:79:00:00:01 arpa
```

autosynch-mode

Use the **autosynch** command to configure the SROS to synchronize the **startup-config** and **SROS.BIZ** files located in the system flash memory and the compact flash card. Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

The AutoSynch™ feature is disabled by default.

Functional Notes

The AutoSynch™ features configures the system to synchronize the **startup-config** and **SROS.BIZ** files located in the system flash memory and the compact flash card. When enabled, the system compares the two files in the two locations and replaces the files located in the system flash memory with the ones from the compact flash card (regardless of which set of files is more current). This allows the customer to maintain the version of the operating system, and the configuration for that operating system, at the desired level. To accomplish this, a synchronization check is performed on the system any time there is a change in **startup-config** or **SROS.BIZ** on the compact flash card.

The autosynch feature allows for quick installation and updates of routers by inserting a compact flash card containing the desired software (must be renamed from the desired operating system software, such as J03.01.biz to **SROS.BIZ**) and startup configuration file (must be named **startup-config**) into a router with AutoSynch™ enabled. The ProCurve Secure Routers automatically boot from the compact flash (and secondarily from internal flash). After booting, with AutoSynch™ enabled, the router will synchronize the files in system flash memory with the desired files from compact flash.

Caution	<i>Deleting the SROS.BIZ and startup-config files from the compact flash card (using the erase command) deletes the files from the system flash memory as well.</i>

Functional Notes

Status commands associated with the AutoSynch™ feature include **show version flash SROS.BIZ** and **show autosynch-status**.

The **show version flash SROS.BIZ** command opens the specified .biz file and returns the current Secure Router OS version information.

```
ProCurve>enable
#show version flash SROS.BIZ
```

Version:J03.01.00

The **show autosynch-status** command displays the current AutoSynch™ configuration and the statistics for the **SROS.BIZ** and **startup-config** files (if AutoSynch™ is enabled).

```
ProCurve>enable
#show autosynch-status
```

AutoSynch: Mode - Enabled

AutoSynch: SROS.BIZ synched
AutoSynch: startup-config synched

Usage Examples

The following example enables the autosynch feature:

```
ProCurve(config)#autosynch-mode
```


banner [exec | login | motd] <character> <message> <character>

Use the **banner** command to specify messages to be displayed in certain situations. Use the **no** form of this command to delete a previously configured banner.

Syntax Description

exec	Creates a message to be displayed when any exec-level process takes place.
login	Creates a message to be displayed before the username and password login prompts.
motd	Creates a message-of-the-day (MOTD) banner.
<character>	Banner text delimiter character. Press Enter after the delimiter to begin input of banner text.
<message>	Specifies the text message you wish to display. End with the character that you chose as your delimiter.

Default Values

By default, no banners are configured.

Functional Notes

Banners appear in the following order (if configured):

- MOTD banner appears at initial connection.
- Login banner follows the MOTD banner.
- Exec banner appears after successful log in.

Usage Examples

The following example configures the system to display a message of the day:

```
ProCurve(config)#banner motd *The system will be shut down today from 7PM to 11PM*
```

boot config [cflash | flash] <filename> [cflash | flash] <backup filename>

Use the **boot config** command to modify system boot parameters.

Syntax Description

cflash	Specifies primary/backup configuration file located in compact flash memory.
flash	Specifies primary/backup configuration file located in flash memory.
<filename>	Specifies the filename of the configuration file (filenames are case-sensitive).
<backup filename>	Specifies a name for the backup configuration file.

Default Values

No default is necessary for this command.

Usage Examples

The following example specifies the file **myconfig**, located in flash memory, as the system boot file:

ProCurve(config)#**boot config flash myconfig**

boot system [cflash | flash] <filename> [cflash | flash] [no-backup | <backup filename>]

Use the **boot config** command to specify the system software loaded at startup.

Syntax Description

cflash	Specifies primary/backup file located in compact flash memory.
flash	Specifies primary/backup file located in flash memory.
<filename>	Specifies the filename of the software (filenames are case-sensitive) - software files should have a .biz or .BIZ extension.
no-backup	Specifies that no backup software is to be saved to the system.
<backup filename>	Specifies a name for the backup software.

Default Values

No default is necessary for this command.

Usage Examples

The following example specifies the file **SROS.BIZ**, stored in flash memory, as the startup software:

```
ProCurve(config)#boot system flash SROS.BIZ
```

bridge <group#> protocol ieee

The **bridge protocol ieee** command configures a bridge group for the IEEE Spanning Tree Protocol. Use the **no** form of this command (with the appropriate arguments) to delete this setting.

Syntax Description

<group#>	Specifies bridge group number (1 to 255) (using the bridge command).
ieee	IEEE 802.1 Ethernet spanning-tree protocol.

Default Values

By default, all configured bridge interfaces implement **ieee** spanning-tree protocol.

Usage Examples

The following example deletes the bridge protocol setting for bridge-group 17:

```
ProCurve(config)#no bridge 17 protocol ieee
```

crypto ca authenticate <name>

Use the **crypto ca authenticate** command to initiate CA authentication procedures.

Syntax Description

<name>	Specifies CA profile by alphanumeric string of up to 32 characters.

Default Values

No defaults necessary for this command.

Functional Notes

The type of authentication procedure is based on the **enrollment** command and its settings. See *enrollment terminal* [on page 509](#) and *enrollment url <url>* [on page 510](#) for more information. When **enrollment** is set to **terminal**, the CA authentication process is done manually, as shown in the example which follows (see **Usage Examples** for this command).

Usage Examples

The following example initiates the CA authentication process:

ProCurve(config)#**crypto ca authenticate testCAprofile**

Enter the base 64 encoded CA certificate. End with two consecutive carriage returns or the word "quit" on a line by itself:

-----BEGIN X509 CERTIFICATE-----

```
MIIDEDCCAs6gAwIBAgICAXlwCwYHKoZlZigEAWUAMFoxCzAJBgNVBAYTAkZJMSQw
IgYDVQQKEExtTU0ggQ29tbXVuaWNhdGlvbnMgU2VjdXJpdHkxETAPBgNVBAsTCFdl
YiB0ZXN0MRlwEAYDVQQDEwIUZXN0IENBIDQwHhcNMDMwMTA5MTYyNTE1WhcNMDMx
MjMxMjM1OTU5WjBaMQswCQYDVQQGEwJGSTEKMCIGA1UEChMbU1NIIENvbW11bmli
YXRpb25zIFNlY3VyaXR5MR5MR5MR5MR5MR5MR5MR5MR5MR5MR5MR5MR5MR5MR5MR5
dCBDQSA0MIIIBtzCCASsGBYqGSM44BAEwggEeAoGBAPTo+NdCWh87hOSnuZ7dUL07
twjZZwY3beLHnDsERhfN8XoOZZcfulKc/lqTrYiu7M5yPJsXQ3u8dbCb6RWFU0A
T5Nd7/4cNn/hCmhbe6xqsNZUsOcTZJxvClq8thkNo+gXg5bw0fiElgxZ/IEbFWL
UzeO8KgM4izkq0CrGtaFAhUA2+ja4RgbbgTgJk+qTXAxicG/8JMCgYBZvcPMO2/Y
Zc2sXYrBPtv6k2ZGGYqXAUZ98/txm37JwQGafygePJ/64oeisVeDcLf2FTjveex
W5saydjSK00jXjreRZcJFEDmfRhUtWR8K8tm8mEnB3eg9n09lkWibljihHn7n5MF
tBBAdBRHyctsr3DyofnieTt3DY78MDsNbgOBhQACgYEA6EKDS2lXrdMsogHfVvob
PkDSv2FjOsP5Tomc/tf9jvvuf6+v9XTw+uAg1BU9/TyjGzAtnRrCvOUkTYoVxRY
vdDOI3GR2RcyNVdGrhYXWY1I5XuB5+NWij8VUQOgfXsJgbEMvPemECeYwQ4ASdhD
vw0E8NI2AEkJXsCAvYfXWzujlzAhMAsGA1UdDwQEAWIBhjASBgNVHRMBAf8ECDAG
AQH/AgEyMAsGBYqGSM44BAMFAAMvADAsAhRa0ao0FbRQeWCC2oC24OZ1YZi8egIU
IZhxKAclhXksZHvOj+yIld5x0ec=
```

-----END X509 CERTIFICATE-----

quit

Hash: 4e904504dc4e5b95e08129430e2a0b97ceef0ad1394f905b42df2dfb8f751be0244a711bb0
6eddaa2f07dd640c187f14c16fa0bed28e038b28b6741a880539d6ed06a68b7e324bfdde6f3d0b17
83d94e58fd4943f5988a7a0f27f6b6b932dc0410378247160752853858dbe7a1951245cfb14b109e
ffc430e177623720de56f4

* Do you accept this certificate? [y]**y**

crypto ca certificate chain <name>

Use the **crypto ca certificate chain** command to enter the Certificate Configuration for the specified CA. See *Certificate Configuration Command Set* [on page 516](#) for more information.

Syntax Description

<name>	Specifies CA profile by alphanumeric string of up to 32 characters.

Default Values

No defaults necessary for this command.

Functional Notes

Typically used only in the **running-config** and **startup-config** to restore certificates.

Usage Examples

The following example enters the Certificate Configuration for the CA profile **MyProfile**:

```
ProCurve(config)#crypto ca certificate chain MyProfile
ProCurve(config-cert-chain)#
```

crypto ca enroll <name>

Use the **crypto ca enroll** command to begin CA enrollment procedures.

Syntax Description

<name>	Specifies CA profile by alphanumeric string of up to 32 characters.

Default Values

No defaults necessary for this command.

Functional Notes

The type of enrollment procedure is based on the **enrollment** command and its settings. See *enrollment terminal* [on page 509](#) and *enrollment url* [<url> on page 510](#) for more information. This command initiates a dialog that is used to fill in the parameters that make up an enrollment request to be forwarded to a certificate authority. Note that some of the parameters (such as IP address) may be filled in using the values supplied in the **crypto ca profile** (in which case, the enrollment dialog will not prompt for those parameters). Once all required parameters are defined using the dialog, this command assembles them into an enrollment request to be sent to a certificate authority (including the generation of public and private keys). See **crypto ca profile** for more information.

If **enrollment** is set to **terminal**, you may view the request on the terminal screen.

If **enrollment** is set to **url**, the request is sent automatically to the certificate authority using the URL specified by the **enrollment url** command.

Usage Examples

The following example shows a typical enrollment dialog:

ProCurve(config)#**crypto ca enroll MyProfile**

**** Press CTRL+C to exit enrollment request dialog. ****

* Enter signature algorithm (RSA or DSS) [rsa]:**rsa**

* Enter the modulus length to use [512]:**1024**

* Enter the subject name as an X.500 (LDAP) DN:**CN=Router,C=US,L=Huntsville,S=AL**

--The subject name in the certificate will be CN=CN=Router,C=US,L=Huntsville,S=AL.

* Include an IP address in the subject name [n]:**y**

* Enter IP address or name of interface to use:**10.200.1.45**

* Include fully qualified domain name [n]:**y**

* Enter the fully qualified domain name to use:**FullyQualifiedDomainName**

* Include an email address [n]:**y**

* Enter the email address to use:**myemail@email.com**

Generating request (including keys)....

crypto ca import <name> certificate

Use the **crypto ca import certificate** command to import a certificate manually via the console terminal.

Syntax Description

<name>	Specifies CA profile by alphanumeric string of up to 32 characters.

Default Values

No defaults necessary for this command.

Functional Notes

Puts CLI in mode where the certificate can be entered manually. Enter **quit** and a carriage return (or simply enter two consecutive carriage returns) to exit this mode. Abort this mode by pressing **Ctrl-C**. This command only applies if the **enrollment** command is set to **terminal**. See *enrollment terminal* [on page 509](#).

Usage Examples

The following example imports a certificate via the console terminal:

ProCurve(config)#**crypto ca import MyProfile certificate**

Enter the PM-encoded certificate. End with two consecutive carriage returns or the word "quit" on a line by itself:

```
-----BEGIN CERTIFICATE-----
MIIDWTCCAwwOgAwIBAgIKFLCsOgAAAAAAtjANBgkqhkiG9w0BAQUFADBjMQswCQYD
VQQGEwJVUzEQMA4GA1UECBMHQXUxYXNjaFNTETMBEGA1UEBxMKSHVudHN2aWxsZTEa
MBGGA1UEChMRQWR0cmFuVGvjaFN1cHBvcnQxETAPBgNVBAMTCHRzcm91dGVyMB4X
DTAzMDYyNTE0MTM1NV0xDTAzMTIwNjE0NDkxM1owJDEPMA0GA1UEChMGYWR0cmFu
MREwDwYDVQQDEwhNeVJvdXRlcjBcMA0GCsQGSIB3DQEBAQUAA0sAMEgCQQQCIUKqs
fbTalej5m9gk2DMsbC9df3TilBz+7nRx3ZzGw75AQsqEMYeBY5aWi62W59jmxGSE
WX+E8EwBVbZ6JKk5AgMBAAGjggHWMIIIB0jAXBgNVHREEDAOhwQKCgoKggZNeUZx
ZG4wHQYDVR0OBBYEFJAvBRljx1PROnkZ4v0D89yB1eErMIGcBgNVHSMegZQwgZGA
FHGwIRAr11495MgrLNpILzjvrb4JoWekZTBjMQswCQYDVQQGEwJVUzEQMA4GA1UE
CBMHQXUxYXNjaFNTETMBEGA1UEBxMKSHVudHN2aWxsZTEaMBGGA1UEChMRQWR0cmFu
VGvjaFN1cHBvcnQxETAPBgNVBAMTCHRzcm91dGVyghAZqI7OwISgsUhfaSeGh0Ot
MGkGA1UdHwRiMGAwLaAroCmGJ2h0dHA6Ly90c3JvdXRlcj9DZXJ0RW5yb2xsL3Rz
cm91dGVyLmNybDAvoC2gK4YpZmlsZTovL1xcdHNyb3V0ZXJcQ2VydEVucm9sbFcx
c3JvdXRlcj5jcmwwY0GCCsGAQUFBwEBBIBGAMH4wPAYIKwYBBQUHMAKGMGh0dHA6
Ly90c3JvdXRlcj9DZXJ0RW5yb2xsL3Rzcm91dGVyX3Rzcm91dGVyLmNydDA+Bggr
BgEFBQcwAoYyZmlsZTovL1xcdHNyb3V0ZXJcQ2VydEVucm9sbFcx0c3JvdXRlcj90
c3JvdXRlcj5jcnQwDQYJKoZIhvcNAQEFBQADQQBSGD4JbGJGk53qvyy0xXVoMQvy
U8xNjUdvWqjgFOI+2m8ZYJcfhnt11rbP2f3Wm9TpjLe1WuBNxmpNjC9A2ab0
-----END CERTIFICATE-----
```

Success!

crypto ca import <name> crl

Use the **crypto ca import crl** command to import a CRL manually via the console terminal.

Syntax Description

<name>	Specifies CA profile by alphanumeric string of up to 32 characters.

.

Default Values

No defaults necessary for this command.

Functional Notes

Puts CLI in a mode where the CRL can be entered manually. Enter **quit** and a carriage return (or simply enter two consecutive carriage returns) to exit this mode. This command only applies if the **enrollment** command is set to **terminal**. See *enrollment terminal* [on page 509](#).

Usage Examples

The following allows you to manually paste in the CA's CRL:

```
ProCurve(config)#crypto ca import MyProfile crl
```

crypto ca profile <name>

Use the **crypto ca profile** command to define a CA and to enter the CA Profile Configuration. See *CA Profile Configuration Command Set* [on page 505](#) for more information.

Syntax Description

<name>	Specifies CA profile by alphanumeric string of up to 32 characters.

Default Values

No defaults necessary for this command.

Functional Notes

Use this to specify the type of enrollment, as well as enrollment request parameters. See the *Functional Notes* of the command *crypto ca enroll <name>* [on page 256](#) for more information.

Usage Examples

The following example creates the CA profile called **MyProfile** and enters the CA Profile Configuration for that certificate authority:

```
ProCurve(config)#crypto ca profile MyProfile  
Configuring New CA Profile MyProfile.  
ProCurve(ca-profile)#
```

crypto ike

Use the **crypto ike** command to define the system-level local ID for IKE negotiations and to enter the IKE Client or IKE Policy command sets.

Variations of this command include the following:

crypto ike client configuration pool <poolname>

crypto ike local-id address

crypto ike policy <policy priority>

Syntax Description

client configuration pool <poolname>	Creates a local pool named the <poolname> of your choice and enters the IKE Client. Clients that connect via an IKE policy that specifies this pool-name will be assigned values from this pool. See the section for more information.
local-id address	Sets the local ID during IKE negotiation to be the IP address of the interface from which the traffic exits. This setting can be overridden on a per-policy basis using the local-id command in the IKE Policy (see <i>local-id [address asn1-dn fqdn user-fqdn] <ipaddress or name></i> on page 469 for more information).
policy <policy priority>	Creates an IKE policy with the <policy priority> of your choice and enters the IKE Policy. See <i>IKE Policy Command Set</i> on page 462 for more information.

Default Values

There are no default settings for this command.

Usage Examples

The following example creates an IKE policy with a policy priority setting of 1 and enters the IKE Policy for that policy:

```
ProCurve(config)#crypto ike policy 1
ProCurve(config-ike)#
```

Technology Review

The following example configures an Secure Router OS product for VPN using IKE aggressive mode with pre-shared keys. The Secure Router OS product can be set to initiate IKE negotiation in main mode or aggressive mode. The product can be set to respond to IKE negotiation in main mode, aggressive mode, or any mode. In this example, the device is configured to initiate in aggressive mode and to respond to any mode.

This example assumes that the Secure Router OS product has been configured with a WAN IP Address of 63.97.45.57 on interface **ppp 1** and a LAN IP Address of 10.10.10.254 on interface **ethernet 0/1**. The Peer Private IP Subnet is 10.10.20.0.

For more detailed information on VPN configuration, refer to the *VPN Configuration Guide* located on the provided with your unit.

Step 1:

Enter the Global configuration mode (i.e., config terminal mode).

```
ProCurve>enable  
#configure terminal
```

Step 2:

Enable VPN support using the **ip crypto** command. This command allows crypto maps to be applied to interfaces, and enables the IKE server to listen for IKE negotiation sessions on UDP port 500.

```
ProCurve(config)#ip crypto
```

Step 3:

Set the local ID. During IKE negotiation, local-ids are exchanged between the local device and the peer device. In the Secure Router OS, the default setting for all local-ids is configured by the **crypto ike local-id** command. The default setting is for all local-ids to be the IPv4 address of the interface over which the IKE negotiation is occurring. In the future, a unique system-wide Hostname or Fully Qualified Domain Name could be used for all IKE negotiation.

```
ProCurve(config)#crypto ike local-id address
```

Step 4:

Create IKE policy. In order to use IKE negotiation, an IKE policy must be created. Within the system, a list of IKE policies is maintained. Each IKE policy is given a priority number in the system. That priority number defines the position of that IKE policy within the system list. When IKE negotiation is needed, the system searches through the list, starting with the policy with priority of 1, looking for a match to the peer IP address.

An individual IKE policy can override the system local-id setting by having the **local-id** command specified in the IKE policy definition. This command in the IKE policy is used to specify the type of local-id and the local-id data. The type can be of IPv4 address, Fully Qualified Domain Name, or User-Specified Fully Qualified Domain Name.

An IKE policy may specify one or more peer IP addresses that will be allowed to connect to this system. To specify multiple unique peer IP addresses, the **peer A.B.C.D** command is used multiple times within a single IKE policy. To specify that all possible peers can use a default IKE policy, the **peer any** command is given instead of the **peer A.B.C.D** command inside of the IKE policy. The policy with the **peer any** command specified will match to any peer IP address (and therefore should be given the highest numerical priority number). This will make the policy the last one to be compared against during IKE negotiation.

```
ProCurve(config)#crypto ike policy 10  
ProCurve(config-ike)#no local-id  
ProCurve(config-ike)#peer 63.105.15.129  
ProCurve(config-ike)#initiate aggressive
```

```
ProCurve(config-ike)#respond anymode
ProCurve(config-ike)#attribute 10
ProCurve(config-ike-attribute)#encryption 3des
ProCurve(config-ike-attribute)#hash sha
ProCurve(config-ike-attribute)#authentication pre-share
ProCurve(config-ike-attribute)#group 1
ProCurve(config-ike-attribute)#lifetime 86400
```

Step 5:

Define the remote-id settings. The **crypto ike remote-id** command is used to define the remote-id for a peer connecting to the system, specify the preshared-key associated with the specific remote-id, and (optionally) determine that the peer matching this remote-id should not use mode config (by using the **no-mode-config** keyword). See *crypto ike remote-id* [on page 264](#) for more information.

```
ProCurve(config)#crypto ike remote-id address 63.105.15.129 preshared-key
```

mysecret123

Step 6:

Define the transform-set. A transform-set defines the encryption and/or authentication algorithms to be used to secure the data transmitted over the VPN tunnel. Multiple transform-sets may be defined in a system. Once a transform-set is defined, many different crypto maps within the system can reference it. In this example, a transform-set named **highly_secure** has been created. This transform-set defines ESP with Authentication implemented using 3DES encryption and SHA1 authentication.

```
ProCurve(config)#crypto ipsec transform-set highly_secure esp-3des esp-sha-hmac
ProCurve(cfg-crypto-trans)#mode tunnel
```

Step 7:

Define an ip-access list. An Extended Access Control List is used to specify which traffic needs to be sent securely over the VPN tunnel. The entries in the list are defined with respect to the local system. The source IP address will be the source of the traffic to be encrypted. The destination IP address will be the receiver of the data on the other side of the VPN tunnel.

```
ProCurve(config)#ip access-list extended corporate_traffic
ProCurve(config-ext-nacl)#permit ip 10.10.10.0 0.0.0.255 10.10.20.0 0.0.0.255 log
deny ip any any
```

Step 8:

Create crypto map. A Crypto Map is used to define a set of encryption schemes to be used for a given interface. A crypto map entry has a unique index within the crypto map set. The crypto map entry will specify whether IKE is used to generate encryption keys or if manually specified keys will be used. The crypto map entry will also specify who will be terminating the VPN tunnel, as well as which transform-set or sets will be used to encrypt and/or authenticate the traffic on that VPN tunnel. It also specifies the lifetime of all created IPSec Security Associations.

```
ProCurve(config)#crypto map corporate_vpn 1 ipsec-ike
```

```
ProCurve(config-crypto-map)#match address corporate_traffic
ProCurve(config-crypto-map)#set peer 63.105.15.129
ProCurve(config-crypto-map)#set transform-set highly_secure
ProCurve(config-crypto-map)#set security-association lifetime kilobytes 8000
ProCurve(config-crypto-map)#set security-association lifetime seconds 28800
ProCurve(config-crypto-map)#no set pfs
```

Step 9:

Configure public interface. This process includes configuring the IP address for the interface and applying the appropriate crypto map to the interface. Crypto maps are applied to the interface on which encrypted traffic will be transmitted.

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip address 63.97.45.57 255.255.255.248
ProCurve(config-ppp 1)#crypto map corporate_vpn
ProCurve(config-ppp 1)#no shutdown
```

Step 10:

Configure private interface to allow all traffic destined for the VPN tunnel to be routed to the appropriate gateway.

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#ip address 10.10.10.254 255.255.255.0
ProCurve(config-eth 0/1)#no shutdown
ProCurve(config-eth 0/1)#exit
```

crypto ike remote-id

Use the **crypto ike remote-id** command to specify the remote ID and to associate a pre-shared key with the remote ID.

Note *For VPN configuration example scripts, refer to the technical support note **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

address <IPv4 address>	Specifies a remote ID of IPv4 type.
any	Wildcard that allows any remote ID (type and value).
asn1-dn <name>	Specifies an Abstract Syntax Notation Distinguished Name as the remote ID (enter this value in LDAP format).
fqdn <fqdn>	Specifies a fully qualified domain name as the remote ID.
user-fqdn <fqdn>	Specifies a user fully qualified domain name or email address as the remote ID.
preshared-key <keyname>	Associates a pre-shared key with this remote ID.
no-mode-config	Optional keyword used to specify that the peer matching this remote ID should not use mode config.
no-xauth	Optional keyword used to specify that the peer matching this remote ID should not use xauth.
nat-t [v1 v2] [allow force disable]	Optional keyword that denotes whether peers matching this remote ID should allow, disable, or force NAT traversal versions 1 and 2.

Default Values

There are no default settings for this command.

Functional Notes

The **fqdn** and **user-fqdn** <WORD> line can include wildcard characters. The wildcard characters are "*" for a 0 or more character match and "?" for a single character match. Currently, the "?" cannot be set up using the CLI, but it can be transferred to the unit via the startup-config.

Example for **user-fqdn**:

john*@domain.com

will match:

johndoe@domain.com

johnjohn@myemail.com

john@myemail.com

Example for **fqdn**:

***.domain.com**

will match:

www.domain.com

ftp.domain.com

one.www.domain.com

The **address** remote ID can be in the form of a single host address or in the form of an IP address wildcard.

Example for **address** type:

crypto ike remote id address 10.10.10.0 0.0.0.255

will match:

10.10.10.1

10.10.10.2

and all IP addresses in the form of 10.10.10.X (where X is 0-255)

The **asn1-dn** <WORD> line can include wildcard characters. The wildcard characters are "*" for a 0 or more character match and "?" for a single character match. Currently, the "?" cannot be set up using the CLI, but it can be transferred to the unit via the startup-config.

Example for typical **asn1-dn** format with no wildcards:

crypto ike remote-id asn1-dn "CN=MyRouter, C=US, S=CA, L=Roseville, O=HP, OU=TechSupport"

(matches only remote ID strings with all fields exactly the same)

Example for typical **asn1-dn** format with wildcards used to match a string within a field:

crypto ike remote-id asn1-dn "CN=*, C=*, S=*, L=*, O=*, OU=*"

(matches any asn1-dn remote ID string from a peer)

Example for typical **asn1-dn** format with wildcards used to match a portion of the remote ID:

crypto ike remote-id asn1-dn "CN=*, C=US, S=CA, L=Roseville, O=HP, OU=*"

(matches any remote ID string with the same values for the C, S, L, and O fields, and any values in the CN and OU fields)

Example for typical **asn1-dn** format with wildcards used to match a portion of a field:

crypto ike remote-id asn1-dn "CN=My*, C=US, S=CA, L=Roseville, O=HP, OU=TechSupport"

(matches remote ID strings with all fields exactly the same, but with any CN field beginning with "My")

Usage Examples

The following example assigns a remote ID of 63.97.45.57 and associates the pre-shared key **mysecret** with the remote ID:

ProCurve(config)#**crypto ike remote-id address 63.97.45.57 preshared-key mysecret**

crypto ipsec transform-set <setname> <parameters>

Use the **crypto ipsec transform-set** command to define the transform configuration for securing data (e.g., esp-3des, esp-sha-hmac, etc.). The transform-set is then assigned to a crypto map using the map's **set transform-set** command. See *set transform-set <setname1 - setname6>* [on page 491](#).

Note *For VPN configuration example scripts, refer to the technical support note **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<setname>	Specifies a name for the transform-set you are about to define.
<parameters>	Assigns a combination of up to three security algorithms. This field is a valid combination of the following: • ah-md5-hmac, ah-sha-hmac • esp-des, esp-3des, esp-aes-128-cbc, esp-aes-192-cbc, esp-aes-256-cbc, esp-null • esp-md5-hmac, esp-sha-hmac

Default Values

There are no default settings for this command.

Functional Notes

Crypto map entries do not directly contain the transform configuration for securing data. Instead, the crypto map is associated with transform sets which contain specific security algorithms.

If no transform-set is configured for a crypto map, the entry is incomplete and will have no effect on the system.

Usage Examples

The following example first creates a transform-set (**Set1**) consisting of two security algorithms (up to three may be defined), and then assigns the transform-set to a crypto map (**Map1**):

```
ProCurve(config)#crypto ipsec transform-set Set1 esp-3des esp-sha-hmac
ProCurve(cfg-crypto-trans)#exit
```

```
ProCurve(config)#crypto map Map1 1 ipsec-ike
ProCurve(config-crypto-map)#set transform-set Set1
```

crypto map

Use the **crypto map** command to define crypto map names and numbers and to enter the associated (either Crypto Map IKE or Crypto Map Manual).

Variations of this command include the following:

```
crypto map <mapname> <mapindex> ipsec-ike
crypto map <mapname> <mapindex> ipsec-manual
```

Note *For VPN configuration example scripts, refer to the technical support note **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<mapname>	Names the crypto map. You can assign the same name to multiple crypto maps, as long as the map index numbers are unique.
<mapindex>	Assigns a crypto map sequence number.
ipsec-ike	Specifies the Crypto Map IKE (see <i>Crypto Map IKE Command Set</i> on page 483). This supports IPsec entries that will use IKE to negotiate keys.
ipsec-manual	Specifies the Crypto Map Manual (see <i>Crypto Map IKE Command Set</i> on page 483). This supports manually configured IPsec entries.

Default Values

There are no default settings for this command.

Functional Notes

Crypto map entries do not directly contain the transform configuration for securing data. Instead, the crypto map is associated with transform sets which contain specific security algorithms (see *crypto ipsec transform-set <setname> <parameters>* [on page 266](#)).

Crypto map entries do not directly contain the selectors used to determine which data to secure. Instead, the crypto map entry refers to an access control list. An access control list is assigned to the crypto map using the **match address** command (see *ike-policy <policy number>* [on page 485](#)).

If no transform-set or access-list is configured for a crypto map, the entry is incomplete and will have no effect on the system.

When you apply a crypto map to an interface (using the **crypto map** command within the interface's), you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.

Technology Review

A crypto map entry is a single policy that describes how certain traffic is to be secured. There are two types of crypto map entries: ipsec-manual and ipsec-ike. Each entry is given an index, which is used to sort the ordered list. When a non-secured packet arrives on an interface, the crypto map set associated with that interface is processed in order. If a crypto map entry matches the non-secured traffic, the traffic is discarded.

When a packet is to be transmitted on an interface, the crypto map set associated with that interface is processed in order. The first crypto map entry that matches the packet will be used to secure the packet. If a suitable SA (security association) exists, that is used for transmission. Otherwise, IKE is used to establish an SA with the peer. If no SA exists, and the crypto map entry is "respond only", the packet is discarded.

When a secured packet arrives on an interface, its SPI (security parameter index) is used to look up an SA. If an SA does not exist, or if the packet fails any of the security checks (bad authentication, traffic does not match SA selectors, etc.), it is discarded. If all checks pass, the packet is forwarded normally.

Usage Examples

The following example creates a new IPSec IKE crypto map called **testMap** with a map index of **10**:

```
ProCurve(config)#crypto map testMap 10 ipsec-ike
ProCurve(config-crypto-map)#
```

enable password [md5] <password>

Use the **enable password** command to define a password (with optional encryption) for accessing the Enable mode. Use the **no enable password** command to remove a configured password.

Note *To prevent unauthorized users from accessing the configuration functions of your device, immediately install an Enable-level password.*

Syntax Description

md5	Optional. Specifies Message Digest 5 (md5) as the encryption protocol to use when displaying the enable password during show commands. If the md5 keyword is not used, encryption is not used when displaying the enable password during show commands
<password>	String (up to 30 characters in length) to use as the Enable Security mode password.

Default Values

By default, there is no configured enable password.

Usage Examples

To provide extra security, the Secure Router OS can encrypt the enable password when displaying the current configuration. For example, the following is a **show configuration** printout (password portion) with an unencrypted enable password (PASSWORD):

!

enable password PASSWORD

!

Alternately, the following is a **show configuration** printout (password portion) with an enable password of password using md5 encryption:

!

enable password md5 encrypted 5aa5fbae7d01a90e79fb57705ce74676

!

event-history on

Use the **event-history on** command to enable event logging for the Secure Router OS system. Event log messages will not be recorded unless this command has been issued (regardless of the **event-history priority** configured). The event log may be displayed using the **show event-history** command. Use the **no** form of this command to disable the event log.

Syntax Description

No subcommands.

Default Values

By default, the Secure Router OS event logging capabilities are disabled.

Functional Notes

The event history provides useful information regarding the status of the system and individual port states. Use the event history as a troubleshooting tool when identifying system issues. The following is a sample event history log.

ProCurve#show event-history

Using 526 bytes

2002.07.12 15:34:01 T1.t1 1/1 Yellow

2002.07.12 15:34:01 INTERFACE_STATUS.t1 1/1 changed state to down.

2002.07.12 15:34:02 T1.t1 1/1 No Alarms

2002.07.12 15:34:02 INTERFACE_STATUS.t1 1/1 changed state to up.

2002.07.12 15:34:03 INTERFACE_STATUS.eth 0/1 changed state to up.

2002.07.12 15:34:10 OPERATING_SYSTEM Warm Start

2002.07.12 15:34:12 PPP.NEGOTIATION LCP up

2002.07.12 15:34:12 PPP.NEGOTIATION IPCP up

Usage Examples

The following example enables the Secure Router OS event logging feature:

ProCurve(config)#**event-history on**

event-history priority [error | fatal | info | notice | warning]

Use the **event-history priority** command to set the threshold for events stored in the event history. All events with the specified priority or higher will be kept for viewing in the local event log. The event log may be displayed using the **show event-history** command. Use the **no** form of this command to keep specified priorities from being logged.

Syntax Description

Sets the minimum priority threshold for logging messages to the event history. The following priorities are available (ranking from lowest to highest):

Error

When selected, events with **error** and **fatal** priorities are logged.

Fatal

When selected, only events with a **fatal** priority are logged.

Info

When selected, all events are logged.

Notice

When selected, events with **notice**, **warning**, **error**, and **fatal** priorities are logged.

Warning

When selected, events with **warning**, **error**, and **fatal** priorities are logged.

Default Values

By default, no event messages are logged to the event history.

Functional Notes

The event history provides useful information regarding the status of the system and individual port states. Use the event history as a troubleshooting tool when identifying system issues. The following is a sample event history log.

ProCurve#show event-history

Using 526 bytes

2002.07.12 15:34:01 T1.t1 1/1 Yellow

2002.07.12 15:34:01 INTERFACE_STATUS.t1 1/1 changed state to down.

2002.07.12 15:34:02 T1.t1 1/1 No Alarms

2002.07.12 15:34:02 INTERFACE_STATUS.t1 1/1 changed state to up.

2002.07.12 15:34:03 INTERFACE_STATUS.eth 0/1 changed state to up.

2002.07.12 15:34:10 OPERATING_SYSTEM Warm Start

2002.07.12 15:34:12 PPP.NEGOTIATION LCP up

2002.07.12 15:34:12 PPP.NEGOTIATION IPCP up

Usage Examples

The following example logs all events to the event history:

```
ProCurve>enable
```

```
ProCurve#config terminal
```

```
ProCurve(config)#event-history priority info
```


exception report [file-name <filename>]

Use the **exception report** command to change the default output filename for generated exception reports. An exception report contains a snap shot of current system processes to use when troubleshooting.

Syntax Description

file-name <filename>	Optional. Specifies a filename for the exception report other than the default filename. The specified filename replaces the exception-report in the default filename.

Default Values

By default, the exception report filename is **exception-report-yyyyMMddHHmmss**. (The yyyyMMddHHmmss is automatically populated with the actual year, month, day, hour, minutes, and seconds when the report was generated.) Specifying a new filename results in the following format: **filename-yyyyMMddHHmmss**.

Usage Example

The following example specifies the output filename for an exception report:

```
ProCurve(config)#exception report file-name thereport
```

```
ProCurve(config)#exit
```

```
ProCurve#exception report generate
```

```
Exception report generated.
```

```
ProCurve#sh flash
```

```
1744 startup-config
```

```
45676 thereport-20050708080537
```

```
ProCurve#config t
```

```
ProCurve(config)#no exception report file-name
```

```
ProCurve(config)#exit
```

```
Appropriate commands must be issued to preserve configuration.
```

```
ProCurve#exception report generate
```

```
Exception report generated.
```

```
ProCurve#sh flash
```

```
1744 startup-config
```

```
45676 thereport-20050708080537
```

```
45900 exception-report-20050708080552
```

ftp authentication <listname>

Use the **ftp authentication** command to attach AAA login authentication lists to the FTP server (see *aaa authentication login <listname> [none | line | enable | local | group]* [on page 241](#) for more information). This list is only used if the AAA subsystem has been activated with the **aaa on** command.

Syntax Description

<listname>	Specifies the named list created with the aaa authentication login command. Enter default to use the AAA default login list.

Default Values

There is no default configuration for the list. If AAA is turned on but no **ftp authentication** list has been assigned, FTP denies all login attempts.

Usage Examples

The following example attaches the authentication list, **MyList**, to the FTP server:

```
ProCurve(config)#ftp authentication MyList
```

The following example specifies that the Secure Router OS use the default AAA login list for FTP authentication:

```
ProCurve(config)#ftp authentication default
```

hostname <name>

Creates a name used to identify the unit. This alphanumeric string should be used as a unique description for the unit. This string will be displayed in all prompts.

Syntax Description

<name>	Identifies the unit by alphanumeric string of up to 32 characters.

Default Values

<name>	Router

Usage Examples

The following example creates a hostname for the Secure Router OS device of **ATL_RTR** to identify the system as the Atlanta router:

```
ProCurve(config)#hostname ATL_RTR
```

interface <port-type> <slot/port>

Activates the Interface Configuration mode for the listed physical interface.

Syntax Description

<port-type>	Identifies the physical port type of the installed Interface Module, Backup Module or Ethernet port. Type interface ? for a complete list of valid interfaces.
<slot/port>	Specifies an interface based on its physical location (slot and port). For example, if you have a T1/DSX-1 installed in Slot 1 of an Secure Router OS product: • The WAN-T1 port would be specified in the CLI as t1 1/1. • The DSX-1 port would be specified as t1 1/2. • If (for example) a backup module is also installed, then the DBU port would be specified as bri 1/3. • If you are specifying a port that is built into the base unit (e.g., the Ethernet port), the slot number is 0. For example, the Ethernet (LAN) port would be specified as ethernet 0/1.

Default Values

No default values required for this command.

Usage Examples

The following example enters the serial interface for a serial module installed in slot 1:

```
ProCurve(config)#interface serial 1/1
```

interface frame-relay <label/> point-to-point

Use the **interface frame-relay** command to create a virtual Frame Relay interface (or sublink, if specified) that is identified using the entered number label. In addition, entering this command activates the Frame Relay interface. The **point-to-point** keyword (optional) can be used to identify the Frame Relay endpoint as a point-to-point link (versus multipoint). Use the **no** form of this command to delete a configured virtual Frame Relay interface.

To specify a virtual Frame Relay sub-interface, the following syntax applies:

interface frame-relay <label>.<sublink label>

Syntax Description

<label>	Specifies the numerical virtual Frame Relay interface identifying label (valid range: 1 to 1024)
<sublink label>	Numerical label for the virtual sublink (valid range: 1-255)
point-to-point	Optional. Identifies the Frame Relay interface as a point-to-point link (versus multilink) By default, all created Frame Relay interfaces are point-to-point.

Default Values

By default, there are no configured virtual Frame Relay interfaces or sublinks.

Functional Notes

Creating an endpoint that uses a layer 2 protocol (such as Frame Relay) is generally a four-step process:

Step 1:

Create the Frame Relay virtual endpoint (using the **interface frame-relay** command) and set the signaling method (using the **frame-relay lmi-type** command). Also included in the Frame Relay virtual endpoint are all the applicable Frame Relay timers, logging thresholds, encapsulation types, etc. Generally, most Frame Relay virtual interface parameters should be left at their default state. For example, the following creates a Frame Relay interface labeled **7** and sets the signaling method to **ansi**.

```
ProCurve(config)#interface frame-relay 7
```

```
ProCurve(config-fr 7)#frame-relay lmi-type ansi
```

Step 2:

Create the sub-interface and configure the PVC parameters. Using the sub-interface, apply access policies to the interface, create bridging interfaces, configure backup, assign an IP address, and set the PVC data-link control identifier (DLCI). For example, the following creates a Frame Relay sub-interface labeled **22**, sets the DLCI to **30**, and assigns an IP address of **193.44.69.1/30** to the interface.

```
ProCurve(config-fr 7)#interface fr 7.22
```

```
ProCurve(config-fr 7.22)#frame-relay interface-dlci 30
```

```
ProCurve(config-fr 7.22)#ip address 193.44.69.1 255.255.255.252
```

Step 3: (VALID ONLY FOR T1 INTERFACES)

Specify the group of DS0s used for signaling on the T1 interface by creating a **tdm-group**. Group any number of aggregate DS0s together to create a data pipe for layer 2 signaling. Also use the **tdm-group** command to specify the per-DS0 signaling rate on the interface. For example, the following creates a tdm-group labeled **9** containing 20 DS0s (each DS0 having a data rate of 56 kbps).

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#tdm-group 9 timeslots 1-20 speed 56  
ProCurve(config-t1 1/1)#exit
```

Step 4:

Make the association between the layer 2 endpoint and the physical interface using the **bind** command. Supported layer 2 protocols include Frame Relay and point-to-point protocol (PPP). For example, the following creates a bind (labeled **5**) to make an association between the Frame Relay virtual interface (**fr 7**) and the tdm-group configured on interface t1 1/1 (**tdm-group 9**).

```
ProCurve(config)#bind 5 t1 1/1 9 fr 7
```

Usage Examples

The following example creates a Frame Relay virtual interface (labeled **1**) and enters the Frame Relay Interface Configuration mode:

```
ProCurve(config)#interface fr 1  
ProCurve(config-fr 1)#
```

interface hdlc <label>

Use the **interface hdlc** command to create a virtual high level data link control interface that is identified using the entered number label. In addition, entering this command activates the HDLC interface. Use the **no** form of this command to delete a configured virtual HDLC interface.

Syntax Description

<label>	Specifies the numerical virtual HDLC interface identifying label (valid range: 1 to 1024)

Default Values

By default, there are no configured HDLC interfaces.

Functional Notes

Creating an endpoint that uses a layer 2 protocol (such as HDLC) is generally a four-step process:

Step 1:

Create the HDLC virtual endpoint (using the **interface hdlc**) command and enter the HDLC configuration commands.

```
ProCurve(config)#interface hdlc 7
```

```
ProCurve(config-hdlc 7)#
```

Step 2:

Configure the interface parameters to apply access policies to the interface, create bridging interfaces, configure backup, and assign an IP address. For example, the following assigns an IP address of **193.44.69.1/30** to the interface.

```
ProCurve(config-hdlc 7)#ip address 193.44.69.1 255.255.255.252
```

Step 3: (VALID ONLY FOR T1 INTERFACES)

Specify the group of DS0s used for signaling on the T1 interface by creating a **tdm-group**. Group any number of aggregate DS0s together to create a data pipe for layer 2 signaling. Also use the **tdm-group** command to specify the per-DS0 signaling rate on the interface. For example, the following creates a tdm-group labeled **9** containing 20 DS0s (each DS0 having a data rate of 56 kbps).

```
ProCurve(config)#interface t1 1/1
```

```
ProCurve(config-t1 1/1)#tdm-group 9 timeslots 1-20 speed 56
```

```
ProCurve(config-t1 1/1)#exit
```

Step 4:

Make the association between the layer 2 endpoint and the physical interface using the **bind** command. Supported layer 2 protocols include Frame Relay, point-to-point protocol (PPP), and HDLC. For example, the following creates a bind (labeled **5**) to make an association between the HDLC virtual interface (**hdlc 7**)

and the tdm-group configured on interface t1 1/1 (**tdm-group 9**).

```
ProCurve(config)#bind 5 t1 1/1 9 hdlc 7
```

Usage Examples

The following example creates a HDLC virtual interface (labeled **1**) and enters the HDLC Interface Configuration mode:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#
```


interface loopback <label>

Use the **interface loopback** command to create a virtual interface that can be assigned layer 3 and higher properties and is always up unless the router is shut down. Use the **no** form of this command to delete a configured loopback interface.

Syntax Description

<label>	Specifies the numerical virtual loopback interface identifying label (valid range: 1 to 1024)

Default Values

By default, there are no configured loopback interfaces.

Usage Examples

The following example creates a loopback virtual interface (labeled 1) and enters the Loopback Interface Configuration mode:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#
```

interface ppp <label>

Use the **interface ppp** command to create a virtual point-to-point protocol (PPP) interface that is identified using the entered number label. In addition, entering this command activates the PPP interface. Use the **no** form of this command to delete a configured virtual PPP interface.

Syntax Description

<label>	Specifies the numerical virtual PPP interface identifying label (range: 1 to 1024).

Default Values

By default, there are no configured PPP interfaces.

Functional Notes

Creating an endpoint that uses a layer 2 protocol (such as PPP) is generally a four-step process:

Step 1:

Create the PPP virtual endpoint (using the **interface ppp**) command and enter the PPP.

```
ProCurve(config)#interface ppp 7
```

```
ProCurve(config-ppp 7)#
```

Step 2:

Configure the interface parameters to apply access policies to the interface, create bridging interfaces, configure backup, and assign an IP address. For example, the following assigns an IP address of **193.44.69.1/30** to the interface.

```
ProCurve(config-ppp 7)#ip address 193.44.69.1 255.255.255.252
```

Step 3: (VALID ONLY FOR T1 INTERFACES)

Specify the group of DS0s used for signaling on the T1 interface by creating a **tdm-group**. Group any number of aggregate DS0s together to create a data pipe for layer 2 signaling. Also use the **tdm-group** command to specify the per-DS0 signaling rate on the interface. For example, the following creates a tdm-group labeled **9** containing 20 DS0s (each DS0 having a data rate of 56 kbps).

```
ProCurve(config)#interface t1 1/1
```

```
ProCurve(config-t1 1/1)#tdm-group 9 timeslots 1-20 speed 56
```

```
ProCurve(config-t1 1/1)#exit
```

Step 4:

Make the association between the layer 2 endpoint and the physical interface using the **bind** command. Supported layer 2 protocols include Frame Relay and point-to-point protocol (PPP). For example, the following creates a bind (labeled **5**) to make an association between the PPP virtual interface (**ppp 7**) and the tdm-group configured on interface t1 1/1 (**tdm-group 9**).

```
ProCurve(config)#bind 5 t1 1/1 9 ppp 7
```

Usage Examples

The following example creates a PPP virtual interface (labeled **1**) and enters the PPP Interface Configuration mode:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#
```

interface tunnel <id>

Use the **interface tunnel** command to create a virtual tunnel interface and enters the Tunnel Configuration command set. See *Tunnel Configuration Command Set* [on page 800](#) for details. Use the **no** form of this command to delete a configured virtual tunnel interface.

Syntax Description

<id>	Specifies the numerical tunnel interface identifying label (valid range: 1 to 1024).

Default Values

By default, there are no configured tunnel interfaces.

Functional Notes

A tunnel may become operational only under the following conditions:

1. The tunnel must have an IP address defined.
2. A valid source address or interface must be configured.
3. A valid destination address must be configured.
4. The physical interface used as the source for the tunnel must be operational.
5. The tunnel can not be in a recursive routing loop.
6. If keepalives are enabled, keepalive processing must be successful. See *keepalive <period> <retries>* [on page 840](#) for details.

Technology Review

A tunnel interface enables standard point-to-point encapsulation between two links. Each endpoint must have a unique tunnel configured. Tunneling allows an arbitrary payload protocol to be encapsulated within a delivery protocol to provide point-to-point communications. The tunnel alone does not provide encryption or any other means of high security. The tunnel interface is not a physical interface, so traffic will be routed to the tunnel by the routing engine for encapsulation or decapsulation and typically forwarded out a physical interface. A common tunnel implementation is the use of a GRE tunnel to transport IP multicast traffic, such as that used by routing protocols across a link that only has IP unicast connectivity (such as IPSec).

Usage Examples

The following example creates a tunnel interface (labeled 1) and enters the Tunnel Configuration mode:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#
```

ip access-list extended <listname>

Use the **ip access-list extended** command to create an empty access list and enter the extended access-list. Use the **no** form of this command to delete an access list and all the entries contained in it.

The following lists the complete syntax for the **ip access-list extended** commands:

<action> <protocol> <source IP> <source port> <destination ip> <destination port>

Example:

Source IP Address

[**permit** | **deny**] [**ip** | **tcp** | **udp**] [**any** | **host** <A.B.C.D> | <A.B.C.D> <W.W.W.W>]
 <source port>* [**any** | **host** <A.B.C.D> | <A.B.C.D> <W.W.W.W>] <destination port>*

Destination IP Address

Example:

Source IP Address

[**permit** | **deny** **icmp**] [**any** | **host** <A.B.C.D> | <A.B.C.D> <W.W.W.W>]
 [**any** | **host** <A.B.C.D> | <A.B.C.D> <W.W.W.W>] <icmp-type>* <icmp-code>* <icmp-message>*

Destination IP Address

* = optional

Syntax Description

<listname>	Alphanumeric descriptor for identifying the configured access list (all access list descriptors are case-sensitive)
<protocol>	Specifies the data protocol such as ip, icmp, tcp, udp, or a specific protocol (0-255)
<source ip>	Specifies the source IP address used for packet matching

IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range". Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care". For example, entering **deny 192.168.0.0 0.0.0.255** will deny all traffic from the 192.168.0.0/24 network.

Syntax Description (Continued)

<source port>	Optional. The source port is used only when <i><protocol></i> is tcp or udp																				
	The following keywords and port numbers are supported for the <i><source port></i> field: any Match any destination port eq <port number> Match only packets on a given port number gt <port number> Match only packets with a port number higher than the one listed host <port number> Match a single destination host lt <port number> Match only packets with a port number lower than the one listed neq <port number> Match only packets that do not contain the specified port number range <port number> Match only packets that contain a port number specified in the listed range The <i><port number></i> may be specified using the following syntax: <0-65535>. Specifies the port number used by TCP or UDP to pass information to upper layers. All ports below 1024 are considered well-known ports and are controlled by the Internet Assigned Numbers Authority (IANA). All ports above 1024 are dynamically assigned ports that include registered ports for vendor-specific applications <port list> The Secure Router OS provides a condensed list of port numbers that may be entered using a text name The following is the list of UDP port numbers that may be identified using the text name (in bold): <table> <tr> <td>biff (Port 512)</td><td>ntp (Port 123)</td></tr> <tr> <td>bootpc (Port 68)</td><td>pim-auto-rp (496)</td></tr> <tr> <td>bootps(Port 67)</td><td>rip (Port 520)</td></tr> <tr> <td>discard (Port 9)</td><td>snmp (Port 161)</td></tr> <tr> <td>dnsix (Port 195)</td><td>snmptrap (Port 162)</td></tr> <tr> <td>domain (Port 53)</td><td>sunrpc (Port 111)</td></tr> <tr> <td>echo (Port 7)</td><td>syslog (Port 514)</td></tr> <tr> <td>isakmp (Port 500)</td><td>tacacs (Port 49)</td></tr> <tr> <td>mobile-ip (Port 434)</td><td>talk (Port 517)</td></tr> <tr> <td>nameserver (Port 42)</td><td>tftp (Port 69)</td></tr> </table>	biff (Port 512)	ntp (Port 123)	bootpc (Port 68)	pim-auto-rp (496)	bootps (Port 67)	rip (Port 520)	discard (Port 9)	snmp (Port 161)	dnsix (Port 195)	snmptrap (Port 162)	domain (Port 53)	sunrpc (Port 111)	echo (Port 7)	syslog (Port 514)	isakmp (Port 500)	tacacs (Port 49)	mobile-ip (Port 434)	talk (Port 517)	nameserver (Port 42)	tftp (Port 69)
biff (Port 512)	ntp (Port 123)																				
bootpc (Port 68)	pim-auto-rp (496)																				
bootps (Port 67)	rip (Port 520)																				
discard (Port 9)	snmp (Port 161)																				
dnsix (Port 195)	snmptrap (Port 162)																				
domain (Port 53)	sunrpc (Port 111)																				
echo (Port 7)	syslog (Port 514)																				
isakmp (Port 500)	tacacs (Port 49)																				
mobile-ip (Port 434)	talk (Port 517)																				
nameserver (Port 42)	tftp (Port 69)																				

Syntax Description (Continued)

netbios-dgm (Port 138)	time (Port 37)
netbios-ns (Port 137)	who (Port 513)
netbios-ss (Port 139)	xdmcp (Port 177)

The following is the list of TCP port numbers that may be identified using the text name (in **bold**):

bgp (Port 179)	lpd (Port 515)
chargen (Port 19)	nntp (Port 119)
cmd (Port 514)	pim-auto-rp (Port 496)
daytime (Port 13)	pop2 (Port 109)
discard (Port 9)	pop3 (Port 110)
domain (Port 53)	smtp (Port 25)
echo (Port 7)	sunrpc (Port 111)
exec (Port 512)	syslog (Port 514)
finger (Port 79)	tacacs (Port 49)
ftp (Port 21)	talk (Port 517)
gopher (Port 70)	tftp (Port 69)
hostname (Port 101)	telnet (Port 23)
ident (Port 113)	time (Port 37)
irc (Port 194)	uucp (Port 540)
klogin (Port 543)	whois (Port 43)
kshell (Port 544)	www (Port 80)
login (Port 513)	

<destination ip>

Specifies the destination IP address used for packet matching

IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range". Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care". For example, entering **deny 192.168.0.0 0.0.0.255** will deny all traffic from the 192.168.0.0/24 network.

Syntax Description (Continued)

<i><destination port></i>	Optional. Only valid when <i><protocol></i> is tcp or udp (See previously listed <i><source port></i> for more details)
<i><icmp-type></i>	Optional. Filter packets using ICMP defined (and numbered) messages carried in IP datagrams (used to send error and control information). Valid range is 0 to 255.
<i><icmp-code></i>	Optional. ICMP packets that are filtered using the ICMP message type (using the <i><icmp-type></i> keyword) may also be filtered using the ICMP message code (valid range: 0 to 255).
<i><icmp-message></i>	An <i><icmp-type></i> must be specified when entering an <i><icmp-code></i>. Optional. Filter packets using ICMP descriptive message rather than the corresponding type and code associations.

Default Values

By default, all Secure Router OS security features are disabled and there are no configured access lists.

Functional Notes

Access control lists (ACLs) are used as packet selectors by other Secure Router OS systems; by themselves they do nothing. ACLs are composed of an ordered list of entries with an implicit **deny all** at the end of each list. An ACL entry contains two parts: an action (permit or deny) and a packet pattern. A permit ACL is used to allow packets (meeting the specified pattern) to enter the router system. A deny ACL advances the Secure Router OS to the next access policy entry. The Secure Router OS provides two types of ACLs: standard and extended. Standard ACLs allow source IP address packet patterns only. Extended ACLs may specify patterns using most fields in the IP header and the TCP or UDP header.

ACLs are performed in order from the top of the list down. Generally, the most specific entries should be at the top and the most general at the bottom.

The following commands are contained in the access-list extended:

remark

Use the remark command to associate a descriptive tag (up to 80 alphanumeric characters encased in quotation marks) to the access-list. Enter a functional description for the list such as "This list blocks all outbound web traffic".

log

Using the log keyword logs a message (if debug access-list is enabled for this access list) when the access list finds a packet match.

Usage Examples

The following example creates an access list **AllowIKE** to allow all IKE (UDP Port 500) packets from the 190.72.22.55.0/24 network:

```
ProCurve(config)#ip access-list extended AllowIKE
ProCurve(config-ext-nacl)#permit udp 190.72.22.55.0 0.0.0.255 eq 500 any eq 500
```

For more details, refer to the *ProCurve SROS Documentation* CD for technical support notes regarding access-list configuration.

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access control list (using the **ip access-list** command) to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range". Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care". For example, entering **deny 192.168.0.0 0.0.0.255** will deny all traffic from the 192.168.0.0/24 network.

Step 3:

Create an access control policy (using the **ip policy-class** command) that uses a configured access list. Secure Router OS access policies are used to allow, discard, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list <access list names> **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet or the Web GUI.

discard list <access list names> **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list <access list names> **address** <IP address> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list <access list names> **interface** <interface> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list <access list names> **address** <IP address>

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Caution	<i>Before applying an access control policy to an interface, verify your Telnet connection will not be affected by the policy. If a policy is applied to the interface you are connecting through and it does not allow Telnet traffic, your connection will be lost.</i>

Step 4:

Apply the created access control policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy** <policy name>. The following example assigns access policy **MatchAll** to the Ethernet 0/1 interface:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#access-policy MatchAll
```

ip access-list standard <listname> [permit | deny] <ip address>

Use the **ip access-list standard** command to create an empty access list and enter the standard access-list. Use the **no** form of this command to delete an access list and all the entries contained in it.

The following lists the complete syntax for the **ip access-list standard** commands:

ip access-list standard <listname> [permit | deny] any [permit | deny] host <ip address> [permit | deny] <ip address> <wildcard>

Syntax Description

<listname>	Identifies the configured access list by alphanumeric descriptor (all access list descriptors are case-sensitive).
[permit deny]	Permits or denies entry to the routing system for specified packets.
<ip address>	Specifies the source IP address used for packet matching.
IP addresses can be expressed in one of three ways: 1. Using the keyword any to match any IP address. For example, entering deny any will effectively shut down the interface that uses the access list because all traffic will match the any keyword. 2. Using the host <A.B.C.D> to specify a single host address. For example, entering permit host 196.173.22.253 will allow all traffic from the host with an IP address of 196.173.22.253. 3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range". Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care". For example, entering deny 192.168.0.0 0.0.0.255 will deny all traffic from the 192.168.0.0/24 network.	

Default Values

By default, all Secure Router OS security features are disabled and there are no configured access lists.

Functional Notes

Access control lists are used as packet selectors by access policies (ACPs); by themselves they do nothing. ACLs are composed of an ordered list of entries with an implicit **deny all** at the end of each list. An ACL entry contains two parts: an action (permit or deny) and a packet pattern. A permit ACL is used to allow packets (meeting the specified pattern) to enter the router system. A deny ACL advances the Secure Router OS to the next access policy entry. The Secure Router OS provides two types of ACLs: standard and extended. Standard ACLs allow source IP address packet patterns only. Extended ACLs may specify patterns using most fields in the IP header and the TCP or UDP header.

ACLs are performed in order from the top of the list down. Generally the most specific entries should be at the top and the most general at the bottom.

The following commands are contained in the **access-list standard**:

remark

Use the remark command to associate a descriptive tag (up to 80 alphanumeric characters encased in quotation marks) to the access-list. Enter a functional description for the list such as "This list blocks all outbound web traffic".

log

use the log keyword to log a message (if debug access-list is enabled for this access list) when the access list finds a packet match.

permit or deny any

Use the any keyword to match any IP address received by the access list. For example, the following allows all packets through the configured access list:

```
ProCurve(config)#ip access-list standard MatchAll
ProCurve(config-std-nacl)#permit any
```

permit or deny host <ip address>

Use the host <A.B.C.D> keyword to specify a single host address. For example, the following allows all traffic from the host with an IP address of 196.173.22.253.

```
ProCurve(config)#ip access-list standard MatchHost
ProCurve(config-std-nacl)#permit host 196.173.22.253
```

permit or deny <ip address> <wildcard>

Use the <A.B.C.D> <wildcard> format to match all IP addresses in a "range". Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care". For example, the following denies all traffic from the 192.168.0.0/24 network:

```
ProCurve(config)#ip access-list standard MatchNetwork
ProCurve(config-std-nacl)#deny 192.168.0.0 0.0.0.255
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a “range”. Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a “don’t care”. For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an access policy that uses a configured access list. Secure Router OS access policies are used to allow, discard, or manipulate (using **NAT**) data for each physical interface. Each ACP consists of a selector (**access list**) and an action (**allow**, **discard**, **NAT**). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list <access list names> **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet or the Web GUI.

discard list <access list names> **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list <access list names> **address** <IP address> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list <access list names> **interface** <interface> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list <access list names> **address** <IP address>

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Caution

Before applying an access control policy to an interface, verify your Telnet connection will not be affected by the policy. If a policy is applied to the interface you are connecting through and it does not allow Telnet traffic, your connection will be lost.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy** <policy name>. The following example assigns access policy **MatchAll** to the Ethernet 0/1 interface:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#access-policy MatchAll
```

Usage Examples

The following example creates an access list **UnTrusted** to deny all packets from the 190.72.22.248/30 network:

```
ProCurve(config)#ip access-list standard UnTrusted
```

```
ProCurve(config-std-nacl)#deny 190.72.22.248 0.0.0.3
```

For more details, refer to the *ProCurve SROS Documentation CD* for technical support notes regarding access-list configuration.

ip as-path-list <listname>

Use the **ip as-path-list** command to create IP autonomous system (AS) path lists for route-map use. Use the **no** form of this command to delete the AS path list.

Syntax Description

<listname>	Specifies the name of the AS-path list. See <i>AS Path List Command Set</i> on page 992 for more information of the available options.

Default Values

By default, no as-path lists are defined.

Functional Notes

AS path lists define a list of AS specifications that once created, may then be referenced in a route map. See the Usage Examples section below.

Usage Examples

The following example creates the AS path list **list5** and enters the IP **as-path-list** command mode:

```
ProCurve(config)#ip as-path-list list5  
ProCurve(config-as-path-list)#
```

ip classless

Use the **ip classless** command to forward classless packets to the best supernet route available. A classless packet is a packet addressed for delivery to a subnet of a network with no default network route.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Functional Notes

Secure Router OS products only function in classless mode. You cannot disable this feature.

Usage Examples

The following example enables the system to forward classless packets:

```
ProCurve(config)#ip classless
```


ip crypto

Use the **ip crypto** command to enable Secure Router OS VPN functionality and allow crypto maps to be added to interfaces. Use the **no** form of this command to disable the VPN functionality.

Note *Disabling the Secure Router OS security features (using the **no ip crypto** command) does not affect VPN configuration settings (with the exception of the removal of all crypto maps from the interfaces). All other configuration parameters will remain intact, and VPN functionality will be disabled.*

*For VPN configuration example scripts, refer to the technical support note **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

No subcommands.

Default Values

By default, all Secure Router OS VPN functionality is disabled.

Functional Notes

VPN-related settings will not go into effect until you enable VPN functionality using the **ip crypto** command. The Secure Router OS allows you to perform all VPN-related configuration prior to enabling **ip crypto**, with the exception of assigning a **crypto map** to an interface. The **no ip crypto** command removes all crypto maps from the interfaces. Enabling **ip crypto** enables the IKE server on UDP port 500. The **no** form of this command disables the IKE server on UDP port 500.

Usage Examples

The following example enables VPN functionality:

```
ProCurve(config)#ip crypto
```

ip default-gateway <ip address>

Use the **ip default-gateway** command to specify a default gateway if (and only if) IP routing is NOT enabled on the unit. Use the **ip route** command to add a default route to the route table when using IP routing functionality. See *ip route* <ip address> <subnet mask> <interface or ip address> <administrative distance> [on page 338](#) for more information.

Syntax Description

<ip address>	Specifies the default gateway IP address in the form of dotted decimal notation (example: 192.22.71.50).

Default Values

By default, there is no configured default-gateway.

Functional Notes

Only use the **ip default-gateway** when IP routing is disabled on the router. For all other cases, use the **ip route 0.0.0.0 0.0.0.0 <ip address>** command.

Usage Examples

The following example disables IP routing and configures a default gateway for 192.22.71.50:

```
ProCurve(config)#no ip routing
ProCurve(config)#ip default gateway 192.22.71.50
```

ip dhcp-server database local

Use the **ip dhcp-server database local** command to configure a DHCP database agent with local bindings. Use the **no** form of this command to disable this option.

Syntax Description

No subcommands.

Default Values

No default values.

Usage Examples

The following example configures the DHCP database agent with local bindings:

```
ProCurve(config)#ip dhcp-server database local
```

ip dhcp-server excluded-address *<start ip> <end ip>*

Use the **ip dhcp-server excluded-address** command to specify IP addresses that cannot be assigned to DHCP clients. Use the **no** form of this command to remove a configured IP address restriction.

Syntax Description

<i><start ip></i>	Specifies the lowest IP address (using dotted decimal notation) in the range OR a single IP address to be excluded.
<i><end ip></i>	Optional. Specifies the highest IP address (using dotted decimal notation) in the range. This field is not required when specifying a single IP address.

Default Values

By default, there are no excluded IP addresses.

Functional Notes

The Secure Router OS DHCP server (by default) allows all IP addresses for the DHCP pool to be assigned to requesting clients. This command is used to ensure that the specified address is never assigned by the DHCP server. When static addressed hosts are present in the network, it is helpful to exclude the IP addresses of the host from the DHCP IP address pool. This will avoid IP address overlap.

Usage Examples

The following example excludes an IP address of 172.22.5.100 and the range 172.22.5.200 through 172.22.5.250:

```
ProCurve(config)#ip dhcp-server excluded-address 172.22.5.100
ProCurve(config)#ip dhcp-server excluded-address 172.22.5.200 172.22.5.250
```

ip dhcp-server ping packets <#packets>

Use the **ip dhcp-server ping packets** command to specify the number of ping packets the DHCP server will transmit before assigning an IP address to a requesting DHCP client. Transmitting ping packets verifies that no other hosts on the network are currently configured with the specified IP address. Use the **no** form of this command to prevent the DHCP server from using ping packets as part of the IP address assignment process.

Syntax Description

<#packets>	Specifies the number of DHCP ping packets sent on the network before assigning the IP address to a requesting DHCP client

Default Values

By default, the number of DHCP server ping packets is set to 2 packets.

Functional Notes

Before assigning an IP address to a requesting client, the Secure Router OS DHCP server transmits a ping packet on the network to verify there are no other network hosts already configured with the specified address. If the DHCP server receives no reply, the IP address is assigned to the requesting client and added to the DHCP database as an assigned address. Configuring the **ip dhcp-server ping packets** command with a value of **0** prevents the DHCP server from using ping packets as part of the IP address assignment process.

Usage Examples

The following example configures the DHCP server to transmit 4 ping packets before assigning an address:

```
ProCurve(config)#ip dhcp-server ping packets 4
```

ip dhcp-server ping timeout <milliseconds>

Use the **ip dhcp-server ping timeout** command to specify the interval (in milliseconds) the DHCP server will wait for a response to a transmitted DHCP ping packet. The DHCP server transmits ping packets before assigning an IP address to a requesting DHCP client. Transmitting ping packets verifies that no other hosts on the network are currently configured with the specified IP address. Use the **no** form of this command to return to the default timeout interval.

Syntax Description

<milliseconds>	Specifies the number of milliseconds (valid range: 1 to 1,000) the DHCP server will wait for a response to a transmitted DHCP ping packet.

Default Values

<milliseconds>	500 milliseconds

Functional Notes

Before assigning an IP address to a requesting client, the Secure Router OS DHCP server transmits a ping packet on the network to verify there are no other network hosts already configured with the specified address. If the DHCP server receives no reply, the IP address is assigned to the requesting client and added to the DHCP database as an assigned address.

Usage Examples

The following example configures the DHCP server to wait 900 milliseconds for a response to a transmitted DHCP ping packet before considering the ping a failure:

```
ProCurve(config)#ip dhcp-server ping timeout 900
```

ip dhcp-server pool <name>

Use the **ip dhcp-server pool** command to create a DHCP address pool and enter the DHCP pool. Use the **no** form of this command to remove a configured DHCP address pool. See the section *DHCP Pool Command Set* [on page 394](#) for more information.

Syntax Description

<name>	Identifies the configured DHCP server address pool by alphanumeric string (up to 32 characters in length) (example SALES).

Default Values

By default, there are no configured DHCP address pools.

Functional Notes

Use the **ip dhcp-server pool** to create multiple DHCP server address pools for various segments of the network. Multiple address pools can be created to service different segments of the network with tailored configurations.

Usage Examples

The following example creates a DHCP server address pool (labeled SALES) and enters the DHCP server pool:

```
ProCurve(config)#ip dhcp-server pool SALES
ProCurve(config-dhcp)#
```

ip domain-lookup

Use the **ip domain-lookup** command to enable the IP DNS (domain naming system), allowing DNS-based host translation (name-to-address). Use the **no** form of this command to disable DNS.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Functional Notes

Use the **ip domain-lookup** command to enable the DNS client in the router. This will allow the user to input web addresses instead of IP addresses for applications such as ping, Telnet, and traceroute.

Usage Examples

The following example enables DNS:

```
ProCurve(config)#ip domain-lookup
```


ip domain-name <name>

Use the **ip domain-name** command to define a default IP domain name to be used by the Secure Router OS to resolve host names. Use the **no** form of this command to disable this function.

Syntax Description

<name>	Default IP domain name used to resolve unqualified host names. Do not include the initial period that separates the unresolved name from the default domain name.

Default Values

By default, this command is disabled.

Functional Notes

Use the **ip domain-name** command to set a default name which will be used to complete any IP host name that is invalid (i.e., any name that is not recognized by the name-server). When this command is enabled, any IP host name that is not initially recognized will have the **ip domain-name** appended to it and the request will be resent.

Usage Examples

The following example defines **procurve** as the default domain name:

```
ProCurve(config)#ip domain-name procurve
```

ip domain-proxy

Use the **ip domain-proxy** command to enable DNS proxy for the router. This enables the router to act as a proxy for other units on the network.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

When this command is enabled, incoming DNS requests will be handled by the router. It will first search its host table for the query, and if it is not found there the request will be forwarded to the servers configured with the **ip name-server** command.

Usage Examples

The following example enables DNS proxy:

```
ProCurve(config)#ip domain-proxy
```

ip firewall

Use the **ip firewall** command to enable Secure Router OS security features including access control policies and lists, Network Address Translation (NAT), and the stateful inspection firewall. Use the **no** form of this command to disable the security functionality.

Note *Disabling the Secure Router OS security features (using the **no ip firewall** command) does not affect security configuration. All configuration parameters will remain intact, but no security data processing will be attempted.*

*Regarding the use of IKE negotiation for VPN with **ip firewall** enabled, there can be up to six channel groups with 2-8 interfaces per group. Dynamic protocols are not yet supported (only static). A physical interface can be a member of only one channel-group.*

Syntax Description

No subcommands.

Default Values

By default, all Secure Router OS security features are disabled.

Functional Notes

This command enables firewall processing for all interfaces with a configured policy class. Firewall processing consists of the following functions:

Attack Protection: Detects and discards traffic that matches profiles of known networking exploits or attacks.

Session Initiation Control: Allows only sessions that match traffic patterns permitted by access-control policies to be initiated through the router.

Ongoing Session Monitoring and Processing: Each session that has been allowed through the router is monitored for any irregularities that match patterns of known attacks or exploits. This traffic will be dropped. Also, if NAT is configured, the firewall modifies all traffic associated with the session according to the translation rules defined in NAT access-policies. Finally, if sessions are inactive for a user-specified amount of time, the session will be closed by the firewall.

Application Specific Processing: Certain applications need special handling to work correctly in the presence of a firewall. Secure Router OS uses ALGs (application-level gateways) for these applications.

The Secure Router OS includes several security features to provide controlled access to your network. The following features are available when security is enabled (using the **ip firewall** command):

1. Stateful Inspection Firewall

The Secure Router OS (and your unit) act as an application-level gateway and employ a stateful inspection firewall that protects an organization's network from common cyber attacks including TCP syn-flooding, IP spoofing, ICMP redirect, land attacks, ping-of-death, and IP reassembly problems. In addition, further security is added with use of Network Address Translation (NAT) and Port Address Translation (PAT) capability.

2. Access Policies (ACPs)

Secure Router OS access control policies are used to allow, discard, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded.

3. Access Lists (ACLs)

Access control lists are used as packet selectors by ACPs; by themselves they do nothing. ACLs are composed of an ordered list of entries. Each entry contains two parts: an action (permit or deny) and a packet pattern. A permit ACL is used to permit packets (meeting the specified pattern) to enter the router system. A deny ACL advances the Secure Router OS to the next access policy entry. The Secure Router OS provides two types of ACLs: standard and extended. Standard ACLs allow source IP address packet patterns only. Extended ACLs may specify patterns using most fields in the IP header and the TCP or UDP header.

Usage Examples

The following example enables the Secure Router OS security features:

```
ProCurve(config)#ip firewall
```

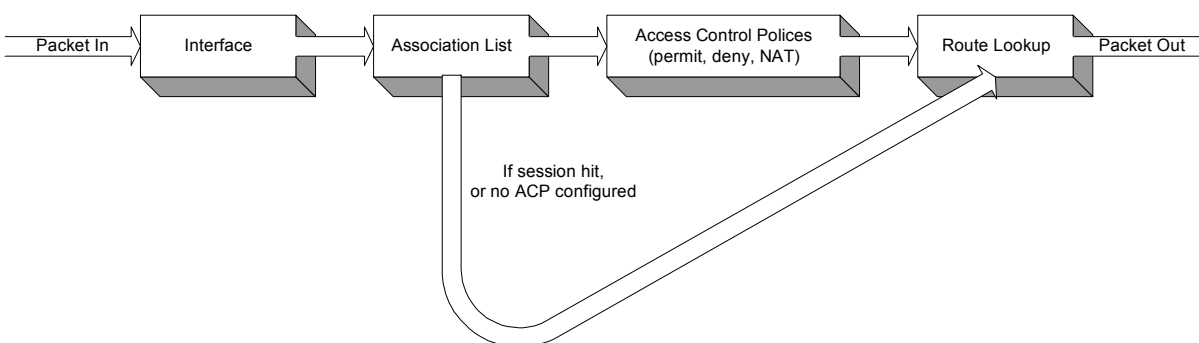
Technology Review

Concepts:

Access control using the Secure Router OS firewall has two fundamental parts: Access Control Lists (ACLs) and Access Policy Classes (ACPs). ACLs are used as packet selectors by other Secure Router OS systems; by themselves they do nothing. ACPs consist of a selector (ACL) and an action (allow, discard, NAT). ACPs integrate both allow and discard policies with NAT. ACPs have no effect until they are assigned to a network interface.

Both ACLs and ACPs are order dependent. When a packet is evaluated, the matching engine begins with the first entry in the list and progresses through the entries until it finds a match. The first entry that matches is executed.

Packet Flow:



Case 1: Packets from interfaces with a configured policy class to any other interface

ACPs are applied when packets are received on an interface. If an interface has not been assigned a policy class, by default it will allow all received traffic to pass through. If an interface has been assigned a policy class but the firewall has not been enabled with the **ip firewall** command, traffic will flow normally from this interface with no firewall processing.

Case 2: Packets that travel in and out a single interface with a configured policy class

These packets are processed through the ACPs as if they are destined for another interface (identical to Case 1).

Case 3: Packets from interfaces without a configured policy class to interfaces with one

These packets are routed normally and are not processed by the firewall. The **ip firewall** command has no effect on this traffic.

Case 4: Packets from interfaces without a configured policy class to other interfaces without a configured policy class

This traffic is routed normally. The **ip firewall** command has no effect on this traffic.

Attack Protection:

When the **ip firewall** command is enabled, firewall attack protection is enabled. The Secure Router OS blocks traffic (matching patterns of known networking exploits) from traveling through the device. For some of these attacks, the user may manually disable checking/blocking while other attack checks are always on anytime the firewall is enabled.

The table (on the following pages) outlines the types of traffic discarded by the Firewall Attack Protection Engine. Many attacks use similar invalid traffic patterns; therefore attacks other than the examples listed below may also be blocked by the firewall. To determine if a specific attack is blocked by the Secure Router OS firewall, please contact technical support.

Invalid Traffic Pattern	Manually Enabled?	OS Firewall Response	Common Attacks
Larger than allowed packets	No	Any packets that are longer than those defined by standards will be dropped.	Ping of Death
Fragmented IP packets that produce errors when attempting to reassemble	No	The firewall intercepts all fragments for an IP packet and attempts to reassemble them before forwarding to destination. If any problems or errors are found during reassembly, the fragments are dropped.	SynDrop, TearDrop, OpenTear, Nestea, Targa, Newtear, Bonk, Boink

Invalid Traffic Pattern	Manually Enabled?	OS Firewall Response	Common Attacks
Smurf Attack	No	The firewall will drop any ping responses that are not part of an active session.	Smurf Attack
IP Spoofing	No	The firewall will drop any packets with a source IP address that appears to be spoofed. The IP route table is used to determine if a path to the source address is known (out of the interface from which the packet was received). For example, if a packet with a source IP address of 10.10.10.1 is received on interface fr 1.16 and no route to 10.10.10.1 (through interface fr 1.16) exists in the route table, the packet is dropped.	IP Spoofing
ICMP Control Message Floods and Attacks	No	The following types of ICMP packets are allowed through the firewall: echo, echo-reply, TTL expired, dest. Unreachable, and quench. These ICMP messages are only allowed if they appear to be in response to a valid session. All others are discarded.	Twinge
Attacks that send TCP URG packets	Yes	Any TCP packets that have the URG flag set are discarded by the firewall.	Winnuke, TCP XMAS Scan
Falsified IP Header Attacks	No	The firewall verifies that the packet's actual length matches the length indicated in the IP header. If it does not, the packet is dropped.	Jolt/Jolt2
Echo	No	All UDP echo packets are discarded by the firewall.	Char Gen
Land Attack	No	Any packets with the same source and destination IP addresses are discarded.	Land Attack
Broadcast Source IP	No	Packets with a broadcast source IP address are discarded.	
Invalid TCP Initiation Requests	No	TCP SYN packets that have ack, urg rst, or fin flags set are discarded.	
Invalid TCP Segment Number	No	The sequence numbers for every active TCP session are maintained in the firewall session database. If the firewall received a segment with an unexpected (or invalid) sequence number, the packet is dropped.	
IP Source Route Option	No	All IP packets containing the IP source route option are dropped.	

Application Specific Processing:

The following applications and protocols require special processing to operate concurrently with NAT/firewall functionality. The Secure Router OS firewall includes ALGs for handling these applications and protocols:

AOL Instant Messenger (AIM®)

VPN ALGS: ESP and IKE

FTP

H.323: H.245 Q.931 ASN1 PER decoding and Encoding

ICQ®

IRC

Microsoft® Games

Net2Phone

PPTP

Quake®

Real-Time Streaming Protocol

SMTP

HTTP

CUseeme

SIP

L2TP

PcAnywhere™

SQL

Microsoft Gaming Zone

To determine if a specific application requires special processing, contact technical support.

ip firewall alg [ftp | h323 | pptp | sip]

Use the **ip firewall alg** command to enable the application level gateway (ALG) for a particular application. Use the **no** form of this command to disable ALG for the application.

Syntax Description

ftp	Enables the FTP ALG.
h323	Enables the H323 ALG. H.323 is a generic recommendation from the ITU that sets standards for multimedia communications over networks without guaranteed Quality of Service (QoS)
pptp	Enables the PPTP ALG.
sip	Enables the SIP ALG.

Default Values

By default, the ALG for FTP, H323, PPTP, and SIP are enabled.

Functional Notes

Enabling the Application Layer Gateway (ALG) for a specific protocol gives the firewall additional information about that complex protocol and causes the firewall to perform additional processing for packets of that protocol. When the ALG is disabled, the firewall treats the complex protocol as any other simple protocol. The firewall needs no special knowledge to work well with simple protocols.

Warning	<i>Disabling the IP firewall ALG may cause the firewall to block some of the traffic for the specified protocol.</i>

Usage Examples

The following example disables ALG for FTP:

```
ProCurve(config)#no ip firewall alg ftp
```

ip firewall check reflexive-traffic

Use the **ip firewall check reflexive-traffic** command to enable the Secure Router OS stateful inspection firewall to process traffic from a primary subnet to a secondary subnet on the same interface through the firewall. Use the **no** form of this command to disable this feature.

Note *The Secure Router OS security features must be enabled (using the **ip firewall** command) for the stateful inspection firewall to be activated.*

Syntax Description

No subcommands.

Default Values

All Secure Router OS security features are disabled by default until the **ip firewall** command is issued at the Global Configuration prompt. In addition, the reflexive traffic check is disabled until the **ip firewall check reflexive-traffic** command is issued.

Functional Notes

This command allows the firewall to process traffic from a primary subnet to a secondary subnet on the same interface through the firewall. If enabled, this traffic will be processed through the access-policy on that interface and any actions specified will be executed on the traffic.

Usage Examples

The following example enables the Secure Router OS reflexive-traffic check:

```
ProCurve(config)#ip firewall check reflexive-traffic
```

ip firewall attack-log threshold <value>

Use the **ip firewall attack-log threshold** command to specify the number of attack mounting attempts the Secure Router OS will identify before generating a log message. Use the **no** form of this command to return to the default threshold.

Note	<i>The Secure Router OS security features must be enabled (using the ip firewall command) for the stateful inspection firewall to be activated.</i>

Syntax Description

<value>	Specifies the number of attack mounting attempts the Secure Router OS will identify before generating a log message (valid range: 0 to 4294967295).

Default Values

<value>	100

Usage Examples

The following example specifies a threshold of 25 attacks before generating a log message:

```
ProCurve(config)#ip firewall attack-log threshold 25
```

ip firewall check syn-flood

Use the **ip firewall check syn-flood** command to enable the Secure Router OS stateful inspection firewall to filter out phony TCP service requests and allow only legitimate requests to pass through. Use the **no** form of this command to disable this feature.

Note	<i>The Secure Router OS security features must be enabled (using the ip firewall command) for the stateful inspection firewall to be activated.</i>

Syntax Description

No subcommands.

Default Values

All Secure Router OS security features are disabled by default until the **ip firewall** command is issued at the Global Configuration prompt. In addition, the SYN-flood check is disabled until the **ip firewall check syn-flood** command is issued.

Functional Notes

SYN Flooding is a well-known denial of service attack on TCP-based services. TCP requires a three-way handshake before actual communications begin between two hosts. A server must allocate resources to process new connection requests that are received. A potential intruder is capable of transmitting large amounts of service requests (in a very short period of time), causing servers to allocate all resources to process the phony incoming requests. Using the **ip firewall check syn-flood** command configures the Secure Router OS stateful inspection firewall to filter out phony service requests and allow only legitimate requests to pass through.

Usage Examples

The following example enables the Secure Router OS syn-flood check:

```
ProCurve(config)#ip firewall check syn-flood
```

ip firewall check winnuke

Use the **ip firewall check winnuke** command to enable the Secure Router OS stateful inspection firewall to discard all Out of Band (OOB) data (to protect against WinNuke attacks). Use the **no** form of this command to disable this feature.

Note *The Secure Router OS security features must be enabled (using the **ip firewall** command) for the stateful inspection firewall to be activated.*

Syntax Description

No subcommands.

Default Values

All Secure Router OS security features are disabled by default until the **ip firewall** command is issued at the Global Configuration prompt. Issuing the **ip firewall** command enables the WinNuke check.

Functional Notes

WinNuke attack is a well-known denial of service attack on hosts running Microsoft Windows® operating systems. An intruder sends Out of Band (OOB) data over an established connection to a Windows user. Windows cannot properly handle the OOB data and the host reacts unpredictably. Normal shut-down of the hosts will generally return all functionality. Using the **ip firewall check winnuke** command configures the Secure Router OS stateful inspection firewall to filter all OOB data to prevent network problems.

Usage Examples

The following example enables the firewall to filter all OOB data:

```
ProCurve(config)#ip firewall check winnuke
```

ip firewall policy-log threshold <value>

Use the **ip firewall policy-log threshold** command to specify the number of connections required by an access control policy before the Secure Router OS will generate a log message. Use the **no** form of this command to return to the default threshold.

Note	<i>The Secure Router OS security features must be enabled (using the ip firewall command) for the stateful inspection firewall to be activated.</i>

Syntax Description

<value>	Specifies the number of access policy connections the Secure Router OS will identify before generating a log message (valid range: 0 to 4,294,967,295).

Default Values

<value>	100

Usage Examples

The following example specifies a threshold of 15 connections before generating a log message:

```
ProCurve(config)#ip firewall policy-log threshold 15
```

ip forward-protocol udp <port number>

Use the **ip forward-protocol udp** command to specify the protocols and ports the Secure Router OS allows when forwarding broadcast packets. Use the **no** form of this command to disable a specified protocol or port from being forwarded.

Note *The **ip helper** command must be used in conjunction with the **ip forward-protocol** command to configure the Secure Router OS to forward UDP broadcast packets.*

Syntax Description

<port number>	Specifies the UDP traffic type (using source port)

The following is the list of UDP port numbers that may be identified using the text name:

biff (Port 512)	pim-auto-rp (496)
bootps (Port 67)	rip (Port 520)
discard (Port 9)	snmp (Port 161)
dnsix (Port 195)	snmptrap (Port 162)
domain (Port 53)	sunrpc (Port 111)
echo (Port 7)	syslog (Port 514)
isakmp (Port 500)	tacacs (Port 49)
mobileip (Port 434)	talk (Port 517)
nameserver (Port 42)	tftp (Port 69)
netbios-dgm (Port 138)	time (Port 37)
netbios-ns (Port 137)	who (Port 513)
netbios-ss (Port 139)	xmcp (Port 177)
ntp (Port 123)	

Alternately, the <port number> may be specified using the following syntax:

<0-65535>. Specifies the port number used by UDP to pass information to upper layers. All ports below 1024 are considered well-known ports and are controlled by the Internet Assigned Numbers Authority (IANA). All ports above 1024 are dynamically assigned ports that include registered ports for vendor-specific applications.

Default Values

By default, the Secure Router OS forwards broadcast packets for all protocols and ports.

Functional Notes

Use this command to configure the Secure Router OS to forward UDP packets across the WAN link to allow remote devices to connect to a UDP service on the other side of the WAN link.

Usage Examples

The following example forwards all Domain Name Server broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#ip helper-address 192.33.5.99
```

ip ftp access-class <polycyname> in

Use the **ip ftp access-class in** command to assign an access policy to all self-bound File Transfer Protocol (FTP) sessions.

Syntax Description

<polycyname>	Specifies the configured access policy (ACP) to apply to inbound FTP traffic

Default Values

By default, all ftp access is allowed.

Usage Examples

The following example applies the configured ACP (labeled Inbound_FTP) to inbound FTP traffic:

```
ProCurve(config)#ip ftp access-class Inbound_FTP in
```


ip ftp agent

Use the **ip ftp agent** command to enable the file transfer protocol (FTP) agent.

Syntax Description

No subcommands.

Default Values

By default, the FTP agent is enabled.

Usage Examples

The following example enables the IP FTP agent:

```
ProCurve(config)#ip ftp agent
```

ip ftp source-interface <interface>

Use the **ip ftp source-interface** command to use the specified interface's IP address as the source IP address for FTP traffic transmitted by the unit. Use the **no** form of this command if you do not wish to override the normal source IP address.

Syntax Description

<interface>	Specifies the interface to be used as the source IP address for FTP traffic. Type ip ftp source- interface? for a complete list of valid interfaces.

Default Values

No default value is necessary for this command.

Functional Notes

This command allows you to override the *sender* field in the IP packet. If you have multiple interfaces in your unit, changing the *sender* tells the receiver where to send replies. This functionality can also be used to allow packets to get through firewalls that would normally block the flow.

Usage Examples

configures the unit to use the **loopback 1** interface as the source IP for FTP traffic:

```
ProCurve(config)#ip ftp source-interface loopback 1
```

ip host <name> <address1>

Use the **ip host** command to define an IP host name. This allows you to statically map host names and addresses in the host cache. Use the **no** form of this command to remove defined maps.

Syntax Description

<name>	Name of the host.
<address1>	IP address associated with this IP host.

Default Values

By default, the host table is empty.

Functional Notes

The name may be any combination of numbers and letters as long as it is not a valid IP address or does not exceed 256 characters.

Usage Examples

The following example defines two static mappings:

```
ProCurve(config)#ip host mac 10.2.0.2
ProCurve(config)#ip host dal 172.38.7.12
```

**ip http [access-class <listname> in | authentication <listname> |
secure-access-class <listname> in | secure-server | server |
session-limit | session-timeout]**

Use the **ip http** command to enable web access to the unit.

Syntax Description

access-class	Enables HTTP for all incoming connections associated with a specific access list.
<listname>	Specifies the access list name.
in	Applies to all incoming connections.
authentication	Assigns the specified AAA list to HTTP authentication.
secure-access-class	Applies to all self-bound HTTPS connections.
secure-server	Enables the SSL server.
server	Enables the HTTP server connection.
session-limit	Sets the maximum number of sessions allowed. Valid range is 0 to 100 with 100 as the default.
session-timeout	Sets the session timeout. Valid range is 10 to 86,400 seconds. The default is 600.

Default Values

By default, this command is disabled.

Usage Examples

The following example enables web access to the router:

ProCurve(config)#**ip http server**

ip igmp join <group-address>

Use the **ip igmp join** command to instruct the router stack to join a specific group. The stack may join multiple groups.

Syntax Description

<group-address>	IP address of a multicast group.

Default Values

No defaults necessary for this command.

Functional Notes

This command aids in debugging, allowing the router's IP stack to connect to and respond on a multicast group. The local stack operates as an IGMP host on the attached segment. In multicast stub applications, the global helper address takes care of forwarding IGMP joins/responses on the upstream interface. The router may respond to ICMP echo requests for the joined groups.

Usage Examples

The following example configures the unit to join with the specified multicast group:

```
ProCurve(config)#ip igmp join 172.0.1.50
```

ip load-sharing [per-destination | per-packet]

Use the **ip load-sharing** command to configure whether parallel routes in the route table are used to load-share forwarded packets. If this command is disabled, the route table uses a single "best" route for a given subnet. If this command is enabled, the route table can use multiple "best" routes and alternate between them.

Syntax Description

per-destination	Specifies that the route used for forwarding a packet be based on a hash of the source and destination IP address in the packet.
per-packet	Specifies that each forwarding route lookup rotates through all the parallel "best" routes. (Parallel routes are defined as routes to the same subnet with the same metrics that only differ by their next hop address.)

Default Values

By default, ip load-sharing is disabled.

Usage Examples

The following example turns on load-sharing per destination:

ProCurve(config)# ip load-sharing per-destination

The following example disables load-sharing :

ProCurve(config)# no ip load-sharing

ip mcast-stub helper-address *<ip address>*

Use the **ip mcast-stub helper-address** command to specify an IP address toward which IGMP host reports and leave messages are forwarded. This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub downstream** and **ip mcast-stub upstream** commands. Use the **no** form of this command to return to default.

Syntax Description

<code><ip address></code>	IGMP host reports and leave messages are forwarded toward this address.

Default Values

By default, no helper-address is configured.

Functional Notes

Helper-address is configured globally and applies to all multicast-stub downstream interfaces. The address specified may be the next upstream hop or any upstream address on the distribution tree for the multicast source, up to and including the multicast source. The router selects, from the list of multicast-stub upstream interfaces, the interface on the shortest path to the specified address. The router then proxies, on the selected upstream interface (using an IGMP host function), any host joins/leaves received on the downstream interface(s). The router retransmits these reports with addresses set as if the report originated from the selected upstream interface.

For example, if the router receives multiple joins for a group, it will not send any extra joins out the upstream interface. Also, if it receives a leave, it will not send a leave until it is certain that there are no more subscribers on any downstream interface.

Usage Examples

The following example specifies 172.45.6.99 as the helper-address:

```
ProCurve(config)#ip mcast-stub helper-address 172.45.6.99
```

ip multicast-routing

Use the **ip multicast routing** command to enable the multicast router process. The command does not affect other multicast-related configuration. Use the **no** form of this command to disable. Disabling this command prevents multicast forwarding but does not remove other multicast commands and processes.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Usage Examples

The following example enables multicast functionality:

```
ProCurve(config)#ip multicast-routing
```


ip name-server <server-address1-6>

Use the **ip name-server** command to designate one or more name servers to use for name-to-address resolution. Use the **no** form of this command to remove any addresses previously specified.

Syntax Description

<server-address1-6> Specifies up to six name-server addresses.

Default Values

By default, no name servers are specified.

Usage Examples

The following example specifies host 172.34.1.111 as the primary name server and host 172.34.1.2 as the secondary server:

```
ProCurve(config)#ip name-server 172.34.1.111 172.34.1.2
```

This command will be reflected in the configuration file as follows:

```
ip name-server 172.34.1.111 172.34.1.2
```

ip policy-class <polycyname> max-sessions <number>

Use the **ip policy-class** command to create an access control policy and enter the access control policy. Use the **no** form of this command to delete an access policy and all the entries contained in it.

Note *Configured access policies will only be active if the **ip firewall** command has been entered at the Global Configuration mode prompt to enable the Secure Router OS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.*

Caution *Before applying an access control policy to an interface, verify your Telnet connection will not be affected by the policy. If a policy is applied to the interface you are connecting through and it does not allow Telnet traffic, your connection will be lost.*

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor (maximum of 255 characters). All access policy descriptors are case-sensitive.
max-sessions <number>	Optional. Configure a maximum number of allowed policy sessions. This number must be within the appropriate range limits. The limits are either 1 to 4000 or 1 to 30,000 (depending on the type of Secure Router OS device you are using).

Default Values

By default, all Secure Router OS security features are disabled and there are no configured access lists.

Functional Notes

Secure Router OS access control policies are used to allow, discard, or manipulate (using **NAT**) data for each physical interface. Each ACP consists of a selector (**access list**) and an action (**allow**, **discard**, **NAT**). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded.

The following commands are contained in the **policy-class**:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> policy <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> policy <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy

listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet or the Web GUI.

discard list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list *<access list names>* **address** *<IP address>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list *<access list names>* **interface** *<interface>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Usage Examples

See the **Technology Review** (which follows) for command syntax examples.

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** *<A.B.C.D>* to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.

3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a “range”. Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a “don’t care”. For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an access policy that uses a configured access list. Secure Router OS access policies are used to allow, discard, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

```
allow list <access list names>
discard list <access list names>
allow list <access list names> policy <access policy name>
discard list <access list names> policy <access policy name>
allow list <access list names> self
discard list <access list names> self
nat source list <access list names> address <IP address> overload
nat source list <access list names> interface <interface> overload
nat destination list <access list names> address <IP address>
```

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**. The following example assigns access policy **MatchAll** to the Ethernet 0/1 interface:

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#access-policy MatchAll
```

ip policy-timeout <protocol> <range> <port> <seconds>

Use multiple **ip policy-timeout** commands to customize timeout intervals for protocols (TCP UDP ICMP) or specific services (by listing the particular port number). Use the **no** form of this command to return to the default timeout values.

Syntax Description

<protocol>	Specifies the data protocol such as ICMP, TCP, or UDP.
<range>	Optional. Customizes timeout intervals for a range of TCP or UDP ports.
<port>	Service port to apply the timeout value to; valid only for specifying TCP and UDP services (not allowed for ICMP).

The following is the list of UDP port numbers that may be identified using the text name (in **bold**):

all-ports	ntp (Port 123)
biff (Port 512)	pim-auto-rp (496)
bootpc (Port 68)	rip (Port 520)
bootps (Port 67)	snmp (Port 161)
discard (Port 9)	snmptrap (Port 162)
dnsix (Port 195)	sunrpc (Port 111)
domain (Port 53)	syslog (Port 514)
echo (Port 7)	tacacs (Port 49)
isakmp (Port 500)	talk (Port 517)
mobile-ip (Port 434)	tftp (Port 69)
nameserver (Port 42)	time (Port 37)
netbios-dgm (Port 138)	who (Port 513)
netbios-ns (Port 137)	xdmcp (Port 177)
netbios-ss (Port 139)	

The following is the list of TCP port numbers that may be identified using the text name (in **bold**):

all_ports	login (Port 513)
bgp (Port 179)	lpd (Port 515)
chargen (Port 19)	nntp (Port 119)
cmd (Port 514)	pim-auto-rp (Port 496)
daytime (Port 13)	pop2 (Port 109)
discard (Port 9)	pop3 (Port 110)
domain (Port 53)	smtp (Port 25)
echo (Port 7)	sunrpc (Port 111)
exec (Port 512)	syslog (Port 514)

Syntax Description (Continued)

<port> <i>*Optional</i>	finger (Port 79)	tacacs (Port 49)
	ftp (Port 21)	talk (Port 517)
	ftp-data (Port 20)	telnet (Port 23)
	gopher (Port 70)	time (Port 37)
	hostname (Port 101)	uucp (Port 540)
	ident (Port 113)	whois (Port 43)
	irc (Port 194)	www (Port 80)
	klogin (Port 543)	
	kshell (Port 544)	
<seconds>	Wait interval (in seconds) before an active session is closed (valid range: 0 to 4294967295 seconds).	

Default Values

<seconds>	The following default policy timeout intervals apply: tcp (600 seconds; 10 minutes) udp (60 seconds; 1 minute) icmp (60 seconds; 1 minute)

Usage Examples

The following example creates customized policy timeouts for the following:

internet traffic (TCP Port 80) timeout 24 hours (86400 seconds)
telnet (TCP Port 23) timeout 20 minutes (1200 seconds)
FTP (21) timeout 5 minutes (300 seconds)
All other TCP services timeout 8 minutes (480 seconds)

```
ProCurve(config)#ip policy-timeout tcp www 86400
ProCurve(config)#ip policy-timeout tcp telnet 1200
ProCurve(config)#ip policy-timeout tcp ftp 300
ProCurve(config)#ip policy-timeout tcp all_ports 480
```

The following example creates customized policy timeouts for UDP netbios ports 137-139 of 200 seconds and UDP ports 6000-7000 of 300 seconds:

```
ProCurve(config)#ip policy-timeout udp range netbios-ns netbios-ss 200
ProCurve(config)#ip policy-timeout udp 6000 7000 300
```

ip prefix-list <listname> description <"text">

Use the **ip prefix-list description** command to create and name prefix lists.

Syntax Description

<listname>	Specifies a particular prefix list.
description <text>	Assigns text (set apart by quotation marks) used as a description for the prefix list. Maximum length is 80 characters.

Default Values

No default values are necessary for this command.

Functional Notes

This command adds a string of up to 80 characters as a description for a prefix list. It also creates the prefix list if a prefix list of that name does not already exist.

Usage Examples

The following example adds a description to the prefix-list **test**:

```
ProCurve(config)#ip prefix-list test description "An example prefix list"
```

ip prefix-list <listname> seq <sequence#> [permit | deny] <network/len> [le <le-value> | ge <ge-value>]

Use the **ip prefix-list seq** command to specify a prefix to be matched or a range of mask lengths.

Syntax Description

<listname>	Specifies a particular prefix list.
<sequence#>	Specifies the entry's unique sequence number which determines the processing order. Lower-numbered entries are processed first. Range: 1 to 4,294,967,294.
permit	Permits access to matching entries.
deny	Denies access to matching entries.
<network/len>	Specifies the network number and network mask length.
le <le-value>	Specifies the upper end of the range. Range: 0 to 32.
ge <ge-value>	Specifies the lower end of the range. Range: 0 to 32.

Default Values

If no ge or le parameters are specified, an exact match is assumed. If only ge is specified, the range is assumed to be from ge-value to 32. If only le is specified, the range is assumed to be from len to le-value.

Functional Notes

This command specifies a prefix to be matched. Optionally, it may specify a range of mask lengths. The following rule must be followed: $\text{len} < \text{ge-value} \leq \text{le-value}$. A prefix list with no entries allows all routes. A route that does not match any entries in a prefix list is dropped. As soon as a route is permitted or denied, there is no further processing of the rule in the prefix list. A route that is denied at the beginning entry of a prefix list will not be allowed, even if it matches a permitting entry further down the list.

Usage Examples

The following example creates a prefix list entry in the prefix list **test** matching only the 10.0.0.0/8 network:

```
ProCurve(config)#ip prefix-list test seq 5 deny 10.0.0.0/8
```

The following example creates a prefix list entry in the prefix list **test** matching any network of length 24 or less:

```
ProCurve(config)#ip prefix-list test seq 10 permit 0.0.0.0/0 le 24
```


ip radius source-interface *<interface>*

Use the **ip radius source-interface** command to specify the NAS (network-attached storage) IP address attribute passed with the RADIUS authentication request packet.

Syntax Description

<i><interface></i>	Specifies the source interface (in the format type slot/port). Type ip radius source-interface ? for a complete list of interfaces.

Default Values

By default, no source interface is defined.

Functional Notes

If this value is not defined, the address of the source network interface is used.

Usage Examples

The following example configures the Ethernet 0/1 port to be the source interface:

ProCurve(config)#**ip radius source-interface ethernet 0/1**

ip route *<ip address>* *<subnet mask>* *<interface or ip address>*
<administrative distance>

Use the **ip route** command to add a static route to the route table. This command can be used to add a default route by entering **ip route 0.0.0.0 0.0.0.0** and specifying the interface or IP address. Use the **no** form of this command to remove a configured static route.

Syntax Description

<i><ip address></i>	Specifies the network address (in dotted decimal notation) to add to the route table.
<i><subnet mask></i>	Specifies the subnet mask (in dotted decimal notation) associated with the listed network IP address.
<i><interface or ip address></i>	Specifies the gateway peer IP address (in dotted decimal notation) or a configured interface in the unit. Use the ? command to display a complete list of interfaces.
<i><administrative distance></i>	Specifies an administrative distance associated with this router (1 to 255). The administrative distance provides a way for a router to determine the best route when multiple routes to the same destination exist. The smaller the administrative distance, the more reliable the route.

Default Values

By default, there are no configured routes in route table.

Usage Examples

The following example adds a static route to the **10.220.0.0/16** network through the next-hop router **192.22.45.254** and a default route to **175.44.2.10**:

```
ProCurve(config)#ip route 10.220.0.0 255.255.0.0 192.22.45.254
ProCurve(config)#ip route 0.0.0.0 0.0.0.0 175.44.2.10
```

ip routing

Use the **ip routing** command to enable the Secure Router OS IP routing functionality. Use the **no** form of this command to disable IP routing.

Syntax Description

No subcommands.

Default Values

By default, IP routing is enabled.

Usage Examples

The following example enables the Secure Router OS IP routing functionality:

```
ProCurve(config)#ip routing
```

ip scp server

Use the **ip scp server** command to enable the secure copy server functionality within the Secure Router OS. Enabling the secure copy server allows the Secure Router OS to support the transfer of files using a secure connection. A secure connection helps provide protection against outside forces gaining access to configuration files. An external secure copy server (such as PuTTY) is required to facilitate the transfers from the terminal.

Syntax Description

No subcommands.

Default Value

By default, the secure copy server function is disabled.

Usage Examples

The following example enables the secure copy server function in the Secure Router OS:

```
ProCurve>enable
```

```
ProCurve#config terminal
```

```
ProCurve(config)#ip scp server
```

ip snmp agent

Use the **ip snmp agent** command to enable the Simple Network Management Protocol (SNMP) agent.

Syntax Description

No subcommands.

Default Values

By default, the SNMP agent is disabled.

Functional Notes

Allows a MIB browser to access standard MIBs within the product. This also allows the product to send traps to a trap management station.

Usage Examples

The following example enables the IP SNMP agent:

```
ProCurve(config)#ip snmp agent
```

ip sntp source-interface *<interface>*

The **ip sntp source-interface** command to use the specified interface's IP address as the source IP address for SNTP traffic transmitted by the unit. Use the **no** form of this command if you do not wish to override the normal source IP address.

Syntax Description

<i><interface></i>	Specifies the interface to be used as the source IP address for SNTP traffic. Type ip sntp source-interface? for a complete list of valid interfaces.

Default Values

No default value is necessary for this command.

Functional Notes

This command allows you to override the *sender* field in the IP packet. If you have multiple interfaces in your unit, changing the *sender* tells the receiver where to send replies. This functionality can also be used to allow packets to get through firewalls that would normally block the flow.

Usage Examples

The following example configures the unit to use the **loopback 1** interface as the source IP for SNTP traffic:

```
ProCurve(config)#ip sntp source-interface loopback
```

ip subnet-zero

The **ip subnet-zero** command is the default operation and cannot be disabled. This command signifies the router's ability to route to subnet-zero subnets.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Usage Examples

The following example **subnet-zero** is enabled:

```
ProCurve(config)#ip subnet-zero
```

ip tftp source-interface <interface>

Use the **ip tftp source-interface** command to use the specified interface's IP address as the source IP address for TFTP traffic transmitted by the unit. Use the **no** form of this command if you do not wish to override the normal source IP address.

Syntax Description

<interface>	Specifies the interface to be used as the source IP address for TFTP traffic.

Default Values

No default value is necessary for this command.

Default Values

No default value is necessary for this command.

Functional Notes

This command allows you to override the *sender* field in the IP packet. If you have multiple interfaces in your unit, changing the *sender* tells the receiver where to send replies. This functionality can also be used to allow packets to get through firewalls that would normally block the flow.

Usage Examples

The following example configures the unit to use the **loopback 1** interface as the source IP for TFTP traffic:

```
ProCurve(config)#ip tftp source-interface loopback 1
```


line [console | telnet] <line-number> <ending number>

Use the **line** command to enter the line configuration for the specified console or telnet session. See the sections *Line (Console) Interface Config Command Set* [on page 943](#) and *Line (Telnet) Interface Config Command Set* [on page 954](#) for information on the subcommands.

Syntax Description

console	Specifies the DB-9 (female) CONSOLE port located on the rear panel of the unit. See the sections <i>Line (Console) Interface Config Command Set</i> on page 943 for information on the subcommands found in this.
telnet	Specifies a Telnet session(s) to configure for remote access. See the section <i>Line (Telnet) Interface Config Command Set</i> on page 954 for information on the subcommands found in this.
<line-number>	Specifies the starting Telnet or console session to configure for remote access (valid range for console: 0; valid range for Telnet: 0 to 4). If configuring a single Telnet session, enter the Telnet session number and leave the <ending number> field blank.
<ending number>	Optional. Specifies the last Telnet session to configure for remote access (valid range: 0 to 4). To configure all available Telnet sessions, enter line telnet 0 4 .

Default Values

By default, the Secure Router OS line console parameters are configured as follows:

Data Rate: 9600
Data bits: 8
Stop bits: 1
Parity Bits: 0
No flow control

By default, there are no configured Telnet sessions.

Usage Examples

The following example begins the configuration for the **CONSOLE** port located on the rear of the unit:

```
ProCurve(config)#line console 0
ProCurve(config-con0)#
```

The following example begins the configuration for all available Telnet sessions:

```
ProCurve(config)#line telnet 0 4
ProCurve(config-telnet0-4)#
```

lldp [minimum-transmit-interval | reinitialization-delay | transmit-interval | ttl-multiplier] <numeric value>

Use the **lldp** command to configure global settings that control the way LLDP functions.

Syntax Description

minimum-transmit-interval	Defines the minimum amount of time between transmission of LLDP frames (in seconds).
reinitialization-delay	Minimum amount of time to delay after LLDP is disabled on a port before allowing transmission of additional LLDP frames on that port (in seconds).
transmit-interval	Defines the delay between LLDP frame transmission attempts during normal operation (in seconds).
ttl-multiplier	Defines the multiplier to be applied to the transmit interval to compute the time-to-live for data sent in an LLDP frame.
<numeric value>	Specifies the interval, delay, or multiplier.

Default Values

By default, minimum-transmit-interval = 2 seconds (valid range: 1 through 8192); reinitialization-delay = 2 seconds (valid range 1 through 10); transmit-interval = 30 seconds (valid range 5 through 32,768); and ttl-multiplier = 4 (valid range 2 through 10).

Functional Notes

Once a device receives data from a neighboring device in an LLDP frame, it will retain that data for a limited amount of time. This amount of time is called time-to-live, and it is part of the data in the LLDP frame. The time-to-live transmitted in the LLDP frame is equal to the transmit-interval multiplied by the ttl-multiplier.

Usage Examples

The following example sets the LLDP minimum-transmit-interval to 10 seconds:

```
ProCurve(config)#lldp minimum-transmit-interval 10
```

The following example sets the LLDP reinitialization-delay to 5 seconds:

```
ProCurve(config)#lldp reinitialization-delay 5
```

The following example sets the LLDP transmit-interval to 15 seconds:

```
ProCurve(config)#lldp transmit-interval 15
```

The following example sets the LLDP ttl-multiplier to 2 and the time-to-live for all LLDP frames transmitted from this unit to 30 seconds;

```
ProCurve(config)#lldp transmit-interval 15
```

```
ProCurve(config)#lldp ttl-multiplier 2
```

logging console

Use the **logging console** command to enable the Secure Router OS to log events to all consoles. Use the **no** form of this command to disable console logging.

Syntax Description

No subcommands.

Default Values

By default, logging console is disabled.

Usage Examples

The following example enables the Secure Router OS to log events to all consoles:

```
ProCurve(config)#logging console
```

logging email address-list *<email address>* ; *<email address>*

Use the **logging email** command to specify one or more email addresses that will receive notification when an event matching the criteria configured using the **logging email priority-level** command is logged by the Secure Router OS. See *logging email priority-level [error | fatal | info | notice | warning]* [on page 350](#) for more information. Use the **no** form of this command to remove a listed address.

Syntax Description

<i><email address></i>	Specifies the complete email address to use when sending logged messages (This field allows up to 256 characters.)
	Enter as many email addresses as desired, placing a semi-colon (;) between addresses.

Default Values

By default, there are no configured logging email addresses.

Usage Examples

The following example specifies three email addresses to use when sending logged messages:

```
ProCurve(config)#logging email address-list  
admin@email.com;ntwk@email.com;support@email.com
```

logging email on

Use the **logging email on** command to enable the Secure Router OS email event notification feature. Use the **logging email address-list** command to specify email address(es) that will receive notification when an event matching the criteria configured using the **logging email priority-level** command is logged by the Secure Router OS. See *logging email priority-level [error | fatal | info | notice | warning]* [on page 350](#) and *logging email priority-level [error | fatal | info | notice | warning]* [on page 350](#) for more information. Use the **no** form of this command to disable the email notification feature.

Syntax Description

No subcommands.

Default Values

By default, email event notification is disabled.

Functional Notes

The domain name is appended to the sender name when sending event notifications. See the command *ip domain-name <name>* [on page 305](#) for related information.

Usage Examples

The following example enables the Secure Router OS email event notification feature:

```
ProCurve(config)#logging email on
```

logging email priority-level [error | fatal | info | notice | warning]

Use the **logging email priority-level** command to set the threshold for events sent to the addresses specified using the **logging email address-list** command. All events with the specified priority or higher will be sent to all addresses in the list. The logging email on command must be enabled. See *logging email priority-level [error | fatal | info | notice | warning]* [on page 350](#) and *logging email on* [on page 349](#) for related information. Use the **no** form of this command to return to the default priority.

Syntax Description

Sets the minimum priority threshold for sending messages to email addresses specified using the **logging email address-list** command.

The following priorities are available (ranking from lowest to highest):

Error

When selected, events with **error** and **fatal** priorities are logged.

Fatal

When selected, only events with a **fatal** priority are logged.

Info

When selected, all events are logged.

Notice

When selected, events with **notice**, **warning**, **error**, and **fatal** priorities are logged.

Warning

When selected, events with **warning**, **error**, and **fatal** priorities are logged.

Default Values

<priority>	warning

Usage Examples

The following example sends all messages with **warning** level or greater to the email addresses listed using the **logging email address-list** command:

```
ProCurve(config)#logging email priority-level warning
```

logging email receiver-ip <ip address>

Use the **logging email receiver-ip** command to specify the IP address of the email server to use when sending notification that an event matched the criteria configured using the **logging email priority-level** command. See *logging email priority-level [error | fatal | info | notice | warning]* [on page 350](#) for related information. Use the **no** form of this command to remove a configured address.

Syntax Description

<ip address>	Specifies the IP address (in dotted decimal notation) of the mail server to use when sending logged messages.

Default Values

By default, there are no configured email server addresses.

Usage Examples

The following example specifies an email server (with address 172.5.67.99) to use when sending logged messages:

```
ProCurve(config)#logging email receiver-ip 172.5.67.99
```

logging email sender

Use the **logging email sender** command to specify the sender in an outgoing email message. This name will appear in the **From** field of the receiver's inbox. Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

No default value is necessary for this command.

Usage Examples

The following example sets a sender for outgoing messages:

```
ProCurve(config)#logging email sender myUnit@myNetwork.com
```


logging email source-interface <interface>

Use the **logging email source-interface** command to use the specified interface's IP address as the source IP address for email messages transmitted by the unit. Use the **no** form of this command if you do not wish to override the normal source IP address.

Syntax Description

<interface>	Specifies the interface to be used as the source IP address for email messages.

Default Values

No default value is necessary for this command.

Functional Notes

This command allows you to override the *sender* field in the IP packet. If you have multiple interfaces in your unit, changing the *sender* tells the receiver where to send replies. This functionality can also be used to allow packets to get through firewalls that would normally block the flow.

Usage Examples

The following example configures the unit to use the **loopback 1** interface as the source IP for email messages:

```
ProCurve(config)#logging email source-interface loopback 1
```

logging facility <facility type>

Use the **logging facility** command to specify a syslog facility type for the syslog server. Error messages meeting specified criteria are sent to the syslog server. For this service to be active, you must enable log forwarding. See *logging forwarding* on [page 355](#) for related information. Facility types are described under **Functional Notes** below. Use the **no** form of this command to return it to its default setting.

Syntax Description

<facility type>	Specifies the syslog facility type (see Functional Notes below).

Default Values

The default value is local7.

Functional Notes

The following is a list of all the valid facility types:

auth	Authorization system
cron	Cron facility
daemon	System daemon
kern	Kernel
local0 - local7	Reserved for locally defined messages
lpr	Line printer system
mail	Mail system
news	USENET news
sys9 - sys14	System use
syslog	System log
user	User process
uucp	UNIX-to-UNIX copy system

Usage Examples

The following example configures the syslog facility to the cron facility type:

```
ProCurve(config)#logging facility cron
```

logging forwarding on

Use the **logging forwarding on** command to enable the Secure Router OS syslog event feature. Use the **logging forwarding priority-level** command to specify the event matching the criteria used by the Secure Router OS to determine whether a message should be forwarded to the syslog server. See *logging forwarding priority-level [error | fatal | info | notice | warning]* [on page 356](#) for related information. Use the **no** form of this command to disable the syslog event feature.

Syntax Description

No subcommands.

Default Values

By default, syslog event notification is disabled.

Usage Examples

The following example enables the Secure Router OS syslog event feature:

```
ProCurve(config)#logging forwarding on
```

logging forwarding priority-level [error | fatal | info | notice | warning]

Use the **logging forwarding priority-level** command to set the threshold for events sent to the configured syslog server specified using the **logging forwarding receiver-ip** command. All events with the specified priority or higher will be sent to all configured syslog servers. See *logging email priority-level [error | fatal | info | notice | warning]* [on page 350](#) for more information. Use the **no** form of this command to return to the default priority.

Syntax Description

Sets the minimum priority threshold for sending messages to the syslog server specified using the **logging forwarding receiver-ip** command.

The following priorities are available (ranking from lowest to highest):

Error

When selected, events with **error** and **fatal** priorities are logged.

Fatal

When selected, only events with a **fatal** priority are logged.

Info

When selected, all events are logged.

Notice

When selected, events with **notice**, **warning**, **error**, and **fatal** priorities are logged.

Warning

When selected, events with **warning**, **error**, and **fatal** priorities are logged.

Default Values

<priority>	warning

Usage Examples

The following example sends all messages with **warning** level or greater to the syslog server listed using the **logging forwarding receiver-ip** command.

```
ProCurve(config)#logging forwarding priority-level warning
```

logging forwarding receiver-ip <ip address>

Use this **logging forwarding receiver-ip** command to specify the IP address of the syslog server to use when logging events that match the criteria configured using the **logging forwarding priority-level** command. Enter multiple **logging forwarding receiver-ip** commands to develop a list of syslog servers to use. See *logging forwarding priority-level [error | fatal | info | notice | warning]* [on page 356](#) for related information. Use the **no** form of this command to remove a configured address.

Syntax Description

<ip address>	Specifies the IP address (in dotted decimal notation) of the syslog server to use when logging messages.

Default Values

By default, there are no configured syslog server addresses.

Usage Examples

The following example specifies a syslog server (with address 172.5.67.99) to use when logging messages:

```
ProCurve(config)#logging forwarding receiver-ip 172.5.67.99
```

logging forwarding source-interface

Use the **logging forwarding source-interface** command to configure the specified interface's IP address as the source IP address for the syslog server to use when logging events. Use the **no** form of this command if you do not wish to override the normal source IP address.

Syntax Description

<interface> Specifies the interface to be used as the source IP address for event log traffic.

Default Values

No default value is necessary for this command.

Functional Notes

This command allows you to override the *sender* field in the IP packet. If you have multiple interfaces in your unit, changing the *sender* tells the receiver where to send replies. This functionality can also be used to allow packets to get through firewalls that would normally block the flow.

Usage Examples

configures the unit to use the **loopback 1** interface as the source IP for event log traffic:

```
ProCurve(config)#logging forwarding source-interface loopback 1
```

mac address-table aging-time <aging time>

Use the **mac address-table aging-time** command to set the length of time dynamic MAC addresses remain in the switch or bridge forwarding table. Use the **no** form of this command to reset this length to its default.

Syntax Description

<aging time>	Specifies an aging time (in seconds) from 10 to 1,000,000. Set to 0 to disable the timeout.

Default Values

By default, the aging time is 300 seconds.

Usage Examples

The following example sets the aging time to 10 minutes:

```
ProCurve(config)#mac address-table aging-time 600
```

mac address-table static *<mac address>* **bridge** *<group#>* **interface** [atm | ethernet | frame-relay | hdlc | ppp] *<interface id>*

Use the **mac address-table static** command to insert a static MAC address entry into the MAC address table. Use the **no** form of this command to remove an entry from the table.

Syntax Description

<i><mac address></i>	Specifies a valid 48-bit MAC address.
<i><group#></i>	Specifies a valid bridge group number (1-255).
interface	Specifies the interface the packets are forwarded to.
<i><interface id></i>	Specifies a valid slot/port interface ID (e.g., eth 0/1).

Default Values

By default, there are no static entries configured.

Usage Examples

The following example adds a static MAC address to Ethernet 0/1 on bridge group 4:

```
ProCurve(config)#mac address-table static 00:12:79:00:00:01 bridge 4 interface ethernet 0/1
```


modem countrycode [<countrycode>]

Use the **modem countrycode** command to specify the modem configuration for the applicable country.

Syntax Description

<countrycode>	Specifies the modem configuration for the applicable country.

Default Values

By default, **modem countrycode** is set to **USA/Canada**.

Functional Notes

The following country codes are available for modem configuration:

Algeria	- Algeria Modem configuration
Argentina	- Argentina Modem configuration
Australia	- Australia Modem configuration
Austria	- Austria Modem configuration
Bahrain	- Bahrain Modem configuration
Belgium	- Belgium Modem configuration
Bolivia	- Bolivia Modem configuration
Brazil	- Brazil Modem configuration
Chile	- Chile Modem configuration
China	- China Modem configuration
Colombia	- Colombia Modem configuration
Costa_Rica	- Costa_Rica Modem configuration
Cyprus	- Cyprus Modem configuration
Czechoslovakia	- Czechoslovakia Modem configuration
Denmark	- Denmark Modem configuration
Ecuador	- Ecuador Modem configuration
Egypt	- Egypt Modem configuration
Finland	- Finland Modem configuration
France	- France Modem configuration
Germany	- Germany Modem configuration
Greece	- Greece Modem configuration
Guatemala	- Guatemala Modem configuration
Hong_Kong	- Hong_Kong Modem configuration
Hungary	- Hungary Modem configuration
India	- India Modem configuration
Indonesia	- Indonesia Modem configuration
Ireland	- Ireland Modem configuration
Israel	- Israel Modem configuration
Italy	- Italy Modem configuration
Japan	- Japan Modem configuration
Jordan	- Jordan Modem configuration

Korea	- Korea Modem configuration
Kuwait	- Kuwait Modem configuration
Lebanon	- Lebanon Modem configuration
Malaysia	- Malaysia Modem configuration
Mexico	- Mexico Modem configuration
Morocco	- Morocco Modem configuration
Netherlands	- Netherlands Modem configuration
New_Zealand	- New_Zealand Modem configuration
Norway	- Norway Modem configuration
Oman	- Oman Modem configuration
Panama	- Panama Modem configuration
Peru	- Peru Modem configuration
Philippines	- Philippines Modem configuration
Poland	- Poland Modem configuration
Portugal	- Portugal Modem configuration
Puerto_Rico	- Puerto_Rico Modem configuration
Qatar	- Qatar Modem configuration
Russia	- Russia Modem configuration
Saudi_Arabia	- Saudi_Arabia Modem configuration
Singapore	- Singapore Modem configuration
Slovakia	- Slovakia Modem configuration
Slovenia	- Slovenia Modem configuration
South_Africa	- South_Africa Modem configuration
Spain	- Spain Modem configuration
Sweden	- Sweden Modem configuration
Switzerland	- Switzerland Modem configuration
Syria	- Syria Modem configuration
Taiwan	- Taiwan Modem configuration
Thailand	- Thailand Modem configuration
Trinidad	- Trinidad Modem configuration
Tunisia	- Tunisia Modem configuration
Turkey	- Turkey Modem configuration
UAE	- UAE Modem configuration
UK	- UK Modem configuration
USA/Canada	- USA/Canada Modem configuration
Uruguay	- Uruguay Modem configuration
Venezuela	- Venezuela Modem configuration
Yemen	- Yemen Modem configuration

Usage Example

The following example specifies to use the **USA/Canada** modem configuration.

ProCurve(config)#**modem countrycode USA/Canada**

qos map <mapname><sequence number>

Use the **qos map** command to activate the QoS Map Command Set (which allows you to create and/or edit a QoS map). For details on specific commands, refer to the section *Quality of Service (QoS) Map Commands* [on page 984](#). Use the **no** form of this command to delete a map entry.

Syntax Description

<mapname>	Specifies the QoS map name.
<sequence number>	Specifies a number (valid range: 0 to 65,535) to differentiate this QoS map and to assign match order.

Default Values

No default value is necessary for this command.

Functional Notes

A QoS policy is defined using a QoS map. The QoS map is a named list with sequenced entries. An entry contains a single match reference and one or more actions (priority, set, or both). Multiple map entries for the same QoS map are differentiated by a sequence number. The sequence number is used to assign match order.

Once created, a QoS map must be applied to an interface (using the `qos-policy out <map-name>` command) in order to actively process traffic. Any traffic for the interface that is not sent to the priority queue is sent using the default queuing method for the interface (such as weighted fair queuing). `qos-policy out <mapname>` [on page 660](#) for more information.

Usage Examples

The following example demonstrates basic settings for a QoS map and assigns a map to the frame-relay interface:

```
ProCurve>enable
```

```
ProCurve#config terminal
```

```
ProCurve(config)#qos map VOICEMAP 10
```

```
ProCurve(config-qos-map)#match precedence 5
```

```
ProCurve(config-qos-map)#priority 512
```

```
ProCurve(config-qos-map)#exit
```

```
ProCurve(config)#interface fr 1
```

```
ProCurve(config-fr 1)#qos-policy out VOICEMAP
```

radius-server

Use the **radius-server** command to configure several global RADIUS parameters. Most of these global defaults can be overridden on a per-server basis.

Variations of this command include the following:

radius-server challenge-noecho

radius-server deadtime *<minutes>*

radius-server enable-username *<name>*

radius-server key *<key>*

radius-server retry *<attempts>*

radius-server timeout *<seconds>*

Syntax Description

challenge-noecho	Turns off echoing of user challenge-entry. When echo is turned on, users see the text of the challenge as they type responses. Enabling this option hides the text as it is being entered.
deadtime <i><minutes></i>	Specifies how long a RADIUS server is considered dead once a timeout occurs. The server will not be tried again until after the deadtime expires.
enable-username <i><name></i>	Specifies a username to be used for enable authentication.
key <i><key></i>	Specifies the shared key to use with a RADIUS server.
retry <i><attempts></i>	Specifies how many attempts to make on a RADIUS server before marking it dead.
timeout <i><seconds></i>	Specifies how long to wait for a RADIUS server to respond to a request.

Default Values

challenge-noecho	By default, echo is turned on.
deadtime	1 minute
key	No default
retry	3 attempts
timeout	5 seconds
enable-username	\$enab15\$

Functional Notes

RADIUS servers (as defined with the **radius-server** command) may have many optional parameters. However, they are uniquely identified by their addresses and ports. Port values default to 1812 and 1813 for authorization and accounting, respectively. If a server is added to a named group but is not defined by a **radius-server** command, the server is simply ignored when accessed. Empty server lists are not allowed. When the last server is removed from a list, the list is automatically deleted.

Usage Examples

The following example shows a typical configuration of these parameters:

```
ProCurve(config)#radius-server challenge-noecho
ProCurve(config)#radius-server deadtime 10
ProCurve(config)#radius-server timeout 2
ProCurve(config)#radius-server retry 4
ProCurve(config)#radius-server key my secret key
```

radius-server host

Use the **radius-server host** to specify the parameters for a remote RADIUS server. At a minimum, the address (IP or DNS name) of the server must be given. The other parameters are also allowed and (if not specified) will take default values or fall back on the global RADIUS server's default settings.

Syntax Description

acct-port <port#>	Sends accounting requests to this remote port.
auth-port <port#>	Sends authentication requests to this remote port.
retry <attempts>	Retries server after timeout this number of times (uses RADIUS global setting if not given).
timeout <seconds>	Waits for a response this number of seconds (uses RADIUS global setting if not given).
key <key>	Defines the shared key with the RADIUS server (uses RADIUS global setting if not given). Note that the key must appear last on the input line since it reads the rest of the line beyond the key keyword.

Default Values

acct-port	1813
auth-port	1812

Usage Examples

The following example shows a typical configuration of these parameters:

ProCurve(config)#**radius-server host 1.2.3.4**

ProCurve(config)#**radius-server host 3.3.1.2 acct-port 1646 key my key**

route-map <map-name> [**permit** | **deny**] <sequence number>

Use the **route-map** command to create a route map and enter the Route Map Configuration command set. A route map is a type of filter that matches various attributes and then performs actions on the way the route is redistributed. Use the **no** form of this command to return to default settings.

Syntax Description

<map-name>	Specifies a name for the route map.
permit	Redistributes routes matching the route map attributes.
deny	Specifies not to redistribute routes matching the route map attributes.
<sequence number>	Specifies a sequence number of this route entry. Range is 1 to 4,294,967,295.

Default Values

By default, no route maps are defined.

Functional Notes

Route maps can be assigned to a neighbor using the **route-map** command in the BGP Neighbor command set. See *route-map* <map-name> [in | out] [on page 796](#) for more information.

Usage Examples

The following example creates the route map, specifies that routes matching its criteria will be denied, and assigns a sequence number of 100:

```
ProCurve(config)#route-map MyMap deny 100
ProCurve(config-route-map)#
```

You can then define the attributes of the route map from the Route Map Configuration Command set. Enter a ? at the **(config-route-map)#** prompt to explore the available options.

router bgp <AS number>

Use the **router bgp** command to enter the BGP Configuration mode. Refer to the BGP Configuration Command section for more information.

Syntax Description

<AS number>	Specifies the AS number for this BGP interface.

Default Values

No default values necessary for this command.

Usage Examples

The following example uses the **router bgp** command to enter the BGP Configuration mode:

```
ProCurve(config)#router bgp 65000
ProCurve>(config-bgp)#
```

Technology Review

The following AOS BGP-related guidelines may help guide decisions made during basic BGP implementation.

Ignore route if next hop is unreachable.

Prefer route with largest weight (only used in the local router, set by applying route maps to set this value on desired inbound updates).

Prefer route with largest local preference.

Prefer route injected by this router via network command.

Prefer route with shortest AS_PATH.

Prefer route with lowest origin type. Routes originally injected by the network command or aggregation (IGP) have a lower origin type than those originally injected by redistribution into BGP.

Prefer routes with lowest MED value.

Before the route is installed into the route table (forwarding table), a check is made of other sources that may have information about the same subnet (static routes, IGP, etc.) The route with the lowest administrative distance is installed.

router ospf

Use the **router ospf** command to activate OSPF in the router and to enter the OSPF Configuration mode. See the section *Router (OSPF) Configuration Command Set* [on page 970](#) for more information. Use the **no** form of this command to disable OSPF routing.

Syntax Description

No subcommands.

Default Values

By default, OSPF is disabled.

Functional Notes

The Secure Router OS can be configured to use OSPF with the firewall enabled (using the **ip firewall** command). To do this, configure the OSPF networks as usual, specifying which networks the system will listen for and broadcast OSPF packets to. See *ip firewall* [on page 307](#) for more information.

To apply stateful inspection to packets coming into the system, create a policy-class that describes the type of action desired and then associate that policy-class to the particular interface (see *ip policy-class* [<policyname> max-sessions <number> on page 330](#)). The firewall is intelligent and will only allow OSPF packets that were received on an OSPF configured interface. No modification to the policy-class is required to allow OSPF packets into the system.

Usage Examples

The following example uses the **router ospf** command to enter the OSPF Configuration mode:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#
```

router rip

Use the **router rip** command to enter the RIP Configuration mode. See the section *Router (RIP) Configuration Command Set* [on page 961](#) for more information.

Syntax Description

No subcommands.

Default Values

No default values necessary for this command.

Usage Examples

The following example uses the **router rip** command to enter the RIP Configuration mode:

```
ProCurve(config)#router rip  
ProCurve(config-rip)#
```

Technology Review

The RIP protocol is based on the Bellman-Ford (distance-vector) algorithm. This algorithm provides that a network will converge to the correct set of shortest routes in a finite amount of time, provided that:

- Gateways continuously update their estimates of routes.
- Updates are not overly delayed and are made on a regular basis.
- The radius of the network is not excessive.
- No further topology changes take place.

RIP is described in RFC 1058 (Version 1) and updated in RFCs 1721, 1722, and 1723 for Version 2. Version 2 includes components that ease compatibility in networks operating with RIP V1.

All advertisements occur on regular intervals (every 30 seconds). Normally, a route that is not updated for 180 seconds is considered dead. If no other update occurs in the next 60 seconds for a new and better route, the route is flushed after 240 seconds. Consider a connected route (one on a local interface). If the interface fails, an update is immediately triggered for that route only (advertised with a metric of 16).

Now consider a route that was learned and does not receive an update for 180 seconds. The route is marked for deletion, and even if it was learned on an interface, a poisoned (metric =16) route should be sent by itself immediately and during the next two update cycles with the remaining normal split horizon update routes. Following actual deletion, the poison reverse update ceases. If an update for a learned route is not received for 180 seconds, the route is marked for deletion. At that point, a 120-second garbage collection (GC) timer is started. During the GC timer, expiration updates are sent with the metric for the timed out route set to 16.

If an attached interface goes down, the associated route is immediately (within the same random five-second interval) triggered. The next regular update excludes the failed interface. This is the so-called first hand knowledge rule. If a gateway has first hand knowledge of a route failure (connected interfaces) or reestablishment, the same action is taken. A triggered update occurs, advertising the route as failed (metric = 16) or up (normal metric) followed by the normal scheduled update.

The assumption here is that if a gateway missed the triggered update, it will eventually learn from another gateway in the standard convergence process. This conserves bandwidth.

RIP-Related Definitions:

Route - A description of the path and its cost to a network.

Gateway - A device that implements all or part of RIP - a router.

Hop - Metric that provides the integer distance (number of intervening gateways) to a destination network gateway.

Advertisement - A broadcast or multicast packet to port 520 that indicates the route for a given destination network.

Update - An advertisement sent on a regular 30-second interval including all routes exclusive of those learned on an interface.

snmp-server chassis-id *<id string>*

Use the **snmp-server chassis-id** command to specify an identifier for the Simple Network Management Protocol (SNMP) server. Use the **no** form of this command to return to the default value.

Syntax Description

<i><id string></i>	Identifies the product by alphanumeric string (up to 32 characters in length).

Default Values

<i><id string></i>	Chassis ID

Usage Examples

The following example configures a chassis ID of **A432692**:

```
ProCurve(config)#snmp-server chassis-id A432692
```

snmp-server community <community> view <viewname> [ro | rw] <listname>

Use the **snmp-server community** command to specify a community string to control access to Simple Network Management Protocol (SNMP) information. Use the **no** form of this command to remove a specified community.

Syntax Description

<community>	Specifies the community string (a password to grant SNMP access).
view <viewname>	Optional. Specifies a previously defined view. Views define objects available to the community. For information on creating a new view, see <i>snmp-server view <viewname> <oidtree> [excluded included]</i> on page 382 .
ro	Optional. Keyword to grant read-only access, allowing retrieval of MIB objects.
rw	Optional. Keyword to grant read-write access, allowing retrieval and modification of MIB objects.
<listname>	Optional. Specifies an access-control list name used to limit access. Refer to <i>ip access-list extended <listname></i> on page 285 and <i>ip access-list standard <listname> [permit deny] <ip address></i> on page 291 for more information on creating access-control lists.

Default Values

By default, there are no configured SNMP communities.

Usage Examples

The following example specifies a community named **MyCommunity**, specifies a previously defined view named **blockinterfaces**, and assigns read-write access:

```
ProCurve(config)#snmp-server community MyCommunity view blockinterfaces rw
```

snmp-server contact <*“string”*>

Use the **snmp-server contact** command to specify the Simple Network Management Protocol (SNMP) sysContact string. Use the **no** form of this command to remove a configured contact.

Syntax Description

<i>“<string>”</i>	Alphanumeric string encased in quotation marks (up to 32 characters in length) used to populate the sysContact string.

Default Values

<i><string></i>	Customer Service

Usage Examples

The following example specifies **Network Administrator x4000** for the sysContact string:

```
ProCurve(config)#snmp-server contact "Network Administrator x4000"
```

snmp-server enable traps *<trap type>* [**snmp**]

Use the **snmp-server enable traps** command to enable all Simple Network Management Protocol (SNMP) traps available on your system or specified using the *<trap type>* option. Use multiple **snmp-server enable traps** to enable multiple trap types. Use the **no** form of this command to disable traps (or the specified traps).

Syntax Description

<i><trap type></i>	Optional. Specifies the type of notification trap to enable. Leaving this option blank enables ALL system traps.
snmp	Optional. Enables a subset of traps specified in RFC 1157. The following traps are supported: coldStart warmStart linkUp linkDown authenticationFailure

Default Values

By default, there are no enabled traps.

Usage Examples

The following example enables the SNMP traps:

```
ProCurve(config)#snmp-server enable traps snmp
```

snmp-server host <address> traps <community> <trap type> [snmp]

Use the **snmp-server host traps** command to specify traps sent to an identified host. Use multiple **snmp-server host traps** commands to specify all desired hosts. Use the **no** form of this command to return to the default value.

Syntax Description

<address>	Specifies the IP address of the SNMP host that receives the traps.
<community>	Specifies the community string (used as a password) for authorized agents to obtain access to SNMP information.
<trap type>	Optional. Specifies the type of notification trap to enable. Leaving this option blank enables ALL system traps.
snmp	Optional. Enables a subset of traps specified in RFC 1157. The following traps are supported: coldStart warmStart linkUp linkDown authenticationFailure

Default Values

By default, there are no hosts or traps enabled.

Usage Examples

The following example sends all SNMP traps to the host at address **190.3.44.69** and community string **My Community**:

```
ProCurve(config)#snmp-server host 190.3.44.69 traps My Community snmp
```


snmp-server host <address> traps version <version> <community> <trap type> [snmp]

Use the **snmp-server host traps version** command to specify traps sent to an identified host. Use multiple **snmp-server host traps version** commands to specify all desired hosts. Use the **no** form of this command to return to the default value.

Syntax Description

<address>	Specifies the IP address of the SNMP host that receives the traps.
<version>	Specifies the SNMP version as one of the following: 1 - SNMPv1 2C - SNMPv2C
<community>	Specifies the community string (used as a password) for authorized agents to obtain access to SNMP information.
<trap type>	Optional. Specifies the type of notification trap to enable. Leaving this option blank enables ALL system traps.
snmp	Optional. Enables a subset of traps specified in RFC 1157. The following traps are supported: coldStart warmStart linkUp linkDown authenticationFailure

Default Values

By default, there are no hosts or traps enabled.

Usage Examples

The following example sends all SNMP traps to the host at address **190.3.44.69** and community string **My Community** using SNMPv2C:

```
ProCurve(config)#snmp-server host 190.3.44.69 traps version 2c My Community snmp
```

snmp-server location <*“string”*>

Use the **snmp-server location** command to specify the Simple Network Management Protocol (SNMP) system location string. Use the **no** form of this command to return to the default value.

Syntax Description

<i>“<string>”</i>	Alphanumeric string encased in quotation marks (up to 32 characters in length) used to populate the system location string.

Default Values

<i><string></i>	ProCurve

Usage Examples

The following example specifies a location of **5th Floor Network Room**:

```
ProCurve(config)#snmp-server location “5th Floor Network Room”
```

snmp-server management-url <URL>

Use the **snmp-server management-url** command to specify the URL for the device's management software. Use the no form of this command to remove the management URL.

Syntax Description

<URL>	Specifies the URL for the management software.

Default Values

No default is necessary for this command.

Usage Examples

The following example specifies the URL *http://www.mywatch.com* as the device's management software:

```
ProCurve(config)#snmp-server management-url http://www.mywatch.com
```

snmp-server management-url-label <label>

Use the **snmp-server management-url-label** command to specify a label for the URL of the device's management software. Use the **no** form of this command to remove the label.

Syntax Description

<label>	Specifies a label for the URL of the management software (maximum length 255 characters).

Default Values

No default is necessary for this command.

Usage Examples

The following example specifies the label *watch* for the management software:

```
ProCurve(config)#snmp-server management-url-label watch
```

snmp-server source-interface <interface>

Use the **snmp-server source-interface** command to tell the Secure Router OS where to expect Simple Network Management Protocol (SNMP) traps to originate from (interface type). All SNMP originated packets (including traps and get/set requests) will use the designated interface's IP address. Use the **no** form of this command to remove specified interfaces.

Syntax Description

<interface>	Specifies the physical interface that should originate SNMP traps. Enter snmp-server trap-source ? for a complete list of valid interfaces.

Default Values

By default, there are no trap-source interfaces defined.

Usage Examples

The following example specifies that the Ethernet interface (**ethernet 0/1**) should be the source for all SNMP traps and get/set requests:

```
ProCurve(config)#snmp-server source-interface ethernet 0/1
```

snmp-server view <viewname> <oidtree> [excluded | included]

Use the **snmp-server view** command to create or modify a Simple Network Management Protocol (SNMP) view entry. Use the **no** form of this command to remove an entry.

Syntax Description

<viewname>	Specifies label for the view record being created. The name is a record reference.
<oidtree>	Specifies the object identifier (oid) to include or exclude from the view. To identify the subtree, specify a string using numbers, such as 1.4.2.6.8. Replace a single subidentifier with the asterisk (*) to specify a subtree family.
excluded	Specifies an excluded view.
included	Specifies an included view.

Default Values

No default value necessary for this command.

Usage Examples

The **snmp-server view** command can include or exclude a group of OIDs. The following example shows how to create a view (named **blockInterfaces**) to exclude the OID subtree family 1.3.3.1.2.1.2:

```
ProCurve(config)#snmp-server view blockInterfaces 1.3.6.1.2.1.2.* excluded
```

The following example shows how to create a view (named **block**) to include a specific OID:

```
ProCurve(config)#snmp-server view block 1.3.6.1.2.1.2. included
```

sntp server *<address or hostname>* **version** *<1-3>*

Use the **sntp server** command to set the hostname of the SNTP server as well as the version of SNTP to use. The Simple Network Time Protocol (SNTP) is an abbreviated version of the Network Time Protocol (NTP). SNTP is used to set the time of the Secure Router OS product over a network. The SNTP server usually serves the time to many devices within a network.

Syntax Description

<i><address or hostname></i>	Specifies the IP address or hostname of the SNTP server.
version <i><1-3></i>	Specifies which NTP version is used (1-3).

Default Values

By default, version is set to 1.

Usage Examples

The following example sets the SNTP server to **time.nist.gov** using SNTP version 1 (the default version):

```
ProCurve(config)#sntp server time.nist.gov
```

The following example sets the SNTP server as **time.nist.gov**. All requests for time use version 2 of the SNTP:

```
ProCurve(config)#sntp server time.nist.gov version 2
```

spanning-tree edgeport bpdufilter default

Use the **spanning-tree edgeport bpdufilter default** command to enable the BPDU filter on all ports by default. Use the **no** form of this command to disable the setting.

Syntax Description

No subcommands.

Default Values

Disabled by default.

Functional Notes

The BPDU filter blocks any BPDUs from being transmitted and received on an interface. This can be overridden on an individual port.

Usage Examples

The following example enables the bpdufilter on all ports by default:

```
ProCurve(config)#spanning-tree edgeport bpdufilter default
```

To disable the BPDU filter on a specific interface, issue the appropriate commands for the given interface using the following commands as an example:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#spanning-tree bpdufilter disable
```


spanning-tree edgeport bpduguard default

Use the **spanning-tree edgeport bpduguard default** command to enable the BPDU guard on all ports by default. Use the **no** form of this command to disable the setting.

Syntax Description

No subcommands.

Default Values

Disabled by default.

Functional Notes

The bpduguard blocks any BPDUs from being received on an interface. This can be overridden on an individual port.

Usage Examples

The following example enables the BPDU guard on all ports by default.

```
ProCurve(config)#spanning-tree bpduguard default
```

To disable the BPDU guard on a specific interface, issue the appropriate commands for the given interface using the following commands as an example:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#spanning-tree bpduguard disable
```

spanning-tree edgeport default

Use the **spanning-tree edgeport default** command to configure all ports to be edgeports by default. Use the **no** form of this command to disable the setting.

Syntax Description

No subcommands.

Default Values

Disabled by default.

Usage Examples

The following example configures all interfaces running spanning tree to be edgeports by default:

```
ProCurve(config)#spanning-tree edgeport default
```

An individual interface can be configured to not be considered an edgeport. For example:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#no spanning-tree edgeport
```

spanning-tree forward-time <seconds>

Use the **spanning-tree forward-time** command to specify the delay interval (in seconds) when forwarding spanning-tree packets. Use the **no** form of this command to return to the default interval.

Syntax Description

<seconds>	Forward delay interval in seconds (Range: 4 to 30).

Default Values

<seconds>	15 seconds

Usage Examples

The following example sets the forwarding time to 15 seconds:

```
ProCurve(config)#spanning-tree forward-time 15
```

spanning-tree hello-time <seconds>

Use the **spanning-tree hello-time** command to specify the delay interval (in seconds) between hello bridge protocol data units (BPDUs). To return to the default interval, use the **no** form of this command.

Syntax Description

<seconds>	Delay interval (in seconds) between hello BPDUs. Range: 0 to 1,000,000.'

Default Values

<seconds>	2 seconds

Usage Examples

The following example configures a spanning-tree hello-time interval of 10000 seconds:

```
ProCurve(config)#spanning-tree hello-time 10000
```

spanning-tree max-age <seconds>

Use the **spanning-tree max-age** command to specify the interval (in seconds) the spanning-tree will wait to receive Bridge Protocol Data Units (BPDUs) from the root bridge before assuming the network has changed (thus re-evaluating the spanning-tree topology). Use the **no** form of this command to return to the default interval.

Syntax Description

<seconds>	Wait interval (in seconds) between received BPDUs (from the root bridge). Range: 6 to 40.

Default Values

<seconds>	20 seconds

Usage Examples

The following example configures a max-age interval of 45 seconds:

```
ProCurve(config)#spanning-tree max-age 45
```

spanning-tree mode [rstp | stp]

Use the **spanning-tree mode** command to choose a spanning-tree mode of operation.

Syntax Description

rstp	Enables rapid spanning-tree protocol.
stp	Enables spanning-tree protocol.

Default Values

By default, this is set to rstp.

Usage Examples

The following example sets the spanning-tree mode to rapid spanning-tree protocol:

```
ProCurve(config)#spanning-tree mode rstp
```

spanning-tree pathcost method [short | long]

Use the **spanning-tree pathcost** command to select a short or long pathcost method used by the spanning-tree protocol.

Syntax Description

short	Selects a short pathcost method.
long	Selects a long pathcost method.

Default Values

By default, this is set to short.

Usage Examples

The following example designates the spanning-tree protocol to use a long pathcost method:

ProCurve(config)#**spanning-tree pathcost method long**

spanning-tree priority <value>

Use the **spanning-tree priority** command to set the priority for spanning-tree interfaces. The lower the priority value, the higher the likelihood the configured spanning-tree interface will be the root for the bridge group. To return to the default bridge priority value, use the **no** version of this command.

Syntax Description

<value>	Priority value for the bridge interface. Configuring this value to a low number increases the interface's chance of being the root. Therefore, the maximum priority level would be 0. Range: 0 to 65,535.

Default Values

<value>	32768

Usage Examples

The following example sets **spanning-tree priority** to the maximum level:

```
ProCurve(config)#spanning-tree priority 0
```

username <username> password <password>

Use this command to configure the username and password to use for all protocols requiring a username-based authentication system including FTP server authentication, line (login local-user list), and HTTP access.

Syntax Description

<username>	Alphanumeric string up to 30 characters in length (the username is case-sensitive)
<password>	Alphanumeric string up to 30 characters in length (the username is case-sensitive)

Default Values

By default, there is no established username and password.

Functional Notes

All users defined using the **username/password** command are valid for access to the unit using the **login local-userlist** command.

Usage Examples

The following example creates a username of **procurve** with password **procurve**:

```
ProCurve(config)#username procurve password procurve
```

DHCP POOL COMMAND SET

To activate the DHCP Pool mode, enter the **ip dhcp-server pool** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

client-identifier <identifier> [on page 395](#)
client-name <name> [on page 397](#)
default-router <address> <secondary> [on page 398](#)
dns-server <address> <secondary> [on page 399](#)
domain-name <domain> [on page 400](#)
hardware-address <hardware-address> <type> [on page 401](#)
host <address> [<subnet mask> or <prefix length>] [on page 402](#)
lease <days> <hours> <minutes> [on page 403](#)
netbios-name-server <address> <secondary> [on page 404](#)
netbios-node-type <type> [on page 405](#)
network <address> [<subnet mask> or <prefix length>] [on page 406](#)
ntp-server <ip address> [on page 407](#)
option <option value> [ascii | hex | ip] <value> [on page 408](#)
tftp-server <server> [on page 409](#)
timezone-offset <offset> [on page 410](#)

client-identifier <identifier>

Use the **client-identifier** command to specify a unique identifier (in dotted hexadecimal notation) for a Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove a configured client identifier.

Syntax Description

<identifier>	Specify a custom client identifier using a text string (that is converted to a hexadecimal equivalent) or 7 to 28 hexadecimal numbers (with colon delimiters).
OR	
	Specify the hexadecimal Media Access Control (MAC) address including a hexadecimal number added to the front of the MAC address to identify the media type.
	For example, specifying the client-identifier for a MAC address of d217.0491.1150 defines the client identifier as 01:d2:17:04:91:11:50 (where 01 defines the media type as Ethernet).
	For example, a custom client identifier of 0f:ff:ff:ff:51:04:99:a1 may be entered using the <identifier> option.

Default Values

client-id	Optional. By default, the client identifier is populated using the following formula: TYPE: INTERFACE SPECIFIC INFO : MAC ADDRESS Where TYPE specifies the media type in the form of one hexadecimal byte (refer to <i>hardware-address</i> <hardware-address> <type> on page 401 for a detailed listing of media types) and the MAC ADDRESS is the MAC address assigned to the first Ethernet interface in the unit in the form of six hexadecimal bytes. (For units with a single Ethernet interface, the MAC ADDRESS assigned to Ethernet 0/1 is used in this field). INTERFACE SPECIFIC INFO is only used for Frame Relay interfaces and can be determined using the following:

Functional Notes

DHCP clients use client-identifiers in place of hardware addresses. To create the client identifier, begin with the two-digit numerical code representing the media type and append the client's MAC address. For example, a Microsoft client with an Ethernet (01) MAC address d2:17:04:91:11:50 uses a client identifier of 01:d2:17:04:91:11:50.

Usage Examples

The following example specifies the client identifier for a Microsoft client with an Ethernet MAC address of d217.0491.1150:

```
ProCurve(config)#ip dhcp-server pool Microsoft_Clients  
ProCurve(config-dhcp)#client-identifier 01:d2:17:04:91:11:50
```

client-name <name>

Use the **client-name** command to specify the name of a Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove the configured client name.

Syntax Description

<name>	Identifies the DHCP client (example is client1) with an alphanumeric string (up to 32 characters in length).

Note	<i>The specified client name should not contain the domain name.</i>

Default Values

By default, there are no specified client names.

Usage Examples

The following example specifies a client name of **myclient**:

```
ProCurve(config)#ip dhcp-server pool MyPool
```

```
ProCurve(config-dhcp)#client-name myclient
```

default-router <address> <secondary>

Use the **default-router** command to specify the default primary and secondary routers to use for the Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove the configured router.

Syntax Description

<address>	Specifies the address (in dotted decimal notation) of the preferred router on the client's subnet (example: 192.22.4.254).
<secondary>	Optional. Specifies the address (in dotted decimal notation) of the second preferred router on the client's subnet (example: 192.22.4.253).

Default Values

By default, there are no specified default routers.

Functional Notes

When specifying a router to use as the primary/secondary preferred router, verify that the listed router is on the same subnet as the DHCP client. The Secure Router OS allows a designation for two routers, listed in order of precedence.

Usage Examples

The following example configures a default router with address **192.22.4.253** and a secondary router with address **192.22.4.254**:

```
ProCurve(config)#ip dhcp-server pool MyPool
```

```
ProCurve(config-dhcp)#default-router 192.22.4.253 192.22.4.254
```

dns-server *<address>* *<secondary>*

Use the **dns-server** command to specify the default primary and secondary Domain Name System (DNS) servers to use for the Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove the configured DNS server.

Syntax Description

<i><address></i>	Specifies the address (in dotted decimal notation) of the preferred DNS server on the network (example: 192.72.4.254).
<i><secondary></i>	Optional. Specifies the address (in dotted decimal notation) of the second preferred DNS server on the network (example: 192.100.4.253).

Default Values

By default, there are no specified default DNS servers.

Usage Examples

The following example specifies a default DNS server with address **192.72.3.254** and a secondary DNS server with address **192.100.4.253**:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#dns-server 192.72.3.254 192.100.4.253
```

domain-name <domain>

Use the **domain-name** command to specify the domain name for the Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove the configured domain name.

Syntax Description

<name>	Identifies the DHCP client (e.g., adtran.com) with an alphanumeric string (up to 32 characters in length).

Default Values

By default, there are no specified domain names.

Usage Examples

The following example specifies a domain name of **procurve.com**:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#domain-name procurve.com
```


hardware-address *<hardware-address>* *<type>*

Use the **hardware-address** command to specify the name of a Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove the configured client name.

Syntax Description

<hardware-address> Specifies the hardware address (in hexadecimal notation with colon delimiters) of the preferred router on the client's subnet (example d2:17:04:91:11:50).

<type> Optional. Specifies the hardware protocol of the DHCP client.

ethernet

ieee802

<1-21>

The hardware type field can be entered as follows:

Specifies standard Ethernet networks.

Specifies IEEE 802 standard networks.

Enter one of the hardware types listed in RFC 1700.

The valid hardware types are as follows:

1	10 Mb Ethernet
2	Experimental 3 Mb Ethernet
3	Amateur Radio AX.25
4	Proteon ProNET Token Ring
5	Chaos
6	IEEE 802 Networks
7	ARCNET
8	Hyperchannel
9	Lanstar
10	Autonet Short Address
11	LocalTalk
12	LocalNet (IBM PCNet or SYTEK LocalNet)
13	Ultra link
14	SMDS
15	Frame Relay
16	Asynchronous Transmission Mode (ATM)
17	HDLC
18	Fibre Channel
19	Asynchronous Transmission Mode (ATM)
20	Serial Line
21	Asynchronous Transmission Mode (ATM)

Default Values

<type> 1 - 10 Mb Ethernet

Usage Examples

The following example specifies an Ethernet client with a MAC address of **ae:11:54:60:99:10**:

ProCurve(config)#**ip dhcp-server pool MyPool**

ProCurve(config-dhcp)#**hardware-address ae:11:54:60:99:10 Ethernet**

host <address> [<subnet mask> or <prefix length>]

Use the **host** command to specify the IP address and subnet mask for a manual binding to a Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to remove the configured client address.

Syntax Description

<address>	Specifies the IP address (in dotted decimal notation) for a manual binding to a DHCP client.
<subnet mask>	Optional. Specifies the network mask (subnet) for a manual binding to a DHCP client. If the subnet mask is left unspecified, the DHCP server examines its address pools to obtain an appropriate mask. If no valid mask is found in the address pools, the DHCP server uses the Class A, B, or C natural mask.
<prefix length>	Optional. Alternately, the prefix length may be used to specify the number of bits that comprise the network address. The prefix length must be preceded by a forward slash (/). For example, to specify an IP address with a subnet mask of 255.255.0.0, enter /16 after the address.

Default Values

By default, there are no specified host addresses.

Usage Examples

The following examples show two different ways to specify a client with IP address **12.200.5.99** and a 21-bit subnet mask:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#host 12.200.5.99 255.255.248.0
```

or

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#host 12.200.5.99/21
```

lease <days> <hours> <minutes>

Use the **lease** command to specify the duration of the lease for an IP address assigned to a Dynamic Host Configuration Protocol (DHCP) client. Use the **no** form of this command to return to the default lease value.

Syntax Description

<days>	Specifies the duration of the IP address lease in days.
<hours>	Optional. Specifies the number of hours in a lease. You may only enter a value in the hours field if the days field is specified.
<minutes>	Optional. Specifies the number of minutes in a lease. You may only enter a value in the minutes field if the days and hours fields are specified.

Default Values

By default, an IP address lease is one day.

Usage Examples

The following example specifies a lease of **2 days**:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#lease 2
```

The following example specifies a lease of **1 hour**:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#lease 0 1
```

The following example specifies a lease of **30 minutes**:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#lease 0 0 30
```

netbios-name-server *<address>* *<secondary>*

Use the **netbios-name-server** command to specify the primary and secondary NetBIOS Windows Internet Naming Service (WINS) name servers available for use by the Dynamic Host Configuration Protocol (DHCP) clients. Use the **no** form of this command to remove a configured NetBIOS name server.

Syntax Description

<i><address></i>	Specifies the address (in dotted decimal notation) of the preferred NetBIOS WINS name server on the network (example: 192.72.4.254).
<i><secondary></i>	Optional. Specifies the address (in dotted decimal notation) of the second preferred NetBIOS WINS name server on the network (example: 192.100.4.253).

Default Values

By default, there are no configured NetBIOS WINS name servers.

Usage Examples

The following example specifies a primary NetBIOS WINS name server with an IP address of **172.45.6.99** and a secondary with an IP address of **172.45.8.15**:

```
ProCurve(config)#ip dhcp-server pool MyPool
```

```
ProCurve(config-dhcp)#netbios-name-server 172.45.6.99 172.45.8.15
```

netbios-node-type <type>

Use the **netbios-node-type** command to specify the type of NetBIOS node used with Dynamic Host Configuration Protocol (DHCP) clients. Use the **no** form of this command to remove a configured NetBIOS node type.

Syntax Description

<type>	Specifies the NetBIOS node type used with DHCP clients.
	Valid node types are as follows: b-node (1) - Broadcast node p-node (2) - Peer-to-Peer node m-node (4) - Mixed node h-node (8) - Hybrid node (Recommended)
	Alternately, the node type can be specified using the numerical value listed next to the nodes above.

Default Values

<type>	h-node (8) - Hybrid node

Usage Examples

The following example specifies a client's NetBIOS node type as **h-node**:

```
ProCurve(config)#ip dhcp-server pool MyPool
(ProCurve(config-dhcp)#netbios-node-type h-node
```

Alternately, the following also specifies the client's NetBIOS node type as **h-node**:

```
ProCurve(config-dhcp)#netbios-node-type 8
```

network <address> [*<subnet mask>* or *<prefix length>*]

Use the **network** command to specify the subnet number and mask for an Secure Router OS Dynamic Host Configuration Protocol (DHCP) server address pool. Use the **no** form of this command to remove a configured subnet.

Syntax Description

<i><ip address></i>	Specifies the IP address (in dotted decimal notation) of the DHCP address pool.
<i><subnet mask></i>	Optional. Specifies the network mask (subnet) for the address pool. If the subnet mask is left unspecified, the DHCP server uses the Class A, B, or C natural mask.
<i><prefix length></i>	Optional. Alternately, the prefix length may be used to specify the number of bits that comprise the network address. The prefix length must be preceded by a forward slash (/). For example, to specify an IP address with a subnet mask of 255.255.0.0, enter /16 after the address.

Default Values

By default, there are no configured DHCP address pools.

Usage Examples

The following examples show two different ways to configure an address pool subnet of **192.34.0.0** with a 16-bit subnet mask:

```
ProCurve(config)#ip dhcp-server pool MyPool
(ProCurve(config-dhcp)#network 192.34.0.0 255.255.0.0
```

or

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#network 192.34.0.0 /16
```

ntp-server <ip address>

Use the **ntp-server** command to specify the name of the Network Time Protocol (NTP) server published to the client.

Syntax Description

<i><ip address></i>	Specifies the IP address of the NTP server.

Default Values

By default, no NTP server is defined.

Usage Examples

The following example specifies the IP address of the NTP server:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#ntp-server 192.168.1.1
```

option <option value> [ascii | hex | ip] <value>

Use the **option** command to describe a generic DHCP option to be published to the client. The user may specify any number of generic options to be published to the client.

Syntax Description

<option value>	Specifies the value of the generic DHCP option published to the client. Range: 0 to 255.
ascii	Specifies the DHCP option information in ascii format.
hex	Specifies the DHCP option information in hexadecimal format.
ip	Specifies the DHCP option information in IP format.
<value>	Specifies the ASCII, hexadecimal, or IP value. The value for ascii is simple text. The value for hex is an 8-digit hexadecimal number (32 bit). The value for ip is a standard IP address in the format A.B.C.D.

Default Values

No default value necessary for this command.

Usage Examples

The following example publishes DHCP options to the client:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#option 100 ascii ascii_value
ProCurve(config-dhcp)#option 101 hex AB458E80
ProCurve(config-dhcp)#option 102 ip 192.168.1.1
```


tftp-server <server>

Use the **tftp-server** command to specify the IP address or DNS name of the TFTP server published to the client.

Syntax Description

<server>	Specifies the DNS name or dotted notation IP address of the server.

Default Values

By default, no tftp server is defined.

Usage Examples

The following example specifies the IP address of the TFTP server:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#tftp-server 192.168.1.1
```

The following example specifies the DNS name of the TFTP server:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#tftp-server MyServer.procurve.com
```

timezone-offset <offset>

Use the **timezone-offset** command to specify the timezone adjustment (in hours) published to the client.

Syntax Description

<offset>	Specifies the timezone adjustment (in hours) published to the client. Use an integer from -12 to 12.

Default Values

No default value necessary for this command.

Usage Examples

The following example sets the timezone adjustment for the client to -3 hours. For example, if the server time is configured for eastern time and the client is configured for Pacific time, you can set the client timezone adjustment to -3 hours:

```
ProCurve(config)#ip dhcp-server pool MyPool
ProCurve(config-dhcp)#timezone-offset -3
```

HDLC COMMAND SET

To activate the HDLC mode, enter the **interface hdlc** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycyname> [on page 413](#)

backup commands [begin on page 416](#)

bandwidth <value> [on page 432](#)

bridge-group <group#> [on page 433](#)

crypto map <mapname> [on page 434](#)

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password> [on page 436](#)

fair-queue <threshold> [on page 438](#)

hold-queue <queue size> out [on page 439](#)

ip commands [begin on page 440](#)

keepalive <seconds> [on page 455](#)

lldp receive [on page 456](#)

lldp send [management-address l port-description l system-capabilities l system-description l system-name] [on page 457](#)

lldp send-and-receive [on page 458](#)

mtu <size> [on page 459](#)

qos-policy out <mapname> [on page 460](#)

snmp trap link-status [on page 461](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic on an interface. Use the **no** form of this command to remove an access policy association.

Note	<i>Configured access policies will only be active if the ip firewall command has been entered at the Global Configuration mode prompt to enable the SROS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.</i>

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor (all access policy descriptors are case-sensitive).

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**.

Usage Examples

The following example associates the access policy **UnTrusted** (to allow inbound traffic to the Web server) to the HDLC interface labeled 1:

Enable the SROS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list **InWeb**:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access list with the interface:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the SROS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an access policy that uses a configured access list. SROS access policies are used to permit, deny, or manipulate (using **NAT**) data for each physical interface. Each ACP consists of a selector (**access list**) and an action (**allow**, **discard**, **NAT**). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list <access list names> **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet or the Web GUI.

discard list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list *<access list names>* **address** *<IP address>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list *<access list names>* **interface** *<interface>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter Interface Configuration Mode for the desired interface and enter **access policy** *<policy name>*. The following example assigns access policy **MatchAll** to the HDLC interface:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#access-policy MatchAll
```

backup auto-backup

Use the **backup auto-backup** command to configure the sub-interface to automatically attempt a backup upon failure.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically attempt backup upon a failure.

Usage Examples

The following enables automatic backup on the endpoint:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#backup auto-backup
```


backup auto-restore

Use the **backup auto-restore** command to configure the sub-interface to automatically discontinue backup when all network conditions are operational. Use the **no** form of this command to disable the auto-restore feature.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically restore the primary connection when the failure condition clears.

Usage Examples

The following configures the Secure Router OS to automatically restore the primary connection when the failure condition clears:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#backup auto-restore
```

backup backup-delay <seconds>

Use the **backup backup-delay** command to configure the amount of time the router will wait after the failure condition is recognized before attempting to backup the link. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the delay period (in seconds) a failure must be active before the Secure Router OS will enter backup operation on the interface (valid range: 10 to 86400 seconds).

Default Values

By default, the **backup backup-delay** is set to 10 seconds.

Usage Examples

The following configures the Secure Router OS to wait 60 seconds (on an endpoint with an active alarm condition) before attempting backup operation:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#backup backup-delay 60
```

backup call-mode <role>

Use the **backup call-mode** command to specify whether the configured backup interfaces answers or originates (or a combination of both) backup calls. Use the **no** form of this command to return to the default value.

Syntax Description

<role>	Selects the role the router will take in backup of this sub-interface.
answer	Answers and backs up primary link on failure.
answer-always	Answers and backs up regardless of primary link state.
originate	Originates backup call on primary link failure.
originate-answer	Originates or answers call on primary link failure.
originate-answer-always	Originates on failure; answers and backs up always.

Default Values

By default, the **backup call-mode** is set to **originate-answer**.

Functional Notes

The majority of the configuration for HDLC backup is configured via the HDLC interface configuration commands. However, the numbers dialed are configured in the primary interface. Full sample configurations follow:

Sample config for remote router (dialing out)

```
hostname "Remote7203dl"
enable password password
!
interface eth 0/1
 ip address 192.168.1.254 255.255.255.0
 no shutdown
!
interface modem 1/3
 no shutdown
!
interface t1 1/1
 coding b8zs
 framing esf
 clock source line
 tdm-group 1 timeslots 1-24
 no shutdown
!
interface fr 1 point-to-point
```

```
frame-relay lmi-type ansi
no shutdown
bind 1 t1 1/1 1 fr 1
!
interface fr 1.16 point-to-point
frame-relay interface-dlci 16
ip address 10.1.1.2 255.255.255.252
backup call-mode originate
backup number 5551111 analog
backup number 5552222 analog
!
ip route 0.0.0.0 0.0.0.0 10.1.1.1
!
line telnet 0 4
password password
```

Sample config for central router (dialing in)

```
hostname "Central7203dl"
enable password password
!
interface eth 0/1
ip address 192.168.100.254 255.255.255.0
no shutdown
!
interface modem 1/3
no shutdown
!
interface t1 1/1
coding b8zs
framing esf
clock source line
tdm-group 1 timeslots 1-24
no shutdown
!
interface fr 1 point-to-point
frame-relay lmi-type ansi
no shutdown
bind 1 t1 1/1 1 fr 1
!
interface fr 1.100 point-to-point
frame-relay interface-dlci 100
ip address 10.1.1.1 255.255.255.252
```

```
backup call-mode answer
backup number 555-8888 analog
!
line telnet 0 4
password password
```

Usage Examples

The following configures the Secure Router OS to answer backup calls on this endpoint but never generate calls:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#backup call-mode answer-always
```

backup connect-timeout <seconds>

Use the **backup connect-timeout** command to specify the number of seconds to wait for a connection after a call is attempted before trying to call again or dialing a different number. Recommended value is greater than 60.

Syntax Description

<seconds>	Selects the amount of time in seconds that the router will wait for a connection before attempting another call (valid range: 10 to 300).

Default Values

By default, **backup connect-timeout** is set to 60 seconds.

Usage Examples

The following configures the Secure Router OS to wait 120 seconds before retrying a failed backup call:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#backup connect-timeout 120
```

backup force <state>

Use the **backup force** command to manually override the automatic backup feature. This can be used to force a link into backup to allow maintenance to be performed on the primary link without disrupting data. Use the **no** form of this command to return to the normal backup operation state.

Syntax Description

<state>	Selects the forced backup state of the sub-link.
backup	Forces backup regardless of primary link state.
primary	Forces primary link regardless of its state.

Default Values

By default, this feature is disabled.

Usage Examples

The following configures the Secure Router OS to force this endpoint into backup:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#backup force backup
```

backup maximum-retry <attempts>

Use the **backup maximum-retry** command to select the number of calls the router will make when attempting to backup a link. Use the **no** form of this command to return to the default state. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

<attempts>	Selects the number of call retries that will be made after a link failure (valid range: 0 to 15).
	Setting this value to 0 will allow unlimited retries during the time the network is failed.

Default Values

By default, **backup maximum-retry** is set to 0 attempts.

Usage Examples

The following example configures the Secure Router OS to retry a dial-backup call four times before considering backup operation not available:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#backup maximum-retry 4
```


backup number <digits> [**analog** | **digital-56k** | **digital 64k**] <isdn min chan> <isdn max chan> <interface>

Use the **backup number** command to configure the phone number and the call type the router will dial upon network failure. Multiple entries can be made for an interface to allow alternate sites to be dialed. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode* <role> [on page 419](#).

Syntax Description

<digits>	Specifies the phone numbers to call when the backup is initiated.
analog	Indicates number connects to an analog modem.
digital-56k	Indicates number belongs to a digital 56 kbps per DS0 connection.
digital-64k	Indicates number belongs to a digital 64 kbps per DS0 connection.
<isdn min chan>	Specifies the minimum number of DS0s required for a digital 56 or 64 kbps connection (Range: 1 to 24).
<isdn mas chan>	Specifies the maximum number of DS0s desired for a digital 56 or 64 kbps connection (Range: 1 to 24).
<interface>	Specifies the Frame Relay sub-interface (e.g., hdlc 1) to use when originating or answering using this number.

Default Values

By default, there are no configured **backup numbers**.

Usage Examples

The following example configures the Secure Router OS to dial 704-555-1212 (digital 64 kbps connection) to initiate backup operation on this endpoint using the configured HDLC interface hdlc 1:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#backup number 7045551212 digital-64k 1 1 hdlc 1
```

backup priority <value>

Use the **backup priority** command to select the backup priority for this interface. This allows the user to establish the highest priority backup link and ensure that link will override backups attempted by lower priority links. Use the **no** form of this command to return to the default value. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

<value>	Sets the relative priority of this link (valid range: 0 to 100). A value of 100 designates the highest priority.

Default Values

By default, **backup priority** is set to 50.

Usage Examples

The following example assigns the highest priority to this endpoint:

```
ProCurve(config)#interface hdlc 16
ProCurve(config-hdlc 16)#backup priority 100
```

backup randomize-timers

Use the **backup randomize-timers** command to randomize the call timers to minimize potential contention for resources. Use the **no** form of this command to return to the default value. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

No subcommands.

Default Values

By default, the Secure Router OS does not randomize the backup call timers.

Usage Examples

The following example configures the Secure Router OS to randomize the backup timers associated with this endpoint:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#backup randomize-timers
```

backup redial-delay <seconds>

Use the **backup redial-delay** command to configure the delay after an unsuccessful call until the call will be re-tried. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

<seconds>	Specifies the delay in seconds between attempting to re-dial a failed backup attempt. Range: 10 to 3600.

Default Values

By default, **backup redial-delay** is set to 10 seconds.

Usage Examples

The following example configures a redial delay of 25 seconds on this endpoint:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#backup redial-delay 25
```

backup restore-delay <seconds>

Use the **backup restore-delay** command to configure the amount of time the router will wait after the network is restored before disconnecting the backup link and reverting to the primary. This setting is used to prevent disconnecting the backup link if the primary link is “bouncing” in and out of alarm. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

<seconds>	Specifies the number of seconds the Secure Router OS will wait (after a primary link is restored) before disconnecting backup operation. Range: 10 to 86,400.

Default Values

By default, **backup restore-delay** is set to 10 seconds.

Usage Examples

The following example configures the Secure Router OS to wait 30 seconds before disconnecting backup operation and restoring the primary connection for this endpoint:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#backup restore-delay 30
```

backup schedule [day | enable-time | disable-time]

Use the **backup schedule** command to set the time of day that backup will be enabled. Use this command if backup is desired only during normal business hours and on specific days of the week. Use the **no** form of this command to disable backup (as specified). For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

day	Sets the days to allow backup (valid range: Monday through Sunday).
enable-time	Sets the time of day to enable backup. Time is entered in 24-hour format (00:00).
disable-time	Sets the time of day to disable backup.

Default Values

By default, backup is enabled for all days and times if the **backup auto-backup** command has been issued and the backup schedule has not been entered.

Usage Examples

The following example enables backup Monday through Friday 8:00 am to 7:00 pm:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdLC 1)#backup schedule enable-time 08:00
ProCurve(config-hdLC 1)#backup schedule disable-time 19:00
ProCurve(config-hdLC 1)#no backup schedule day Saturday
ProCurve(config-hdLC 1)#no backup schedule day Sunday
```

backup shutdown

Use the **backup shutdown** command to deactivate all backup functionality in the unit. Backup configuration parameters are kept intact, but the unit will not initiate (or respond) to backup sequences in the event of a network outage. Use the **no** form of this command to reactivate the backup interface. For more detailed information on HDLC backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 419](#).

Syntax Description

No subcommands.

Default Values

By default, all Secure Router OS interfaces are disabled.

Usage Examples

The following example deactivates the configured backup interface:

```
ProCurve(config)#interface hdlc 16  
ProCurve(config-hdlc 16)#backup shutdown
```

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Enter bandwidth in kbps.

Default Values

To view default values use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the HDLC interface to 10 Mbps:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#bandwidth 10000
```


bridge-group <group#>

Use the **bridge-group** command to assign an interface to the specified bridge group. Use the **no** form of this command to remove the interface from the bridge group.

Syntax Description

<group#>	Specifies bridge group number (1 to 255) specified using the bridge-group command.

Default Values

By default, there are no configured bridge groups.

Functional Notes

A bridged network can provide excellent traffic management to reduce collisions and limit the amount of bandwidth wasted with unnecessary transmissions when routing is not necessary. Any two interfaces can be bridged (Ethernet to T1 bridge, Ethernet to Frame Relay sub-interface, etc.).

Usage Examples

The following example assigns the HDLC interface labeled 1 to bridge-group 1:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#bridge-group 1
```

crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the technical support note **Configuring VPN** located on the ProCurve SROS documentation CD provided with your unit.*

Syntax Description

<mapname> Enter the crypto map name that you wish to assign to the interface.

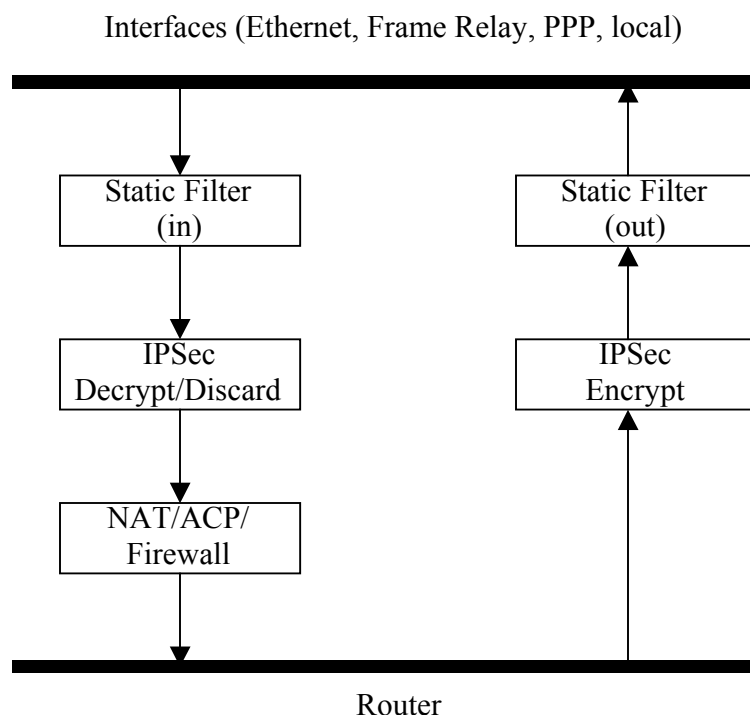
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following notes in mind.

When defining the policy-class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the un-encrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical SROS data-flow logic.



As shown in the previous diagram, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access-group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only un-encrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy-class on an interface. When specifying the ACLs for a crypto map, the source information is the private local-side, un-encrypted source of the data. The destination information will be the far-end, un-encrypted destination of the data. However, ACLs for a policy-class work in reverse. The source information for the ACL in a policy-class is the far-end. The destination information is the local-side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the HDLC 1 interface:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#crypto map MyMap
```

**dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname>
<username> <password>**

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

See **Functional Notes** below for syntax descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A web-based interface provides two levels of control over your domain, catering to average or power users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs, and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service, in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

If your IP address doesn't change often or at all, but you still want an easy name to remember it by (without having to purchase your own domain name) Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com) you need Custom DNS service which also provides full dynamic and static IP address support.

Usage Examples

The following example sets the dynamic-dns to **dyndns-custom** with hostname **host**, username **user**, and password **pass**:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#dynamic-dns dyndns-custom host user pass
```

fair-queue *<threshold>*

Use the **fair-queue** command to enable weighted fair queuing (WFQ) on an interface. Use the **no** form of this command to disable WFQ and enable FIFO (first-in-first-out) queueing for an interface. WFQ is enabled by default for WAN interfaces.

Syntax Description

<i><threshold></i>	Optional. Value that specifies the maximum number of packets that can be present in each conversation sub-queue. Packets received for a conversation after this limit is reached are discarded. Range: 16 to 512.

Default Values

By default, fair-queue is enabled with a threshold of 64 packets.

Usage Examples

The following example enables WFQ on the interface with a threshold set at 100 packets:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#fair-queue 100
```

hold-queue *<queue size>* **out**

Use the **hold-queue** command to change the overall size of an interface's WAN output queue.

Syntax Description

<i><queue size></i>	The total number of packets the output queue can contain before packets are dropped. Range: 16-1000.

Default Values

The default queue size for WFQ is 400. The default queue size for PPP FIFO and Frame Relay round-robin is 200.

Usage Examples

The following example sets the overall output queue size to 700:

```
ProCurve(config)#interface hdlc 1
```

```
ProCurve(config-hdlc 1)#hold-queue 700
```

ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<listname>	Assigned IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the unit to only allow Telnet traffic (as defined in the user-configured **TelnetOnly** IP access list) into the HDLC interface:

```
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#int hdlc 1
ProCurve(config-hdlc 1)#ip access-group TelnetOnly in
```


ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface. Use the optional **secondary** keyword to define a secondary IP address. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional. Keyword used to configure a secondary IP address for the specified interface.

Default Values

By default, there are no assigned IP addresses.

Functional Notes

Use secondary IP addresses to allow dual subnets on a single interface (when you need more IP addresses than the primary subnet can provide). When using secondary IP addresses, avoid routing loops by verifying that all devices on the network segment are configured with secondary IP addresses on the secondary subnet.

Usage Examples

The following example configures a secondary IP address of **192.22.72.101/30**:

```
ProCurve(config)#hdlc 1
```

```
ProCurve(config-hdlc 1)#ip address 192.22.72.101 255.255.255.252 secondary
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note	<i>The ip helper command must be used in conjunction with the ip forward-protocol command to configure the Secure Router OS to forward UDP broadcast packets.</i>

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign a helper-address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is User Datagram Protocol (UDP).
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip helper-address 192.33.5.99
```

ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

immediate-leave	If only one host (or IGMP snooping switch) is connected to the interface, when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	This command controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer, if no receiver responds, the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65535 ms. Default: 1000 ms.
querier-timeout <seconds>	Number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60-300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically-connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip igmp last-member-query-interval 200
```

ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface and place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding.

Usage Examples

The following example enables multicast forwarding and IGMP on the HDLC interface:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip mcast-stub downstream
```

ip mcast-stub helper-enable

Use the **ip mcast-stub helper-enable** command to assign the **ip mcast-stub helper-address** as the IGMP proxy. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address**, **ip mcast-stub upstream**, and **ip mcast-stub downstream** commands. When enabled, the interface becomes a helper forwarding interface. The IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the unit to perform as an IGMP proxy.

Usage Examples

The following example sets the helper address as the IGMP proxy:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#ip mcast-stub helper-enable
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface.

Usage Examples

The following example enables multicast forwarding on the HDLC interface:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip mcast-stub upstream
```

ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key	Specifies a simple-text authentication password to be used by other routers using <i><password></i> the OSPF simple password authentication.
cost <i><value></i>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65535.
dead-interval <i><seconds></i>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, neighboring devices will determine that the device is down. Range: 0 to 32767.
hello-interval <i><seconds></i>	Specifies the interval between hello packets sent on the interface. Range: 0 to 32767.
message-digest-key <i><keyid></i> md5 <i><key></i>	Configures OSPF Message Digest 5 (MD5) authentication (16-byte max) keys.
priority <i><value></i>	Set the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <i><seconds></i>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32767.
transmit-delay <i><seconds></i>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32767.

Default Values

retransmit-interval <i><seconds></i>	5 seconds
transmit-delay <i><seconds></i>	1 second
hello-interval <i><seconds></i>	10 seconds: Ethernet, point-to-point, Frame Relay, and PPP
dead-interval <i><seconds></i>	40 seconds

Usage Example

The following example sets the maximum number of seconds allowed between hello packets to 25,000:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip ospf dead-interval 25000
```

ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Optional. Select message-digest authentication type.
null	Optional. Select for no authentication to be used.

Default Values

By default, this is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the HDLC interface:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#ip ospf authentication null
```


ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Set the network type for broadcast.
point-to-point	Set the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP and Frame Relay default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#ip ospf network broadcast
```

ip proxy-arp <ip address> <subnet mask>

Use the **ip proxy-arp** command to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<ip address>	Defines the proxy ARP IP address in dotted decimal notation (for example: 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, proxy-arp is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the SROS will respond to all proxy ARP requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the HDLC interface:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip proxy-arp
```

ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Only accept received RIP version 1 packets on the interface.
2	Only accept received RIP version 2 packets on the interface.

Default Values

By default, all interfaces implement RIP version 1 (the default value for the version command).

Functional Notes

Use the **ip rip receive version** command to specify a RIP version that overrides the **version** (in the Router RIP) configuration. See *version [1 | 2]* [on page 969](#) for more information.

The SROS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the HDLC interface to accept only RIP version 2 packets:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip rip receive version 2
```

ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Only transmits RIP version 1 packets on the interface.
2	Only transmits RIP version 2 packets on the interface.

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** command to specify a RIP version that overrides the **version** (in the Router RIP) configuration. See *version [1 | 2]* [on page 969](#) for more information.

The SROS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the HDLC interface to transmit only RIP version 2 packets:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip rip send version 2
```

ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note *Using Network Address Translation (NAT) or the SROS firewall capabilities on an interface requires process switching mode (using the **no ip route-cache** command).*

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on the HDLC interface:

```
ProCurve(config)#interface hdlc 1
(config-hdlc 1)#ip route-cache
```

ip unnumbered <interface>

Use the **ip unnumbered** command to use the IP address assigned to the specified interface for all IP processing on the active interface. Use the **no** form of this command to remove the unnumbered configuration.

Syntax Description

<interface>	Specifies the interface (in the format type slot/port) that contains the IP address to use as the source address for all packets transmitted on this interface.

Default Values

By default, all interfaces are configured to use a specified IP address (using the **ip address** command).

Functional Notes

If **ip unnumbered** is enabled on an interface, all IP traffic from the interface will use a source IP address taken from the specified interface. For example, specifying **ip unnumbered eth 0/1** while in the Frame Relay Sub-Interface Configuration mode configures the Frame Relay sub-interface to use the IP address assigned to the Ethernet interface for all IP processing. In addition, the SROS uses the specified interface information when sending route updates over the unnumbered interface.

Usage Examples

The following example configures the HDLC interface to use the IP address assigned to the Ethernet interface (**eth 0/1**):

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#ip unnumbered eth 0/1
```

keepalive <seconds>

Use the **keepalive** command to enable the transmission of keepalive packets on the interface and specify the time interval in seconds between transmitted packets.

Syntax Description

<seconds>	Defines the time interval (in seconds) between transmitted keepalive packets (valid range: 0 to 32,767 seconds).

Default Values

By default, the time interval between transmitted keepalive packets is 10 seconds.

Functional Notes

If three keepalive packets are sent to an interface with no response, the interface is considered down. To detect interface failures quickly, specify a smaller keepalive time.

Usage Examples

The following example specifies a keepalive time of 5 seconds on the HDLC interface:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#keepalive 5
```

lldp receive

Use the **lldp receive** command to allow LLDP packets to be received on this interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to send and receive LLDP packets.

Usage Examples

The following example configures the HDLC interface to receive LLDP packets:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#lldp receive
```


lldp send [management-address | port-description | system-capabilities | system-description | system-name]

Use the **lldp send** command to configure this interface to transmit LLDP packets or to control the types of information contained in the LLDP packets transmitted by this interface.

Syntax Description

management-address	Enables transmission of management address information on this interface.
port-description	Enables transmission of port description information on this interface.
system-capabilities	Enables transmission of this device's system capabilities on this interface.
system-description	Enables transmission of this device's system description on this interface.
system-name	Enables transmission of this device's system name on this interface.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the HDLC interface to transmit LLDP packets containing all enabled information types:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#lldp send
```

lldp send-and-receive

Use the **lldp send-and-receive** command to configure this interface to transmit and receive LLDP packets.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the HDLC interface (hdlc 16) to transmit and receive LLDP packets containing all information types:

```
ProCurve(config)#interface hdlc 16  
ProCurve(config-hdlc 16)#lldp send-and-receive
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit (MTU) size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size> Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:

ATM interfaces	64 to 1520
Ethernet interfaces	64 to 1500
FDL interfaces	64 to 256
HDLC interfaces	64 to 1520
Loopback interfaces	64 to 1500
Tunnel interfaces	64 to 18,190
Virtual Frame Relay sub-interfaces	64 to 1520
Virtual PPP interfaces	64 to 1500

Default Values

<size> The default values for the various interfaces are listed below:

ATM interfaces	1500
Ethernet interfaces	1500
FDL interfaces	256
HDLC interfaces	1500
Loopback interfaces	1500
Tunnel interfaces	1500
Virtual Frame Relay sub-interfaces	1500
Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the HDLC interface:

```
ProCurve(config)#interface hdlc 1
ProCurve(config-hdlc 1)#mtu 1200
```

qos-policy out <mapname>

Use the **qos-policy out** command to apply a previously-configured QoS map to an interface. Use the **no** form of this command to remove the map from the interface. The **out** keyword specifies that this policy will be applied to outgoing packets.

Syntax Description

<mapname>	Specifies the name of a previously-created QoS map.

Default Values

No default value is necessary for this command.

Usage Examples

The following example applies the QoS map **VOICEMAP** to the HDLC interface:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#qos-policy out VOICEMAP
```

snmp trap link-status

Use the **snmp trap link-status** command to control the SNMP variable *ifLinkUpDownTrapEnable* (RFC2863), which enables (or disables) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the HDLC interface:

```
ProCurve(config)#interface hdlc 1  
ProCurve(config-hdlc 1)#no snmp trap link-status
```

IKE POLICY COMMAND SET

To activate the IKE Policy mode, enter the **crypto ike policy** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#crypto ike policy 1
ProCurve(config-ike)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

attribute <polycynumber> [on page 463](#)
client authentication host [on page 464](#)
client authentication server list <listname> [on page 466](#)
client configuration pool <poolname> [on page 467](#)
initiate [aggressive | main] [on page 468](#)
local-id [address | asn1-dn | fqdn | user-fqdn] <ipaddress or name> [on page 469](#)
nat-traversal <version> [allow | disable | force] [on page 470](#)
peer [<ip address> | any] [on page 471](#)
respond [aggressive | anymode | main] [on page 472](#)

Note	<i>For VPN configuration example scripts, refer to the VPN Configuration Guide located on the ProCurve SROS Documentation CD provided with your unit.</i>

attribute <polycynumber>

Use the **attribute** command to define attributes for the associated IKE policy. Multiple attributes can be created for a single IKE policy. Once you enter this command, you are in the IKE Policy Attribute. Refer to *IKE Policy Attributes Command Set* on page 473 for more information.

Syntax Description

<polycynumber>	Assigns a number (range: 1 to 65,535) to the attribute policy. The number is the attribute's priority number and specifies the order in which the resulting VPN proposals get sent to the far end.

This command takes you to the **(config-ike-attribute)#** prompt. From here, you can configure the settings for the attribute as outlined in the section *IKE Policy Attributes Command Set* on page 473.

Default Values

By default, no attribute is defined.

Functional Notes

Multiple attributes on an IKE policy are ordered by number (with the lowest number representing the highest priority).

Usage Examples

The following example defines a policy attribute (**10**) and takes you into the IKE Policy Attributes:

```
ProCurve(config-ike)#attribute 10
ProCurve(config-ike)#
```

client authentication host

Use the **client authentication host** command to enable the unit to act as an Xauth host when this IKE policy is negotiated with a peer.

Variations of this command include the following:

client authentication host username <username>

client authentication host username <username> **password** <word>

client authentication host username <username> **password** <word> **passphrase** <phrase>

Syntax Description

username <username> Enter the value sent via Xauth as the username.

password <word> Enter the value sent via Xauth as the password.

passphrase <phrase> Optional. Enter the value sent via Xauth as the passphrase. This is only used with authentication type OTP (one time password).

Default Values

By default, if this command is not present in the IKE policy the unit does not act as an Xauth host.

Functional Notes

The specified credentials are programmed into the unit and there is no prompt for entering values real time. Therefore, schemes requiring real time input or additional responses (e.g., SecureID) are not supported. The **client authentication host** command and the **client authentication server** commands are mutually exclusive. See *client authentication server list* <listname> on page 466 for more information.

Usage Examples

The following example specifies the login credentials to be sent:

```
ProCurve(config-ike)#client authentication host username jsmith password password1 passphrase phrase
```


client authentication host xauth-type [generic | otp | radius]

Use the **client authentication host xauth-type** command to allow the user to specify the Xauth authentication type if a type other than **generic** is desired.

Syntax Description

generic	Specifies generic authentication type.
otp	Specifies OTP authentication type.
radius	Specifies RADIUS authentication type.

Default Values

By default, this is set to generic.

Functional Notes

This command is used along with the **client authentication host username**. See *client configuration pool <poolname>* on page 467 for more information. When acting as an Xauth host, this command allows the user to specify the Xauth authentication type if a type other than generic is desired.

Usage Examples

The following example sets the Xauth type to **radius**:

```
ProCurve(config-ike)#client authentication host xauth-type radius
```

client authentication server list <listname>

Use the **client authentication server list** command to enable the unit to act as an Xauth server (edge device).

Syntax Description

<listname>	Specifies the named list created with the aaa authentication login command.

Default Values

By default, the router does not act as an Xauth server and extended authentication is not performed.

Functional Notes

When this IKE policy is negotiated and the peer has indicated Xauth via the IKE authentication method and/or the Xauth vendor ID, this command allows the unit to perform as an Xauth server (edge device). The specified AAA login method is used to identify the location of the user authentication database. The **client authentication host** and the **client authentication server** commands are mutually exclusive. See *client configuration pool <poolname>* on page 467 for more information.

Usage Examples

The following example enables Xauth as an Xauth server and specifies which AAA method list to use in locating the user database:

```
ProCurve(config-ike)#client authentication server list clientusers
```

client configuration pool <poolname>

Use the **client configuration pool** command to configure the Secure Router OS to perform as mode-config server (edge device) when an IKE policy is negotiated.

Variations of this command include the following:

```
client configuration pool <poolname>
client configuration pool <poolname> initiate
client configuration pool <poolname> initiate respond
client configuration pool <poolname> respond
client configuration pool <poolname> respond initiate
```

Syntax Description

<poolname>	The pool from which to obtain parameters to assign to the client.

Default Values

By default, if this command is not present in the IKE policy, the device allocates mode-config IP addresses, DNS server addresses, and NetBIOS name server addresses, and mode-config is not performed.

Functional Notes

This command ties an existing client configuration pool to an IKE policy.

Usage Examples

The following example ties the **ConfigPool1** configuration pool to this IKE policy:

```
ProCurve(config-ike)#client configuration pool ConfigPool
```

initiate [aggressive | main]

Use the **initiate** command to allow the IKE policy to initiate negotiation (in main mode or aggressive mode) with peers. Use the **no** form of this command to allow the policy to respond only.

Syntax Description

aggressive	Specifies to initiate using aggressive mode. Aggressive mode can be used when one end of the VPN tunnel has a dynamically assigned address. The side with the dynamic address must be the initiator of the traffic and tunnel. The side with the static address must be the responder.
main	Specifies to initiate using main mode. Main mode requires that each end of the VPN tunnel has a static WAN IP address. Main mode is more secure than aggressive mode because more of the main mode negotiations are encrypted.

Default Values

By default, the **main** initiation is enabled.

Functional Notes

By using the **initiate** and **respond** commands, you can configure the IKE policy to initiate and respond, initiate only, or respond only. It is an error if you have both **initiate** and **respond** disabled.

Usage Examples

The following example enables the Secure Router OS device to initiate IKE negotiation in main mode:

```
ProCurve(config-ike)#initiate main
```

local-id [address | asn1-dn | fqdn | user-fqdn] <ipaddress or name>

Use the **local-id** command to set the local ID for the IKE policy. This setting overrides the system local ID setting (set in the Global configuration mode using the **crypto ike local-id address** command).

Syntax Description

address <ipaddress>	Specifies a remote ID of IPv4 type.
asn1-dn <name>	Specifies an Abstract Syntax Notation Distinguished Name as the remote ID (enter this value in LDAP format).
fqdn <name>	Specifies a fully qualified domain name as the remote ID.
user-fqdn <name>	Specifies a user fully qualified domain name or email address (e.g., user1@hp.com) as the remote ID.

Default Values

By default, the local ID is not defined.

Functional Notes

The local ID for a particular IKE policy can be set in two ways. The first (default) method is to use the global system command:

ProCurve(config)#**crypto ike local-id address**

This command, which by default is executed on start-up, makes the local ID of an IKE policy equal to the IPv4 address of the interface on which an IKE negotiation is occurring. This is particularly useful for products that could have multiple public interfaces.

The second method is to use the IKE policy command:

ProCurve(config-ike)#**local-id [address | fqdn | user-fqdn] <ipaddress or fqdn>**

This policy-specific command allows you to manually set the local ID for an IKE policy on a per-policy basis. You can use both methods simultaneously in the product. Several IKE policies can be created, some of which use the default system setting of the IPv4 address of the public interface. Others can be set to override this system setting and manually configure a local ID specific to those policies. When a new IKE policy is created, they default to **no local-id**. This allows the system local ID setting to be applied to the policy.

Usage Examples

The following example sets the local ID of this IKE policy to the IPv4 address 63.97.45.57:

ProCurve(config-ike)#**local-id address 63.97.45.57**

nat-traversal <version> [allow | disable | force]

Use the **nat-traversal** command to allow, force, or disable NAT traversal version 1 and 2 on a specific Ike policy.

Syntax Description

<version>	Enter v1 or v2 to select the NAT traversal version.
allow	Sets the IKE policy to allow the specified NAT traversal version.
disable	Sets the IKE policy to disable the specified NAT traversal version.
force	Sets the IKE policy to force the specified NAT traversal version.

Default Values

*The defaults for this command are **nat-traversal v1 allow** and **nat-traversal v2 allow**.*

Usage Examples

The following example disables version 2 on Ike policy 1:

```
ProCurve(config)#crypto ike policy 1
```

```
ProCurve(config-ike)#nat-traversal v2 disable
```

peer [*<ip address>* | any]

Use the **peer** command to enter the IP address of the peer device. Repeat this command for multiple peers. Use the **any** keyword if you want to set up a policy that will initiate or respond to any peer.

Syntax Description

<i><ip address></i>	Specifies a peer IP address.
any	Allows any peer to connect to this IKE policy.

Default Values

There are no default settings for this command.

Functional Notes

An IKE policy is incomplete unless one of the peer commands is specified. Only one IKE policy can be configured with **peer any**.

Usage Examples

The following example sets multiple peers on an IKE policy for an initiate-and-respond policy using pre-shared secret, DES, MD5, and Diffie-Hellman Group 1:

```
ProCurve(config)#crypto ike policy 100
ProCurve(config-ike)#peer 63.97.45.57
ProCurve(config-ike)#peer 63.105.15.129
ProCurve(config-ike)#peer 192.168.1.3
ProCurve(config-ike)#respond anymode
ProCurve(config-ike)#initiate main
```

The following example sets up a policy allowing any peer to initiate using pre-shared secret, DES, MD5, and Diffie-Hellman Group 1.

```
ProCurve(config)#crypto ike policy 100
ProCurve(config-ike)#peer any
ProCurve(config-ike)#respond anymode
ProCurve(config-ike)#initiate main
```

Technology Review

IKE policies must have a peer address associated with them to allow certain peers to negotiate with the product. This is a problem when you have "roaming" users (those who obtain their IP address using DHCP or some other dynamic means). To allow for "roaming" users, the IKE policy can be set up with **peer any** to allow any peer to negotiate with the product. There can only be one **peer any** policy in the running configuration.

respond [aggressive | anymode | main]

Use the **respond** command to allow the IKE policy to respond to negotiations by a peer. Use the **no** form of this command to allow the policy to only initiate negotiations.

Syntax Description

aggressive	Specifies to respond only to aggressive mode.
anymode	Specifies to respond to any mode.
main	Specifies to respond only to main mode.

Default Values

By default, respond to any mode is enabled.

Functional Notes

By using the **initiate** and **respond** commands, you can configure the IKE policy to initiate and respond, initiate only, or respond only. It is an error if you have both **initiate** and **respond** disabled.

Usage Examples

The following example configures the router to initiate and respond to IKE negotiations:

```
ProCurve(config-ike)#respond anymode  
ProCurve(config-ike)#initiate main
```

IKE POLICY ATTRIBUTES COMMAND SET

To activate the IKE Policy Attributes mode, enter the **attribute** command at the IKE Policy prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#crypto ike policy 1
ProCurve(config-ike)#attribute 10
ProCurve(config-ike-attribute)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

authentication [*dss-sig* | *pre-share* | *rsa-sig*] [on page 474](#)

encryption [*aes-xxx-cbc* | *des* | *3des*] [on page 475](#)

group [*1* | *2*] [on page 476](#)

hash [*md5* | *sha*] [on page 477](#)

lifetime <*seconds*> [on page 478](#)

Note	<i>For VPN configuration example scripts, refer to the VPN Configuration Guide located on the ProCurve SROS Documentation CD provided with your unit.</i>

authentication [dss-sig | pre-share | rsa-sig]

Use the **authentication** command to configure this IKE policy's use of pre-shared secrets and signed certificates during IKE negotiation.

Syntax Description

dss-sig	Specifies to use DSS-signed certificates during IKE negotiation to validate the peer.
pre-share	Specifies the use of pre-shared secrets during IKE negotiation to validate the peer.
rsa-sig	Specifies to use RSA-signed certificates during IKE negotiation to validate the peer.

Default Values

By default, this command is enabled.

Functional Notes

Both sides must share the same pre-shared secret in order for the negotiation to be successful.

Usage Example

The following example enables pre-shared secrets for this IKE policy:

```
ProCurve(config-ike-attribute)#authentication pre-share
```

encryption [aes-xxx-cbc | des | 3des]

Use the **encryption** command to specify which encryption algorithm this IKE policy will use to transmit data over the IKE-generated SA.

Syntax Description

aes-128-cbc	Specifies the AES 128 CBC encryption algorithm.
aes-192-cbc	Specifies the AES 192 CBC encryption algorithm.
aes-256-cbc	Specifies the AES 256 CBC encryption algorithm.
des	Specifies the DES encryption algorithm.
3des	Specifies the 3DES encryption algorithm.

Default Values

By default, encryption is set to des.

Usage Examples

The following example selects 3DES as the encryption algorithm for this IKE policy:

ProCurve(config-ike-attribute)#**encryption 3des**

group [1 | 2]

Use the **group** command to specify the Diffie-Hellman Group (1 or 2) to be used by this IKE policy to generate the keys (which are then used to create the IPSec SA).

Syntax Description

1	Specifies 768-bit mod P.
2	Specifies 1024-bit mod P.

Default Values

By default, group is set to 1.

Functional Notes

The local IKE policy and the peer IKE policy must have matching group settings in order for negotiation to be successful.

Usage Examples

The following example sets this IKE policy to use Diffie-Hellman Group 2:

```
ProCurve(config-ike-attribute)#group 2
```

hash [md5| sha]

Use the **hash** command to specify the hash algorithm to be used to authenticate the data transmitted over the IKE SA.

Syntax Description

md5	Choose the MD5 hash algorithm.
sha	Choose the SHA hash algorithm.

Default Values

By default, hash is set to **sha**.

Usage Examples

The following example specifies **md5** as the hash algorithm:

```
ProCurve(config-ike-attribute)#hash md5
```

lifetime *<seconds>*

Use the **lifetime** command to specify how long an IKE SA is valid before expiring.

Syntax Description

<i><seconds></i>	Specifies how many seconds an IKE SA will last before expiring.

Default Values

By default, lifetime is set to 28,800 seconds.

Usage Examples

The following example sets a lifetime of two hours:

```
ProCurve(config-ike-attribute)#lifetime 7200
```

IKE CLIENT COMMAND SET

To activate the IKE Client mode, enter the **crypto ike client** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#crypto ike client configuration pool
ProCurve(config-ike-client-pool)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

dns-server <address1> <address2> [on page 480](#)

ip-range <start ip> <end ip> [on page 481](#)

netbios-name-server <address1> <address2> [on page 482](#)

Note	<i>For VPN configuration example scripts, refer to the VPN Configuration Guide located on the ProCurve SROS Documentation CD provided with your unit.</i>

dns-server <address1> <address2>

Use the **dns-server** command to specify the DNS server address(es) to assign to a client.

Syntax Description

<address1>	Assigns the first DNS server address.
<address2>	Optional. Assigns the second DNS server address.

Default Values

By default, no DNS server address is defined.

Usage Examples

The following example defines two DNS server addresses for this configuration pool:

```
ProCurve(config-ike-client-pool)#dns-server 172.1.17.1 172.1.17.3
```


ip-range *<start ip> <end ip>*

Use the **ip-range** command to specify the range of addresses from which the router draws when assigning an IP address to a client.

Syntax Description

<i><start ip></i>	Specifies the first IP address in the range for this pool.
<i><end ip></i>	Specifies the last IP address in the range for this pool.

Default Values

By default, no IP address range is defined.

Usage Examples

The following example defines an IP address range for this configuration pool:

ProCurve(config-ike-client-pool)#**ip-range 172.1.1.1 172.1.1.25**

netbios-name-server *<address1> <address2>*

Use the **netbios-name-server** command to specify the NetBIOS Windows Internet Naming Service (WINS) name servers to assign to a client.

Syntax Description

<i><address1></i>	Specifies the first WINS server address to assign.
<i><address2></i>	Specifies the second WINS server address to assign.

Default Values

By default, no WINS server address is defined.

Usage Examples

The following example defines two WINS server addresses for this configuration pool:

```
ProCurve(config-ike-client-pool)#netbios-name-server 172.1.17.1 172.1.17.25
```

CRYPTO MAP IKE COMMAND SET

To activate the Crypto Map IKE mode, enter a valid version of the **crypto map ipsec-ike** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#crypto map Map-Name 10 ipsec-ike  
ProCurve(config-crypto-map)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

antireplay [on page 484](#)
ike-policy <policy number> [on page 485](#)
match address <listname> [on page 486](#)
set peer <address> [on page 488](#)
set pfs [group1 | group2] [on page 489](#)
set security-association lifetime [kilobytes | seconds] <value> [on page 490](#)
set transform-set <setname1 - setname6> [on page 491](#)

antireplay

Use the **antireplay** command to enable antireplay sequence number checking for all security associations created on this crypto map. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Usage Examples

The following example enables antireplay sequence checking on crypto map VPN 100:

```
ProCurve(config)#crypto map VPN 100 ipsec-ike  
ProCurve(config-crypto-map)#antireplay
```

ike-policy <policy number>

Use the **ike-policy** command to ensure that only a specified IKE policy is used to establish the IPSec tunnel. This prevents any mobile VPN policies from using IPSec policies that are configured for static VPN peer policies.

Syntax Description

<policy number>	Specifies the policy number of the policy to assign to this crypto map.

Default Values

No defaults necessary for this command.

Usage Examples

The following example shows a typical crypto map configuration:

```
ProCurve(config)#crypto ike policy 100
ProCurve(config)#crypto map VPN 10 ipsec-ike
ProCurve(config-crypto-map)#description "Remote Office"
ProCurve(config-crypto-map)#match address VPN-10-vpn-selectors
ProCurve(config-crypto-map)#set peer 10.22.17.13
ProCurve(config-crypto-map)#set transform-set esp-3des-esp-md5-hmac
ProCurve(config-crypto-map)#ike-policy 100
```

match address <listname>

Use the **match address** command to assign an IP access list to a crypto map definition. The access list designates the IP packets to be encrypted by this crypto map. See *ip access-list extended <listname>* on page 285 for more information on creating access lists.

Syntax Description

<listname>	Enter the name of the access list you wish to assign to this crypto map.

Default Values

By default, no IP access lists are defined.

Functional Notes

Crypto map entries do not directly contain the selectors used to determine which data to secure. Instead, the crypto map entry refers to an access control list. An access control list (ACL) is assigned to the crypto map using the **match address** command. If no ACL is configured for a crypto map, then the entry is incomplete and will have no effect on the system.

The entries of the ACL used in a crypto map should be created with respect to traffic sent by the product. The source information must be the local product and the destination must be the peer.

Only extended access lists can be used in crypto maps.

Usage Examples

The following example shows setting up an ACL (called **NewList**) and then assigning the new list to a crypto map (called **NewMap**):

```
ProCurve(config)#ip access-list extended NewList
```

Configuring New Extended ACL "NewList"

```
ProCurve(config-ext-nacl)#exit
```

```
ProCurve(config)#crypto map NewMap 10 ipsec-ike
```

```
ProCurve(config-crypto-map)#match address NewList
```

Technology Review

A crypto map entry is a single policy that describes how certain traffic is to be secured. There are two types of crypto map entries: **ipsec-manual** and **ipsec-ike**. Each entry is given an index, which is used to sort the ordered list.

When an unsecure packet arrives on an interface, the crypto map set associated with that interface is processed in order. If a crypto map entry matches the unsecure traffic, the traffic is discarded.

When a packet is to be transmitted on an interface, the crypto map set associated with that interface is processed in order. The first crypto map entry that matches the packet will be used to secure the packet. If a suitable SA exists, that is used for transmission. Otherwise, IKE is used to establish an SA with the peer. If no SA exists, and the crypto map entry is "respond only," the packet is discarded.

When a secured packet arrives on an interface, its SPI is used to look up an SA. If an SA does not exist, or if the packet fails any of the security checks (bad authentication, traffic does not match SA selectors, etc.), it is discarded. If all checks pass, the packet is forwarded normally.

set peer <address>

Use the **set peer** command to set the IP address of the peer device. This must be set for multiple remote peers.

Syntax Description

<address>	Specifies the IP address of the peer device. If this is not configured, it implies responder only to any peer.

Default Values

There are no default settings for this command.

Functional Notes

If no peer IP addresses are configured, the entry will only be used to respond to IPSec requests; it cannot initiate the requests (since it doesn't know which IP address to send the packet to). If a single peer IP address is configured, the crypto map entry can be used to both initiate and respond to SAs.

The peer IP address is the public IP address of the device which will terminate the IPSec tunnel. If the peer IP address is not static, the product cannot initiate the VPN tunnel. By setting no peer IP address, the product can respond to an IPSec tunnel request.

Usage Examples

The following example sets the peer IP address of 10.100.23.64:

```
ProCurve(config-crypto-map)#set peer 10.100.23.64
```


set pfs [group1 | group2]

Use the **set pfs** command to choose the type of perfect forward secrecy (if any) that will be required during IPsec negotiation of security associations for this crypto map. Use the **no** form of this command to require no PFS.

Syntax Description

group1	Requires IPsec to use Diffie-Hellman Group 1 (768-bit modulus) exchange during IPsec SA key generation.
group2	Requires IPsec to use Diffie-Hellman Group 2 (1024-bit modulus) exchange during IPsec SA key generation.

Default Values

By default, no PFS will be used during IPsec SA key generation.

Functional Notes

If left at the default setting, no perfect forward secrecy (PFS) will be used during IPsec SA key generation. If PFS is specified, then the specified Diffie-Hellman Group exchange will be used for the initial and all subsequent key generation, thus providing no data linkage between prior keys and future keys.

Usage Examples

The following example specifies use of the Diffie-Hellman Group 1 exchange during IPsec SA key generation:

```
ProCurve(config-crypto-map)#set pfs group 1
```

set security-association lifetime [kilobytes | seconds] <value>

Use the **set security-association lifetime** command to define the lifetime (in kilobytes and/or seconds) of the IPSec SAs created by this crypto map.

Syntax Description

kilobytes <value>	Specifies the SA lifetime limit in kilobytes.
seconds <value>	Specifies the SA lifetime limit in seconds.

Default Values

By default, the security-association lifetime is set to 28,800 seconds and there is no default for the kilobytes lifetime.

Functional Notes

Values can be entered for this command in both kilobytes and seconds. Whichever limit is reached first will end the security association.

Usage Examples

The following example sets the SA lifetime to 300 kilobytes and 2 hours:

```
ProCurve(config-crypto-map)#set security-association lifetime kilobytes 300
```

```
ProCurve(config-crypto-map)#set security-association lifetime seconds 7200
```

set transform-set <setname1 - setname6>

Use the **set transform-set** command to assign up to six transform sets to a crypto map.

Syntax Description

<setname>	Assign up to six transform-sets to this crypto map by listing the set names, separated by a space.

Default Values

By default, there is no transform set assigned to the crypto map.

Functional Notes

Crypto map entries do not directly contain the transform configuration for securing data. Instead, the crypto map is associated with transform sets which contain specific security algorithms.

If no transform set is configured for a crypto map, then the entry is incomplete and will have no effect on the system.

Usage Examples

The following example first creates a transform set (**Set1**) consisting of two security algorithms (up to three may be defined), and then assigns the transform set to a crypto map (**Map1**):

```
ProCurve(config)#crypto ipsec transform-set Set1 esp-3des esp-sha-hmac
ProCurve(cfg-crypto-trans)#exit
ProCurve(config)#crypto map Map1 1 ipsec-ike
ProCurve(config-crypto-map)#set transform-set Set1
```

CRYPTO MAP MANUAL COMMAND SET

To activate the Crypto Map Manual mode, enter a valid version of the **crypto map ipsec-manual** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#crypto map Map-Name 10 ipsec-manual
ProCurve(config-crypto-map)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

antireplay [on page 493](#)
ike-policy <policy number> [on page 494](#)
match address <listname> [on page 495](#)
set peer <address> [on page 497](#)
set session-key [inbound | outbound] [on page 498](#)
set transform-set <setname> [on page 502](#)

antireplay

Use the **antireplay** command to enable antireplay sequence number checking for all security associations created on this crypto map. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Usage Examples

The following example enables antireplay sequence checking on crypto map VPN 100:

```
ProCurve(config)#crypto map VPN 100 ipsec-manual  
ProCurve(config-crypto-map)#antireplay
```

ike-policy <policy number>

Use the **ike-policy** command to ensure that only a specified IKE policy is used to establish the IPSec tunnel. This prevents any mobile VPN policies from using IPSec policies that are configured for static VPN peer policies.

Syntax Description

<policy number>	Specifies the policy number of the policy to assign to this crypto map.

Default Values

No defaults necessary for this command.

Usage Examples

The following example shows a typical crypto map configuration:

```
ProCurve(config)#crypto ike policy 100
ProCurve(config)#crypto map VPN 10 ipsec-manual
ProCurve(config-crypto-map)#description "Remote Office"
ProCurve(config-crypto-map)#match address VPN-10-vpn-selectors
ProCurve(config-crypto-map)#set peer 10.22.17.13
ProCurve(config-crypto-map)#set transform-set esp-3des-esp-md5-hmac
ProCurve(config-crypto-map)#ike-policy 100
```

match address <listname>

Use the **match address** command to assign an IP access list to a crypto map definition. The access list designates the IP packets to be encrypted by this crypto map. See *ip access-list extended <listname>* on page 285 for more information on creating access lists.

Syntax Description

<listname>	Specifies the name of the access list you wish to assign to this crypto map.

Default Values

By default, no IP access lists are defined.

Functional Notes

Crypto map entries do not directly contain the selectors used to determine which data to secure. Instead, the crypto map entry refers to an access control list. An access control list (ACL) is assigned to the crypto map using the **match address** command (see *crypto map* on page 267). If no ACL is configured for a crypto map, then the entry is incomplete and will have no effect on the system.

The entries of the ACL used in a crypto map should be created with respect to traffic sent by the product. The source information must be the local product, and the destination must be the peer.

Only extended access lists can be used in crypto maps.

Usage Examples

The following example shows setting up an access list (called **NewList**) and then assigning the new list to a crypto map (called **NewMap**):

```
ProCurve(config)#ip access-list extended NewList
```

Configuring New Extended ACL "NewList"

```
ProCurve(config-ext-nacl)#exit
```

```
ProCurve(config)#crypto map NewMap 10 ipsec-manual
```

```
ProCurve(config-crypto-map)#match address NewList
```

Technology Review

A crypto map entry is a single policy that describes how certain traffic is to be secured. There are two types of crypto map entries: **ipsec-manual** and **ipsec-ike**. Each entry is given an index, which is used to sort the ordered list.

When an unsecured packet arrives on an interface, the crypto map set associated with that interface is processed in order. If a crypto map entry matches the unsecured traffic, the traffic is discarded.

When a packet is to be transmitted on an interface, the crypto map set associated with that interface is processed in order. The first crypto map entry that matches the packet will be used to secure the packet. If a suitable SA exists, that is used for transmission. Otherwise, IKE is used to establish an SA with the peer. If no SA exists, and the crypto map entry is "respond only," the packet is discarded.

When a secured packet arrives on an interface, its SPI is used to look up an SA. If an SA does not exist, or if the packet fails any of the security checks (bad authentication, traffic does not match SA selectors, etc.), it is discarded. If all checks pass, the packet is forwarded normally.

set peer <address>

Use the **set peer** command to set the IP address of the peer device.

Syntax Description

<address>	Specifies the IP address of the peer device.

Default Values

There are no default settings for this command.

Functional Notes

If no peer IP address is configured, the manual crypto map is not valid and not complete. A peer IP address is required for manual crypto maps. To change the peer IP address, the **no set peer** command must be issued first; then the new peer IP address can be configured.

Usage Examples

The following example sets the peer IP address of 10.100.23.64:

```
ProCurve(config-crypto-map)#set peer 10.100.23.64
```

set session-key [inbound | outbound]

Use the **set session-key** command to define the encryption and authentication keys for this crypto map.

Variations of this command include the following:

```
set session-key inbound ah <SPI> <keyvalue>
set session-key inbound esp <SPI> authenticator <keyvalue>
set session-key inbound esp <SPI> cipher <keyvalue>
set session-key inbound esp <SPI> cipher <keyvalue> authenticator <keyvalue>
set session-key outbound ah <SPI> <keyvalue>
set session-key outbound esp <SPI> authenticator <keyvalue>
set session-key outbound esp <SPI> cipher <keyvalue>
set session-key outbound esp <SPI> cipher <keyvalue> authenticator <keyvalue>
```

Syntax Description

inbound	Defines encryption keys for inbound traffic.
outbound	Defines encryption keys for outbound traffic.
ah <SPI>	Specifies authentication header protocol.
esp <SPI>	Specifies encapsulating security payload protocol.
cipher <keyvalue>	Specifies encryption/decryption key.
authenticator <keyvalue>	Specifies authentication key.

Default Values

There are no default settings for this command.

Functional Notes

The inbound local security parameter index (SPI) must equal the outbound remote SPI. The outbound local SPI must equal the inbound remote SPI. The key values are the hexadecimal representations of the keys. They are not true ASCII strings. Therefore, a key of 3031323334353637 represents "01234567".

See the following table for key length requirements.

Algorithm	Minimum key length required
DES	64-bits in length; 8 hexadecimal bytes
3DES	192-bits in length; 24 hexadecimal bytes
AES-128-CBC	128-bits in length; 16 hexadecimal bytes
AES-192-CBC	192-bits in length; 24 hexadecimal bytes
AES-256-CBC	256-bits in length; 32 hexadecimal bytes

Functional Notes (Continued)

MD5	128-bits in length; 16 hexadecimal bytes
SHA1	160-bits in length; 20 hexadecimal bytes

Technology Review

The following example configures an Secure Router OS product for VPN using IPSec manual keys. This example assumes that the Secure Router OS product has been configured with a WAN IP Address of 63.97.45.57 on interface **ppp 1** and a LAN IP Address of 10.10.10.254 on interface **ethernet 0/1**. The Peer Private IP Subnet is 10.10.20.0.

For more detailed information on VPN configuration, refer to the *VPN Configuration Guide* located on the *ProCurve SROS Documentation CD* provided with your unit.

Step 1:

Enter the Global Configuration mode (i.e., config terminal mode).

```
ProCurve>enable
```

```
ProCurve#configure terminal
```

Step 2:

Enable VPN support using the **ip crypto** command. This command allows crypto maps to be applied to interfaces, and enables the IKE server to listen for IKE negotiation sessions on UDP port 500.

```
ProCurve(config)#ip crypto
```

Step 3:

Define the transform set. A transform set defines the encryption and/or authentication algorithms to be used to secure the data transmitted over the VPN tunnel. Multiple transform sets may be defined in a system. Once a transform set is defined, many different crypto maps within the system can reference it. In this example, a transform set named **highly_secure** has been created. This transform set defines ESP with authentication implemented using 3DES encryption and SHA1 authentication.

```
ProCurve(config)#crypto ipsec transform-set highly_secure esp-3des esp-sha-hmac
```

```
ProCurve(cfg-crypto-trans)#mode tunnel
```

Step 4:

Define an IP access list. An Extended Access Control List is used to specify which traffic needs to be sent securely over the VPN tunnel. The entries in the list are defined with respect to the local system. The source IP address will be the source of the traffic to be encrypted. The destination IP address will be the receiver of the data on the other side of the VPN tunnel.

```
ProCurve(config)#ip access-list extended corporate_traffic
```

```
ProCurve(config-ext-nacl)#permit ip 10.10.10.0 0.0.0.255 10.10.20.0 0.0.0.255 log  
deny ip any any
```

Step 5:

Create crypto map and define manual keys. A Crypto Map is used to define a set of encryption schemes to be used for a given interface. A crypto map entry has a unique index within the crypto map set. The crypto map entry will specify whether IKE is used to generate encryption keys or if manually specified keys will be used. The crypto map entry will also specify who will be terminating the VPN tunnel, as well as which transform set or sets will be used to encrypt and/or authenticate the traffic on that VPN tunnel. It also specifies the lifetime of all created IPsec security associations.

The keys for the algorithms defined in the transform set associated with the crypto map will be defined by using the **set session-key** command. A separate key is needed for both inbound and outbound traffic. The key format consists of a string of hexadecimal values without the leading **0x** for each character. For example, a cipher key of **this is my cipher key** would be entered as:

74686973206973206D7920636970686572206B6579.

A unique Security Parameter Index (SPI) is needed for both inbound and outbound traffic. The local system's inbound SPI and keys will be the peer's outbound SPI and keys. The local system's outbound SPI and keys will be the peer's inbound SPI and keys. In this example the following keys and SPIs are used:

- Inbound cipher SPI: 300Inbound cipher key: "2te\$#g89jnr(j!@4rvnfhg5e"
- Outbound cipher SPI: 400Outbound cipher key: "8564hgjelrign*&(gnb#1\$d3"
- Inbound authenticator key:"r5%^ughembkdjhj34\$x.<"
- Outbound authenticator key:"io78*7gner#4(mgnstd!3"
-

ProCurve(config)#**crypto map corporate_vpn 1 ipsec-ike**

ProCurve(config-crypto-map)#**match address corporate_traffic**

ProCurve(config-crypto-map)#**set peer 63.105.15.129**

ProCurve(config-crypto-map)#**set transform-set highly_secure**

ProCurve(config-crypto-map)#**set session-key inbound esp 300 cipher**
32746524236738396A6E72286A21403472766E6668673565 authenticator
7235255E756768656D626B64686A333424782E3C

ProCurve(config-crypto-map)#**set session-key outbound esp 400 cipher**
3835363468676A656C7269676E2A2628676E622331246433 authenticator
696F37382A37676E65722334286D676E73642133

Step 6:

Configure public interface. This process includes configuring the IP address for the interface and applying the appropriate crypto map to the interface. Crypto maps are applied to the interface on which encrypted traffic will be transmitted.

ProCurve(config)#**interface ppp 1**

ProCurve(config-ppp 1)#**ip address 63.97.45.57 255.255.255.248**

ProCurve(config-ppp 1)#**crypto map corporate_vpn**

ProCurve(config-ppp 1)#**no shutdown**

Step 7:

Configure private interface to allow all traffic destined for the VPN tunnel to be routed to the appropriate gateway.

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#ip address 10.10.10.254 255.255.255.0
```

```
ProCurve(config-eth 0/1)#no shutdown
```

```
ProCurve(config-eth 0/1)#exit
```

set transform-set <setname>

Use the **set transform-set** command to assign a transform set to a crypto map.

Syntax Description

<setname>	Assigns a transform set to this crypto map by entering the set name.

Default Values

By default, no transform set is assigned to the crypto map.

Functional Notes

Crypto map entries do not directly contain the transform configuration for securing data. Instead, the crypto map is associated with transform sets which contain specific security algorithms.

If no transform set is configured for a crypto map, then the entry is incomplete and will have no effect on the system. For manual key crypto maps, only one transform set can be specified.

Usage Examples

The following example first creates a transform set (**Set1**) consisting of two security algorithms (up to three may be defined), and then assigns the transform set to a crypto map (**Map1**):

```
ProCurve(config)#crypto ipsec transform-set Set1 esp-3des esp-sha-hmac
ProCurve(cfg-crypto-trans)#exit
```

```
ProCurve(config)#crypto map Map1 1 ipsec-manual
ProCurve(config-crypto-map)#set transform-set Set1
```

RADIUS GROUP COMMAND SET

To activate the Radius Group mode, enter the **aaa group server** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#aaa group server radius myServer  
ProCurve(config-sg-radius)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

server [*acct-port* <port number>| *auth-port* <port number>] [on page 504](#)

server [acct-port <port number>| auth-port <port number>]

Use the **server** command to add a predefined RADIUS server to the current named list of servers. See *radius-server* on page 364 for more information.

Syntax Description

acct-port <port number>	Defines the accounting port value.
auth-port <port number>	Defines the authorization port value.

Default Values

No defaults necessary for this command.

Usage Examples

The following example adds a server to the **myServers** list:

```
ProCurve(config)#aaa group server radius myServers
ProCurve(config-sg-radius)#server 1.2.3.4 acct-port 786 auth-port 1812
ProCurve(config-sg-radius)#server 4.3.2.1
ProCurve(config-sg-radius)#exit
ProCurve(config)#
```

or

```
ProCurve(config)#aaa group server radius myServers
ProCurve(config-sg-radius)#server 4.3.2.1
ProCurve(config-sg-radius)#exit
ProCurve(config)#
```

CA PROFILE CONFIGURATION COMMAND SET

To activate the Certificate Authority (CA) Profile Configuration mode, enter the **crypto ca profile** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#crypto ca profile MyProfile
Configuring New CA Profile MyProfile
ProCurve(ca-profile)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

crl optional [on page 506](#)
email address <email address> [on page 507](#)
enrollment retry [count | period] [on page 508](#)
enrollment terminal [on page 509](#)
enrollment url <url> [on page 510](#)
fqdn <fqdn> [on page 511](#)
ip-address <address> [on page 512](#)
password <password> [on page 513](#)
serial-number [on page 514](#)
subject-name <name> [on page 515](#)

crl optional

Use the **crl optional** command to make CRL verification optional.

Syntax Description

No subcommands.

Default Values

By default, CRL optional is enabled.

Functional Notes

If enabled, the Secure Router OS is able to accept certificates even if no CRL is loaded into the configuration. Currently, this is the only mode supported by the Secure Router OS for CRL negotiations.

Usage Examples

The following example sets CRL verification as optional:

```
ProCurve(ca-profile)#crl optional
```

email address *<email address>*

Use the **email address** command to specify that an email address should be included in the certificate request.

Syntax Description

<i><email address></i>	Specifies the complete email address to use when sending certificate requests. This field allows up to 51 characters.

Default Values

No defaults necessary for this command.

Functional Notes

Configuring this setting simplifies the **crypto ca enroll** dialog, allowing you to enter the email address only once rather than every time you go through the enrollment process. See *crypto ca enroll <name>* on page 256.

Usage Examples

The following example specifies **jdoo@company.com** as the email address to be sent in certificate requests:

```
ProCurve(ca-profile)#email address jdoo@company.com
```

enrollment retry [count | period]

Use the **enrollment retry** command to determine how the Secure Router OS handles certificate requests.

Syntax Description

count <count>	Specifies the number of times the Secure Router OS re-sends a certificate request when it does not receive a response from the previous request. Range: 1 to 100.
period <minutes>	Specifies the time period between certificate request retries. The default is 1 minute between retries. Range: 1 to 60 minutes.

Default Values

By default, period is set to 5 minutes, and count is set to 12 retries.

Usage Examples

The following example configures the Secure Router OS to send certificate requests every two minutes, stopping after 50 retries (if no response is received):

```
ProCurve(ca-profile)#enrollment retry count 50
```

```
ProCurve(ca-profile)#enrollment retry period 2
```

enrollment terminal

Use the **enrollment terminal** command to specify manual (i.e., cut-and-paste) certificate enrollment.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Functional Notes

This mode is overridden if the **enrollment url** command specifies the CA to which automatic certificate requests are to be sent via simple certificate exchange protocol (SCEP). Issuing an **enrollment terminal** command after using the **enrollment url** command deletes the URL and forces the unit to use manual enrollment. See *enrollment url <url>* on page 510 for more information.

Usage Examples

The following example configures the Secure Router OS to accept manual certificate enrollment input:

```
ProCurve(ca-profile)#enrollment terminal
```

enrollment url <url>

Use the **enrollment url** command to specify the URL of the CA to which the Secure Router OS should send certificate requests.

Syntax Description

<url>	Enter the certificate authority's URL (e.g., <code>http://10.10.10.1:400/abcdefg/pkiclient.exe</code>).

Default Values

No defaults necessary for this command.

Functional Notes

When entering the URL **http://** is required, followed by the IP address or DNS name of the CA. If the port number is something other than 80, include it after the IP address or DNS name separated with a colon (:).

The CA may have other necessary information to include in the CGI path before ending with the actual CGI program. An example template to follow is **http://hostname:port/path/to/program.exe**.

NOTE: To use the default program **pkiclient.exe** without specifying it, end the URL with a slash (/). Otherwise, you must enter the program name to use. For example, **http://10.10.10.1:400/abcdefg/** will assume **pkiclient.exe** as the program (but not including the terminating slash is a configuration error).

Specifying this command will override the **enrollment terminal** setting as described previously (see *enrollment terminal* on page 509).

Usage Examples

The following example specifies **http://CAserver/certsrv/mscep/mscep.dll** as the URL to which the Secure Router OS will send certificate requests:

```
ProCurve(ca-profile)#enrollment url http://CAserver/certsrv/mscep/mscep.dll
```

fqdn <fqdn>

Use the **fqdn** command to specify a fully-qualified domain name (FQDN) to be included in the certificate requests.

Syntax Description

<fqdn>	Specifies the FQDN (e.g., company.com) to be included in requests.

Default Values

No defaults necessary for this command.

Functional Notes

Configuring this setting simplifies the **crypto ca enroll** dialog, allowing you to enter the FQDN only once rather than every time you go through the enrollment process. See *crypto ca enroll* <name> on page 256.

Usage Examples

The following example specifies **company.com** as the FQDN to be sent in certificate requests:

```
ProCurve(ca-profile)#fqdn company.com
```

ip-address <address>

Use the **ip-address** command to specify an IP address to be included in the certificate requests.

Syntax Description

<address>	Defines the IP address in dotted decimal notation (e.g., 192.22.73.101).

Default Values

No defaults necessary for this command.

Functional Notes

Configuring this setting simplifies the **crypto ca enroll** dialog, allowing you to enter the IP address only once rather than every time you go through the enrollment process. Refer to *crypto ca enroll <name>* on page 256.

Usage Examples

The following example specifies **66.203.52.193** as the IP address to be sent in certificate requests:

```
ProCurve(ca-profile)#ip-address 66.203.52.193
```


password <password>

Use the **password** command to specify the challenge password for simple certificate exchange protocol (SCEP). Use the **no** form of this command to allow CA requests to be sent automatically (using SCEP) without requiring a password.

Syntax Description

<password>	Specifies the SCEP password (up to 80 characters).

Default Values

By default, no password is required.

Functional Notes

There are two places for configuring a SCEP password:

- At the **(ca-profile)#** prompt.
- If it is not configured at the **(ca-profile)#** prompt, you are prompted to enter one when going through the certificate enrollment process.

The password is sent to the CA from which you are requesting a certificate. The CA may then ask for the password later before a certificate can be revoked. See *crypto ca enroll <name>* on page 256.

Usage Examples

The following example sets the SCEP challenge password to **procurve**:

```
ProCurve(ca-profile)#password procurve
```

serial-number

Use the **serial-number** command to specify that a serial number will be included in the certificate request.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

By default, this command is set to **no serial-number**, which means that the serial number is not included in the certificate requests.

Usage Examples

The following example configures Secure Router OS to include a serial number in the certificate request:

```
ProCurve(ca-profile)#serial-number
```

subject-name <name>

Use the **subject-name** command to specify the subject name used in the certificate request.

Syntax Description

<name>	Specifies a subject name string with up to 256 characters entered in X.500 LDAP format.

Default Values

By default, there is no subject name configured.

Functional Notes

Configuring this setting simplifies the **crypto ca enroll** dialog, allowing you to enter the subject name only once rather than every time you go through the enrollment process. Refer to *crypto ca enroll <name>* on page 256.

Usage Examples

The following example assigns a subject name of **cert** to certificate requests:

```
ProCurve(ca-profile)#subject-name cert
```

CERTIFICATE CONFIGURATION COMMAND SET

To activate the Certificate Configuration mode, enter the **crypto ca certificate chain** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#crypto ca certificate chain MyProfile  
ProCurve(config-cert-chain)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

certificate *<serial-number>* [on page 517](#)

certificate ca *<serial-number>* [on page 518](#)

crl [on page 519](#)

certificate <*serial-number*>

Use the **certificate** command to restore a certificate. Use the **no** form of this command to remove a specific certificate from the certificate chain.

Syntax Description

< <i>serial-number</i> >	Specifies the certificate's serial number (up to 51 characters). This value can be found for existing certificates by using the show run command.

Default Values

No defaults necessary for this command.

Functional Notes

The user typically does not enter this command. It is primarily used to restore certificates from the startup configuration when the product is powered up.

Usage Examples

The following example removes the certificate with the serial number 73f0bfe5ed8391a54d1214390a36cee7:

```
ProCurve(config-cert-chain)#no certificate 73f0bfe5ed8391a54d1214390a36cee7
```

certificate ca <serial-number>

Use the **certificate ca** command to restore a CA certificate. Use the **no** form of this command to remove a specific certificate from the certificate chain for a CA.

Syntax Description

<i><serial-number></i>	Specifies the certificate's serial number (up to 51 characters). This value can be found for existing certificates by using the show run command.

Default Values

No defaults necessary for this command.

Functional Notes

The user typically does not enter this command. It is primarily used to restore certificates from the startup configuration when the product is powered up.

Usage Examples

The following example removes the CA certificate with the serial number 0712:

```
ProCurve(config-cert-chain)#no certificate ca 0712
```

crl

Use the **crl** command to restore a CRL. Use the **no** form of this command to remove the CRL for the specific CA.

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Functional Notes

The user typically does not enter this command. It is primarily used to restore CRLs from the startup configuration when the product is powered up.

Usage Examples

The following example removes the CRL for the current CA:

```
ProCurve(config-cert-chain)#no crl
```

ETHERNET INTERFACE CONFIGURATION COMMAND SET

There are several types of Ethernet interfaces associated with the Secure Router OS:

- Basic Ethernet interfaces (e.g., eth 0/1)
- Ethernet sub-interfaces associated with a VLAN (e.g., eth 0/1.1)
- Ethernet interface range (e.g., eth 0/1, 0/2)

To activate the basic Ethernet Interface Configuration mode, enter the **interface ethernet** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#
```

To activate the Ethernet Sub-Interface Configuration mode, enter the **interface ethernet** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#interface ethernet 0/1.1  
ProCurve(config-eth 0/1.1)#
```

To activate the Ethernet Configuration mode for a range of interfaces, enter the **interface range** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#interface range ethernet 0/1, 0/2  
ProCurve(config-eth 0/1, 0/2)#
```

Note *Not all Ethernet commands apply to all Ethernet types. Use the ? command to display a list of valid commands. For example:*

```
ProCurve>enable
Password: xxxxx
ProCurve#config term
ProCurve(config)#int eth 0/1
ProCurve(config-eth 0/1)#?
```

access-policy - Assign access control policy for this interface
alias - A text name assigned by an SNMP NMS
arp - Set ARP commands
bandwidth - Set bandwidth informational parameter
bridge-group - Assign the current interface to a bridge group
etc....

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)
bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)
shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycyname> [on page 523](#)
arp arpa [on page 526](#)
bandwidth <value> [on page 527](#)
bridge-group <group#> [on page 528](#)
crypto map <mapname> [on page 529](#)
dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password> [on page 531](#)
encapsulation 802.1q [on page 533](#)
full-duplex [on page 534](#)
half-duplex [on page 535](#)

ip commands [begin on page 536](#)

lldp receive [on page 555](#)

lldp send [management-address l port-description l system-capabilities l system-description l system-name l and-receive] [on page 556](#)

lldp send-and-receive [on page 557](#)

mac-address <address> [on page 558](#)

mtu <size> [on page 559](#)

port-auth supplicant enable [username <username> | password <password>] [on page 560](#)

qos-policy out <mapname> [on page 561](#)

snmp trap [on page 562](#)

snmp trap link-status [on page 563](#)

spanning-tree commands [begin on page 564](#)

speed [10 | 100 | auto] [on page 569](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic to an interface. Use the **no** form of this command to remove an access policy association.

Note *Configured access policies will only be active if the **ip firewall** command has been entered at the Global Configuration Mode prompt to enable the SROS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.*

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor (all access policy descriptors are case-sensitive).

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**.

Usage Examples

The following example associates the access policy UnTrusted (to allow inbound traffic to the Web server) to the Ethernet 0/1 interface:

Enable the Secure Router OS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list InWeb:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access policy with the Ethernet 0/1 interface:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a “range”. Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a “don’t care”. For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an IP policy class that uses a configured access list. Secure Router OS access policies are used to permit, deny, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

nat source list <access list names> **address** <IP address> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list <access list names> interface <interface> overload

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list <access list names> address <IP address>

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**. The following example assigns access policy **MatchAll** to the Ethernet 0/1 interface:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#access-policy MatchAll
```

arp arpa

Use the **arp arpa** command to enable address resolution protocol (ARP) on the Ethernet interface.

Syntax Description

arpa	Sets standard address resolution protocol for this interface.

Default Values

The default for this command is arpa.

Usage Examples

The following example enables standard ARP for the Ethernet interface:

```
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#arp arpa
```

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Specifies bandwidth in kbps.

Default Values

To view default values, use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the Ethernet 0/1 interface to 10 Mbps:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#bandwidth 10000
```

bridge-group <group#>

Use the **bridge-group** command to assign an interface to the specified bridge group. Use the **no** form of this command to remove the interface from the bridge group.

Syntax Description

<group#>	Specifies the bridge group number (1 to 255).

Default Values

By default, there are no configured bridge groups.

Functional Notes

A bridged network can provide excellent traffic management to reduce collisions and limit the amount of bandwidth wasted with unnecessary transmissions when routing is not necessary. Any two interfaces can be bridged (e.g., Ethernet to T1 bridge, Ethernet to Frame Relay sub-interface).

Usage Examples

The following example assigns the Ethernet interface to bridge-group 17:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#bridge-group 17
```


crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<mapname> Specifies the crypto map name that you wish to assign to the interface.

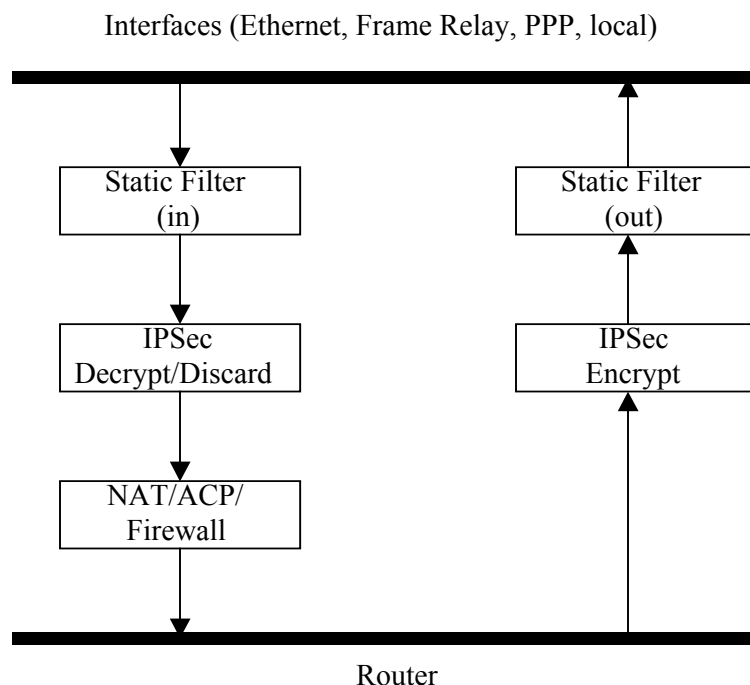
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following notes in mind.

When defining the policy-class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the unencrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical Secure Router OS data-flow logic.



As shown in the diagram above, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access-group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only unencrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy-class on an interface. When specifying the ACLs for a crypto map, the source information is the private local-side, unencrypted source of the data. The destination information will be the far-end, unencrypted destination of the data. However, ACLs for a policy-class work in reverse. The source information for the ACL in a policy-class is the far-end. The destination information is the local-side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the Ethernet interface:

```
ProCurve(config-eth 0/1)#crypto map MyMap
```

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password>

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

See *Functional Notes* below for argument descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A web-based interface provides two levels of control over your domain, catering to average or power users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs, and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

When to Use Dynamic DNS

If your IP address does not change often (or at all) but you still want an easy name to remember it by (without having to purchase your own domain name), Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com), you need Custom DNS service, which also provides full dynamic and static IP address support.

Dynamic DNS service can be extremely helpful for site-to-site VPN connections using dynamic IP addressing. Instead of creating a tunnel to the IP-address, a tunnel to the DNS-name is established.

Usage Examples

The following example sets the dynamic-dns to **dyndns-custom** with hostname **host**, username **user**, and password **pass**:

```
ProCurve(config-eth 0/1)#dynamic-dns dyndns-custom host user pass
```

encapsulation 802.1q

Use the **encapsulation 802.1q** command to put the interface into 802.1q (VLAN) mode.

Syntax Description

No subcommands.

Default Values

No default value is necessary for this command.

Functional Notes

When operating on a circuit that is providing timing, setting the **clock source** to **line** can avoid errors such as Clock Slip Seconds (CSS).

Usage Examples

The following example puts interface **eth 0/1** in 802.1q mode and configures a sub-interface for vlan usage:

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#encapsulation 802.1q
ProCurve(config-eth 0/1)#interface ethernet 0/1.1
ProCurve(config-eth 0/1.1)#vlan-id 3
```

full-duplex

Use the **full-duplex** command to configure the Ethernet interface for full-duplex operation. This allows the interface to send and receive simultaneously. Use the **no** form of this command to return to the default **half-duplex** operation.

Syntax Description

No subcommands.

Default Values

By default, all Ethernet interfaces are configured for half-duplex operation.

Functional Notes

Full-duplex Ethernet is a variety of Ethernet technology currently being standardized by the IEEE. Because there is no official standard, vendors are free to implement their independent versions of full-duplex operation. Therefore, it is not safe to assume that one vendor's equipment will work with another.

Devices at each end of a full-duplex link have the ability to send and receive data simultaneously over the link. Theoretically, this simultaneous action can provide twice the bandwidth of normal (half-duplex) Ethernet. To deploy full-duplex Ethernet, each end of the link must only connect to a single device (a workstation or a switched hub port). With only two devices on a full-duplex link, there is no need to use the medium access control mechanism (to share the signal channel with multiple stations) and listen for other transmissions or collisions before sending data.

Note	<i>Some Ethernet equipment (though rare) is unable to negotiate duplex if speed is manually determined. To avoid incompatibilities, manually set the duplex if the speed is manually set. See speed [10 100 auto] on page 569 for more information.</i>

The 10BaseT, 100BaseTX, and 100BaseFX signalling systems support full-duplex operation (because they have transmit and receive signal paths that can be simultaneously active).

Usage Examples

The following example configures the Ethernet interface for **full-duplex** operation:

```
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#full-duplex
```

half-duplex

Use the **half-duplex** command to configure the Ethernet interface for half-duplex operation. This setting allows the Ethernet interface to either send or receive at any given moment, but not simultaneously. Use the **no** form of this command to disable half-duplex operation.

Syntax Description

No subcommands.

Default Values

By default, all Ethernet interfaces are configured for half-duplex operation.

Functional Notes

Half-duplex Ethernet is the traditional form of Ethernet that employs the Carrier Sense Multiple Access/Collision Detect (CSMA/CD) protocol to allow two or more hosts to share a common transmission medium while providing mechanisms to avoid collisions. A host on a half-duplex link must “listen” on the link and only transmit when there is an idle period. Packets transmitted on the link are broadcast (so it will be “heard” by all hosts on the network). In the event of a collision (two hosts transmitting at once), a message is sent to inform all hosts of the collision and a backoff algorithm is implemented. The backoff algorithm requires the station to remain silent for a random period of time before attempting another transmission. This sequence is repeated until a successful data transmission occurs.

Note	<i>Some Ethernet equipment (though rare) is unable to negotiate duplex if speed is manually determined. To avoid incompatibilities, manually set the duplex if the speed is manually set. See speed [10 100 auto] on page 569 for more information.</i>

Usage Examples

The following example configures the Ethernet interface for **half-duplex** operation:

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#half-duplex
```

ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<listname>	Specifies the assigned IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the router to only allow Telnet traffic (as configured in the user-configured **TelnetOnly** IP access list) into the Ethernet interface:

```
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#interface eth 0/1
ProCurve(config-eth 0/1)#ip access-group TelnetOnly in
```


ip address dhcp

Use the **ip address dhcp** command to use Dynamic Host Configuration Protocol (DHCP) to obtain an address on the Ethernet interface. Use the **no** form of this command to remove a configured IP address (using DHCP) and disable DHCP operation on the interface.

ip address dhcp [**client-id** [*<interface>* | *<identifier>*] **hostname** *<"string">*]

Syntax Description

client-id	Optional. Specifies the client identifier used when obtaining an IP address from a DHCP server.
<i><interface></i>	Specifying an interface defines the client identifier as the hexadecimal MAC address of the specified interface (including a hexadecimal number added to the front of the MAC address to identify the media type). For example, specifying the client-id ethernet 0/1 (where the Ethernet interface has a MAC address of d217.0491.1150) defines the client identifier as 01:d2:17:04:91:11:50 (where 01 defines the media type as Ethernet). Refer to <i>hardware-address <hardware-address> <type></i> on page 401 for a detailed listing of media types.
<i><identifier></i>	Specifies a custom client-identifier using a text string (that is converted to a hexadecimal equivalent) or 7 to 28 hexadecimal numbers (with colon delimiters). For example, a custom client identifier of 0f:ff:ff:ff:ff:51:04:99:a1 may be entered using the <i><identifier></i> option.
hostname	Specifies a text string (to override the global router name) to use as the name in the DHCP option 12 field.
<i><"string"></i>	String (encased in quotation marks) of up to 35 characters to use as the name of the host for DHCP operation.
no-default-route	Keyword used to specify that the Secure Router OS not install the default-route obtained via DHCP.
no-domain-name	Keyword used to specify that the Secure Router OS not install the domain-name obtained via DHCP.
no-nameservers	Keyword used to specify that the Secure Router OS not install the DNS servers obtained via DHCP.

Default Values

client-id	Optional. By default, the client identifier is populated using the following formula: TYPE: INTERFACE SPECIFIC INFO : MAC ADDRESS Where TYPE specifies the media type in the form of one hexadecimal byte (refer to <i>hardware-address <hardware-address> <type></i> on page 401 for a detailed listing of media types) and the MAC ADDRESS is the Media Access Control (MAC) address assigned to the first Ethernet interface in the unit in the form of six hexadecimal bytes. (For units with a single Ethernet interface, the MAC ADDRESS assigned to Ethernet 0/1 is used in this field).

hostname

<"string">

By default, the hostname is the name configured using the Global Configuration **hostname** command.

Functional Notes

Dynamic Host Configuration Protocol (DHCP) allows interfaces to acquire a dynamically assigned IP address from a configured DHCP server on the network. Many Internet Service Providers (ISPs) require the use of DHCP when connecting to their services. Using DHCP reduces the number of dedicated IP addresses the ISP must obtain. Consult your ISP to determine the proper values for the **client-id** and **hostname** fields.

Usage Examples

The following example enables DHCP operation on Ethernet interface 0/1:

```
ProCurve(config)#interface eth 0/1
```

```
ProCurve(config-eth 0/1)#ip address dhcp
```

ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface (only one primary address is allowed). Use the optional **secondary** keyword to define a secondary IP address. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional. Configures secondary IP addresses for the specified interface. Multiple secondary IP addresses may be assigned (no limit).

Default Values

By default, there are no assigned IP addresses.

Functional Notes

Use secondary IP addresses to allow dual subnets on a single interface (when you need more IP addresses than the primary subnet can provide). When using secondary IP addresses, avoid routing loops by verifying that all devices on the network segment are configured with secondary IP addresses on the secondary subnet.

Usage Examples

The following example configures a secondary IP address of **192.22.72.101/30**:

```
ProCurve(config)#interface ethernet 0/1
```

```
ProCurve(config-eth 0/1)#ip address 192.22.72.101 255.255.255.252 secondary
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip dhcp release

Use the **ip dhcp release** command to transmit a message to the DHCP server requesting termination of the IP address lease on that interface.

Caution

*If you are currently connected to the unit using a Telnet session through the Ethernet interface, using the **ip dhcp release** command will terminate your Telnet session and render your Telnet capability inoperable until a new IP address is assigned by the DHCP server.*

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Functional Notes

Dynamic Host Configuration Protocol (DHCP) allows interfaces to acquire a dynamically-assigned IP address from a configured DHCP server on the network. Many Internet Service Providers (ISPs) require the use of DHCP when connecting to their services. Using DHCP reduces the number of dedicated IP addresses the ISP must obtain.

Usage Examples

The following example releases the IP address assigned (by DHCP) on the Ethernet interface (eth 0/1):

```
ProCurve(config)#int eth 0/1
ProCurve(config-eth 0/1)#ip dhcp release
```

ip dhcp renew

Use the **ip dhcp renew** command to transmit a message to the DHCP server requesting renewal of the IP address lease on that interface.

Default Values

No defaults necessary for this command.

Functional Notes

Dynamic Host Configuration Protocol (DHCP) allows interfaces to acquire a dynamically assigned IP address from a configured DHCP server on the network. Many Internet Service Providers (ISPs) require the use of DHCP when connecting to their services. Using DHCP reduces the number of dedicated IP addresses the ISP must obtain.

Usage Examples

The following example renews the IP address assigned (by DHCP) on the Ethernet interface (eth 0/1):

```
ProCurve(config)#int eth 0/1
```

```
ProCurve(config-eth 0/1)#ip dhcp renew
```

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note *The **ip helper-address** command must be used in conjunction with the **ip forward-protocol** command to configure the Secure Router OS to forward UDP broadcast packets. See **ip forward-protocol udp <port number>** on page 318 for more information.*

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign helper address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is User Datagram Protocol (UDP).
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#ip helper-address 192.33.5.99
```

ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

immediate-leave	If only one host (or IGMP snooping switch) is connected to the interface when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	This command controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer (if no receiver responds) the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65,535 ms. Default: 1000 ms.
querier-timeout <seconds>	Number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60-300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65,535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically-connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
ProCurve(config-eth 0/1)#ip igmp last-member-query-interval 200
```

ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface, and to place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding. See *ip mcast-stub helper-address <ip address>* on page 327 and *ip mcast-stub upstream* on page 546 for more information.

Usage Examples

The following example enables multicast forwarding and IGMP on the interface:

```
ProCurve(config-eth 0/1)#ip mcast-stub downstream
```


ip mcast-stub helper-address *<ip address>*

Use the **ip mcast-stub helper-address** command to specify an IP address toward which IGMP host reports and leave messages are forwarded. This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub downstream** and **ip mcast-stub upstream** commands. Use the **no** form of this command to return to default.

Syntax Description

<i><ip address></i>	Specifies the address to which the IGMP host reports and leave messages are forwarded.

Default Values

By default, no helper-address is configured.

Functional Notes

The helper address is configured globally and applies to all multicast-stub downstream interfaces. The address specified may be the next upstream hop or any upstream address on the distribution tree for the multicast source, up to and including the multicast source. The router selects, from the list of multicast-stub upstream interfaces, the interface on the shortest path to the specified address. The router then proxies, on the selected upstream interface (using an IGMP host function), any host joins/leaves received on the downstream interface(s). The router retransmits these reports with addresses set as if the report originated from the selected upstream interface.

For example, if the router receives multiple joins for a group, it will not send any extra joins out the upstream interface. Also, if it receives a leave, it will not send a leave until it is certain that there are no more subscribers on any downstream interface.

Usage Examples

The following example specifies 172.45.6.99 as the helper address:

```
ProCurve(config)#ip mcast-stub helper-address 172.45.6.99
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface. See and *ip mcast-stub downstream* on page 544 for more information.

Usage Examples

The following example enables multicast forwarding on the interface:

```
ProCurve(config-eth 0/1)#ip mcast-stub upstream
```

ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key <password>	Assigns a simple-text authentication password to be used by other routers using the OSPF simple password authentication.
cost <value>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65,535.
dead-interval <seconds>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, the neighboring device is assumed to be down. Range: 0 to 32,767.
hello-interval <seconds>	Specifies the interval between hello packets sent on the interface. Range: 0 to 32,767.
message-digest-key <keyid> md5 <key>	Configures OSPF Message Digest 5 (MD5) authentication (16-byte max) keys.
priority <value>	Sets the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <seconds>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32,767.
transmit-delay <seconds>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32,767.

Default Values

dead-interval <seconds>	40 seconds
hello-interval <seconds>	10 seconds: Ethernet, point-to-point, Frame Relay, and PPP
retransmit-interval <seconds>	5 seconds
transmit-delay <seconds>	1 second

Usage Example

The following example sets the maximum number of seconds allowed between hello packets to 25,000:

```
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#ip ospf dead-interval 25000
```

ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Optional. Selects message-digest authentication type.
null	Optional. Specifies that no authentication be used.

Default Values

By default, this is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the Ethernet interface:

```
ProCurve(config-eth 0/1)#ip ospf authentication null
```

ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Sets the network type for broadcast.
point-to-point	Sets the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP and Frame Relay default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config-eth 0/1)#ip ospf network broadcast
```

ip proxy-arp <ip address> <subnet mask>

Use the **ip proxy-arp** command to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<ip address>	Defines the proxy ARP IP address in dotted decimal notation (for example: 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, ip proxy arp is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the Secure Router OS will respond to all arp requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the Ethernet interface:

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#ip proxy-arp
```

ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface.

Syntax Description

1	Accepts RIP version 1 packets received on the interface.
2	Accepts RIP version 2 packets received on the interface.

Default Values

By default, all interfaces implement RIP version 1.

Functional Notes

Use the **ip rip receive version** command to specify a RIP version that overrides the **version** (in the Router RIP) configuration.

The Secure Router OS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the Ethernet interface to accept only RIP version 2 packets:

```
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#ip rip receive version 2
```

ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface.

Syntax Description

1	Transmits RIP version 1 packets received on the interface.
2	Transmits RIP version 2 packets received on the interface.

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** command to specify a RIP version that overrides the **version** (in the Router RIP) configuration.

The Secure Router OS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the Ethernet interface to transmit only RIP version 2 packets:

```
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#ip rip send version 2
```


ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note *Using Network Address Translation (NAT) or the Secure Router OS firewall capabilities on an interface requires process switching mode (using the **no ip route-cache** command).*

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on the Ethernet interface:

```
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#ip route-cache
```

ip unnumbered <interface>

Use the **ip unnumbered** command to use the IP address assigned to the specified interface for all IP processing on the active interface. Use the **no** form of this command to remove the unnumbered configuration.

Syntax Description

<interface>	Specifies the interface (in the format type slot/port) that contains the IP address to be used as the source address for all packets transmitted on this interface.

Default Values

By default, all interfaces are configured to use a specified IP address (using the **ip address** command).

Functional Notes

If **ip unnumbered** is enabled on an interface, all IP traffic from the interface will use a source IP address taken from the specified interface. For example, specifying **ip unnumbered ppp 1** while in the Ethernet Interface Configuration mode configures the Ethernet interface to use the IP address assigned to the PPP interface for all IP processing. In addition, the Secure Router OS uses the specified interface information when sending route updates over the unnumbered interface.

Usage Examples

The following example configures the Ethernet interface (labeled **eth 0/1**) to use the IP address assigned to the PPP interface (**ppp 1**):

```
ProCurve(config)#interface eth 0/ 1
ProCurve(config-eth 0/1)#ip unnumbered ppp 1
```

lldp receive

Use the **lldp receive** command to allow LLDP packets to be received on this interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to send and receive LLDP packets.

Usage Examples

The following example configures Ethernet interface 0/1 to receive LLDP packets:

```
ProCurve(config-eth 0/1)#lldp receive
```

lldp send [management-address | port-description | system-capabilities | system-description | system-name | and-receive]

Use the **lldp send** command to configure this interface to transmit LLDP packets or to control the types of information contained in the LLDP packets transmitted by this interface.

Syntax Description

management-address	Enables transmission of management address information on this interface.
port-description	Enables transmission of port description information on this interface.
system-capabilities	Enables transmission of this device's system capabilities on this interface.
system-description	Enables transmission of this device's system description on this interface.
system-name	Enables transmission of this device's system name on this interface.
and-receive	Configures this interface to both transmit and receive LLDP packets.

Default Values

Be default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures Ethernet interface 0/1 to transmit LLDP packets containing all enabled information types:

```
ProCurve(config-eth 0/1)#lldp send
```

The following example configures Ethernet interface 0/1 to transmit and receive LLDP packets containing all information types:

```
ProCurve(config-eth 0/1)#lldp send-and-receive
```

lldp send-and-receive

Use the **lldp send-and-receive** command to configure this interface to transmit and receive LLDP packets.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures Ethernet interface 0/1 to transmit and receive LLDP packets containing all information types:

```
ProCurve(config-eth 0/1)#lldp send-and-receive
```

mac-address *<address>*

Use the **mac-address** command to specify the Media Access Control (MAC) address of the unit. The first three values contain the reserved Vendor ID (00:12:79) by default. Use the **no** form of this command to return to the default MAC address.

Syntax Description

<i><address></i>	MAC address entered in a series of six dual-digit hexadecimal values separated by colons (for example 00:12:79:5F:00:D2).

Default Values

A unique default MAC address is programmed in each unit.

Functional Notes

Some network providers require MAC address registration to connect to their networks. "Locking" access to the public network (based on MAC addresses) can cause problems for multi-computer offices. For example, many cable internet providers register the MAC address of your computer's Ethernet card, limiting the use of the network access to the registered computer. Use the **mac-address** command to program the computer's Ethernet card MAC address into the router to spoof the PC connection. (The cable internet provider cannot distinguish between a router using the registered MAC address and the actual registered computer.) Non-registered computers can connect to the network through the router.

Usage Examples

The following example configures a MAC address of **00:12:79:05:30:05**:

```
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#mac-address 00:12:79:05:30:05
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit (MTU) size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size>	Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:	
	ATM interfaces	64 to 1520
	Ethernet interfaces	64 to 1500
	FDL interfaces	64 to 256
	HDLC interfaces	64 to 1520
	Loopback interfaces	64 to 1500
	Tunnel interfaces	64 to 18,190
	Virtual Frame Relay sub-interfaces	64 to 1520
	Virtual PPP interfaces	64 to 1500

Default Values

<size>	The default values for the various interfaces are listed below:	
	ATM interfaces	1500
	Ethernet interfaces	1500
	FDL interfaces	256
	HDLC interfaces	1500
	Loopback interfaces	1500
	Tunnel interfaces	1500
	Virtual Frame Relay sub-interfaces	1500
	Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the Ethernet interface:

```
ProCurve(config)#interface eth 0/1
ProCurve(config-eth 0/1)#mtu 1200
```

port-auth supplicant enable [username <username> | password <password>]

Use the **port-auth supplicant enable** command to enable supplicant functionality and to specify the username and password used for IEEE 802.1x port authentication. The supplicant is the port that will receive services from the port authenticator.

Syntax Description

username <username>	Specifies the username to use during the authentication process. The default username is <i>username</i> .
password <password>	Specifies the password to use during the authentication process. The default password is <i>password</i> .

Default Values

By default, this command disabled.

Functional Notes

If your network infrastructure is configured to use 802.1x on every port, configure the router to function as an 802.1x client. The router (when configured as a 802.1x client) passes username and password information to the connected switch allowing the switch to authenticate the user (router) and enable the port.

Usage Examples

The following example sets the username to **User1** and sets the password to **securePass** for Ethernet interface 0/2:

```
ProCurve(config)#int eth 0/2
```

```
ProCurve(config-eth 0/2)#port-auth supplicant enable username User1 password securePass
```


qos-policy out <mapname>

Use the **qos-policy out** command to apply a previously-configured QoS map to an interface. Use the **no** form of this command to remove the map from the interface. The **out** keyword specifies that this policy will be applied to outgoing packets.

Syntax Description

<mapname>	Specifies the name of a previously-created QoS map (see <i>qos map</i> <mapname> <sequence number> on page 431 for more information).

Default Values

No default value is necessary for this command.

Usage Examples

The following example applies the QoS map **MYMAP** to the Ethernet 0/1 interface:

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#qos-policy out MYMAP
```

snmp trap

Use the **snmp trap** command to enable all supported Simple Network Management Protocol (SNMP) traps on the interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces (except virtual Frame Relay interfaces and sub-interfaces) have SNMP traps enabled.

Usage Examples

The following example enables SNMP capability on the Ethernet interface:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#snmp trap
```

snmp trap link-status

Use the **snmp trap link-status** command to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the interface:

```
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#no snmp trap link-status
```

spanning-tree bpdufilter [enable | disable]

Use the **spanning-tree bpdufilter** command to enable or disable the bpdufilter on a specific interface. This setting overrides the related global setting. Use the **no** version of the command to return to the default setting.

Syntax Description

enable	Enables bpdufilter for this interface.
disable	Disables bpdufilter for this interface.

Default Values

By default, this setting is disabled.

Functional Notes

The bpdufilter blocks any BPDUs from being transmitted and received on an interface.

Usage Examples

The following example enables the bpdufilter on the interface eth 0/1:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#spanning-tree bpdufilter enable
```

The bpdufilter can be disabled on the eth 0/1 by issuing the following commands:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#spanning-tree bpdufilter disable
```

spanning-tree bpduguard [enable | disable]

Use the **spanning-tree bpduguard** command to enable or disable the bpduguard on a specific interface. This setting overrides the related global setting (see *spanning-tree forward-time <seconds>* on page 387). Use the **no** version of the command to return to the default setting.

Syntax Description

enable	Enables bpduguard for this interface.
disable	Disables bpduguard for this interface.

Default Values

By default, this setting is disabled.

Functional Notes

The bpduguard blocks any BPDUs from being received on an interface.

Usage Examples

The following example enables the bpduguard on the interface eth 0/1:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#spanning-tree bpduguard enable
```

The bpduguard can be disabled on the eth 0/1 by issuing the following commands:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#spanning-tree bpduguard disable
```

spanning-tree edgeport

Use the **spanning-tree edgeport** command to configure the interface to be an edgeport. This command overrides the related Global setting. Use the **no** version of the command to return to the default setting.

Syntax Description

No subcommands.

Default Values

By default, this setting is disabled.

Functional Notes

Enabling this command configures the interface to go to a forwarding state when the link goes up.

Usage Examples

The following example configures the interface to be an edgeport:

```
ProCurve(config)#interface eth 0/1  
ProCurve(config-eth 0/1)#spanning-tree edgeport
```

An individual interface can be configured to not be considered an edgeport. For example:

```
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#spanning-tree edgeport disable
```

or

```
ProCurve(config)#interface ethernet 0/1  
ProCurve(config-eth 0/1)#no spanning-tree edgeport
```

spanning-tree link-type [auto | point-to-point | shared]

Use the **spanning-tree link-type** command to configure the spanning tree protocol link type for each interface. Use the **no** version of the command to return to the default setting.

Syntax Description

auto	Link type is determined by the port's duplex settings.
point-to-point	Link type is manually set to point-to-point, regardless of duplex settings.
shared	Link type is manually set to shared, regardless of duplex settings.

Default Values

By default, the interface is set to auto.

Functional Notes

This command overrides the default link type setting determined by the duplex of the individual port. By default, a port configured for half-duplex is set to **shared** link type, and a port configured for full-duplex is set to **point-to-point** link type. Setting the link type manually overrides the default and forces the port to use the specified link type. Use the **link-type auto** command to restore the convention of determining link type based on duplex settings.

Usage Examples

The following example forces the link type to **point-to-point**, even if the port is configured to be half-duplex:

```
ProCurve(config)#interface eth 0/1
```

```
ProCurve(config-eth 0/1)#spanning-tree link-type point-to-point
```

Technology Review

Rapid transitions are possible in RSTP (rapid spanning-tree protocol) by taking advantage of point-to-point links (a port is connected to exactly one other bridge) and edge-port connections (a port is not connected to any additional bridges). Setting the link-type to **auto** allows the spanning-tree to automatically configure the link type based on the duplex of the link. Setting the link type to **point-to-point** allows a half-duplex link to act as if it were a point-to-point link.

spanning-tree port-priority *<priority level>*

Use the **spanning-tree port-priority** command to select the priority level of this interface. To return to the default setting, use the **no** version of this command.

Syntax Description

<priority level> Specifies a value from 0 to 255.

Default Values

By default, this set to 128.

Functional Notes

The only time that this priority level is used is when two interfaces with a path to the root have equal cost. At that point, the level set in this command will determine which port the spanning-tree will use. Set the priority value lower to increase the chance the interface will be used.

Usage Examples

The following example sets the interface to a priority of 100:

```
ProCurve(config)#interface eth 0/1
```

```
ProCurve(config-eth 0/1)#spanning-tree port-priority 100
```


speed [10 | 100 | auto]

Use the **speed** command to configure the speed of an Ethernet interface. Use the **no** form of this command to return to the default value.

Syntax Description

10	10 Mb Ethernet
100	100 Mb Ethernet
auto	Automatically detects 10 or 100 Mb Ethernet and negotiates the duplex setting in the following order: 100/full, 100/half, 10/full, 10/half.

Note	<i>Some Ethernet equipment (though rare) is unable to negotiate duplex if speed is manually determined. To avoid incompatibilities, manually set the duplex if the speed is manually set. See speed [10 100 auto] on page 569 for more information.</i>

Default Values

By default, speed is set to **auto**.

Usage Examples

The following example configures the Ethernet port for 100 Mb operation:

```
ProCurve(config)#interface ethernet 0/1
ProCurve(config-eth 0/1)#speed 100
```

SERIAL INTERFACE CONFIGURATION COMMAND SET

To activate the Serial Interface Configuration mode, enter the **interface serial** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface serial 1/1
ProCurve(config-ser 1/1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

et-clock-source [rxclock | txclock] [on page 571](#)

ignore dcd [on page 572](#)

invert etclock [on page 573](#)

invert rxclock [on page 574](#)

invert txclock [on page 575](#)

serial-mode [v35 | x21] [on page 576](#)

snmp trap [on page 577](#)

snmp trap link-status [on page 578](#)

et-clock-source [rxclock | txclock]

Use the **et-clock-source** command to configure the clock source used when creating the external transmit (ET) reference clock. Use the **no** form of this command to return to the default value.

Syntax Description

rxclock	Specifies using the clock recovered from the receive signal to generate ET clock.
txclock	Specifies using the clock recovered from the transmit signal to generate ET clock.

Default Values

By default, the clock recovered from the transmit signal is used to generate the ET clock.

Functional Notes

The external transmit clock (ET clock) is an interface timing signal (provided by the DTE device) used to synchronize the transfer of transmit data.

Usage Examples

The following example configures the serial interface to recover the clock signal from the received signal and use it to generate ET clock:

```
ProCurve(config)#interface serial 1/1
ProCurve(config-ser 1/1)#et-clock-source rxclock
```

ignore dcd

Use the **ignore dcd** command to specify the behavior of the serial interface when the Data Carrier Detect (DCD) signal is lost. Use the **no** form of this command to return to the default value.

Syntax Description

No subcommands.

Default Values

By default, the serial interface does not ignore a change in status of the DCD signal.

Functional Notes

When configured to follow DCD (default condition), the serial interface will not attempt to establish a connection when DCD is not present. When configured to ignore DCD, the serial interface will continue to attempt to establish a connection even when DCD is not present.

Usage Examples

The following example configures the serial interface to ignore a loss of the DCD signal:

```
ProCurve(config)#interface serial 1/1  
ProCurve(config-ser 1/1)#ignore dcd
```

invert etclock

Use the **invert etclock** command to configure the serial interface to invert the external transmit (ET) reference clock in the data stream before transmitting. Use the **no** form of this command to return to the default value.

Syntax Description

No subcommands.

Default Values

By default, the serial interface does not invert ET clock.

Functional Notes

If the serial interface cable is long, causing a phase shift in the data, the ET clock can be inverted using the **invert etclock** command. This switches the phase of the clock, which compensates for a long cable.

Usage Examples

The following example configures the serial interface to invert ET clock:

```
ProCurve(config)#interface serial 1/1  
ProCurve(config-ser 1/1)#invert etclock
```

invert rxclock

Use the **invert rxclock** command to configure the serial interface to expect an inverted receive clock (found in the received data stream). Use the **no** form of this command to return to the default value.

Syntax Description

No subcommands.

Default Values

By default, the serial interface does not expect an inverted receive clock (**rxclock**).

Functional Notes

If the serial interface cable is long, causing a phase shift in the data, the transmit clock can be inverted (using the **invert txclock** command). This switches the phase of the clock, which compensates for a long cable. If the transmit clock of the connected device is inverted, use the **invert rxclock** command to configure the receiving interface appropriately.

Usage Examples

The following example configures the serial interface to invert receive clock:

```
ProCurve(config)#interface serial 1/1
ProCurve(config-ser 1/1)#invert rxclock
```

invert txclock

Use the **invert txclock** command to configure the serial interface to invert the transmit clock (found in the transmitted data stream) before sending the signal. Use the **no** form of this command to return to the default value.

Syntax Description

No subcommands.

Default Values

By default, the serial interface does not invert transmit clock (**txclock**).

Functional Notes

If the serial interface cable is long, causing a phase shift in the data, the transmit clock can be inverted (using the **invert txclock** command). This switches the phase of the clock, which compensates for a long cable. If the transmit clock of the connected device is inverted, use the **invert rxclock** command to configure the receiving interface appropriately.

Usage Examples

The following example configures the serial interface to invert the transmit clock:

```
ProCurve(config)#interface serial 1/1  
ProCurve(config-ser 1/1)#invert txclock
```

serial-mode [v35 | x21]

Use the **serial-mode** command to specify the electrical mode for the interface. Use the **no** form of this command to return to the default value.

Syntax Description

V35	Configures the interface for use with the V.35 adapter cable (J8757A).
X21	Configures the interface for use with the X.21 adapter cable (J8755A).

Default Values

By default, the serial interface is configured for a V.35 adapter cable.

Functional Notes

The pinouts for each of the available interfaces can be found in the *Hardware Configuration Guide* located on the *ProCurve SROS Documentation* CD (provided in your shipment).

Usage Examples

The following example configures the serial interface to work with the X.21 adapter cable:

```
ProCurve(config)#interface serial 1/1
ProCurve(config-ser 1/1)#serial-mode X21
```


snmp trap

Use the **snmp trap** command to enable all supported Simple Network Management Protocol (SNMP) traps on the interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces (except virtual Frame Relay interfaces and sub-interfaces) have SNMP traps enabled.

Usage Examples

The following example enables SNMP on the serial interface:

```
ProCurve(config)#interface serial 1/1  
ProCurve(config-ser 1/1)#snmp trap
```

snmp trap link-status

Use the **snmp trap link-status** command to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* per RFC2863 to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the serial interface:

```
ProCurve(config)#interface serial 1/1
ProCurve(config-ser 1/1)#no snmp trap link-status
```

T1 INTERFACE CONFIGURATION COMMAND SET

To activate the T1 Interface Configuration mode, enter the **interface t1** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)
bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)
shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

clock source [*internal* | *line* | *through*] [on page 580](#)
coding [*ami* | *b8zs*] [on page 581](#)
fdl [*ansi* | *att* | *none*] [on page 582](#)
framing [*d4* | *esf*] [on page 583](#)
lbo [*long* <-22.5, -15, -7.5, 0> | *short* <0-655>] [on page 584](#)
loopback commands [begin on page 585](#)
remote-alarm [*rai*] [on page 588](#)
remote-loopback [on page 589](#)
show test-pattern [on page 590](#)
snmp trap link-status [on page 591](#)
tdm-group <group number> *timeslots* <1-24> *speed* [*56* | *64*] [on page 592](#)
test-pattern [*clear* | *insert* | *ones* | *p215* | *p220* | *p511* | *qrss* | *zeros*] [on page 593](#)

clock source [internal | line | through]

Use the **clock source** command to configure the source timing used for the interface. Use the **no** form of this command to return to the default value.

Syntax Description

internal	Configures the unit to provide clocking using the internal oscillator.
line	Configures the unit to recover clocking from the T1 circuit.
through	Configures the unit to recover clocking from the circuit connected to the DSX-1 interface.

Default Values

By default, the **clock source** is set to **line**.

Functional Notes

When operating on a circuit that is providing timing, setting the **clock source** to **line** can avoid errors such as Clock Slip Seconds (CSS).

Usage Examples

The following example configures the unit to recover clocking from the primary circuit:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#clock source line
```

coding [ami | b8zs]

Use the **coding** command to configure the line coding for a T1 physical interface. This setting must match the line coding supplied on the circuit by the service provider.

Syntax Description

ami	Configures the line coding for alternate mark inversion (AMI).
b8zs	Configures the line coding for bipolar eight zero substitution (B8ZS).

Default Values

By default, all T1 interfaces are configured with B8ZS line coding.

Functional Notes

The line coding configured in the unit must match the line coding of the T1 circuit. A mismatch will result in line errors (e.g., BPVs).

Usage Examples

The following example configures the T1 interface for AMI line coding:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#coding ami
```

fdl [ansi | att | none]

Use the **fdl** command to configure the format for the facility data link (FDL) channel on the T1 circuit. FDL channels are only available on point-to-point circuits. Use the **no** form of this command to return to the default value.

Syntax Description

ansi	Configures the FDL for ANSI T1.403 standard.
att	Configures the FDL for AT&T TR 54016 standard.
none	Disables FDL on this circuit.

Default Values

By default, the FDL is configured for **ansi**.

Functional Notes

T1 circuits using ESF framing format (specified using the **framing** command) reserve 12 bits as a data link communication channel, referred to as the FDL, between the equipment on either end of the circuit. The FDL allows the transmission of trouble flags such as the Yellow Alarm signal. See *framing [d4 | esf]* on page 583 for related information.

Usage Examples

The following example disables the FDL channel for the T1 circuit:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#fdl none
```

framing [d4 | esf]

Use the **framing** command to configure the framing format for the T1 interface. This parameter should match the framing format supplied by your network provider. Use the **no** form of this command to return to the default value.

Syntax Description

d4	Specifies D4 superframe (SF) format.
esf	Specifies extended superframe (ESF) format.

Default Values

By default, the framing format is configured for **esf**.

Functional Notes

A frame is comprised of a single byte from each of the T1's timeslots; there are 24 timeslots on a single T1 circuit. Framing bits are used to separate the frames and indicate the order of information arriving at the receiving equipment. D4 and ESF are two methods of collecting and organizing frames over the circuit.

Usage Examples

The following example configures the T1 interface for D4 framing:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#framing d4
```

lbo [long <-22.5, -15, -7.5, 0> | short <0-655>]

Use the **lbo** command to configure the line build out (LBO) for the T1 interface. Use the **no** form of this command to return to the default value.

Syntax Description

long <-22.5, -15, -7.5, 0> Configures the LBO (in dB) for T1 interfaces with cable lengths greater than 655 feet. Choices are -22.5, -15, -7.5, and 0 dB.

short <0-655> Configures the LBO (in feet) for T1 interfaces with cable lengths less than 655 feet. Range is 0 to 655 feet.

Default Values

By default, the build out is set to 0 dB.

Functional Notes

Line build out (LBO) is artificial attenuation of a T1 output signal to simulate a degraded signal. This is useful to avoid overdriving a receiver's circuits. The shorter the distance between T1 equipment (measured in cable length), the greater the attenuation value. For example, two units in close proximity should be configured for the maximum attenuation (-22.5 dB).

Usage Examples

The following example configures the T1 interface LBO for -22.5 dB:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#lbo long -22.5
```


loopback network [line | payload]

Use the **loopback network** command to initiate a loopback on the interface toward the network. Use the **no** form of this command to deactivate the loopback.

Syntax Description

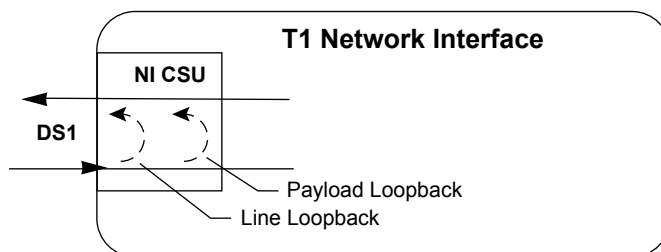
line	Initiates a metallic loopback of the physical T1 network interface.
payload	Initiates a loopback of the T1 framer (CSU portion) of the T1 network interface.

Default Values

No default necessary for this command.

Functional Notes

The following diagram depicts the difference between a line and payload loopback.



Usage Examples

The following example initiates a payload loopback of the T1 interface:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#loopback network payload
```

loopback remote line [fdl | inband]

Use the **loopback remote line** command to send a loopback code to the remote unit to initiate a line loopback. Use the **no** form of this command to send a loopdown code to the remote unit to deactivate the loopback.

Syntax Description

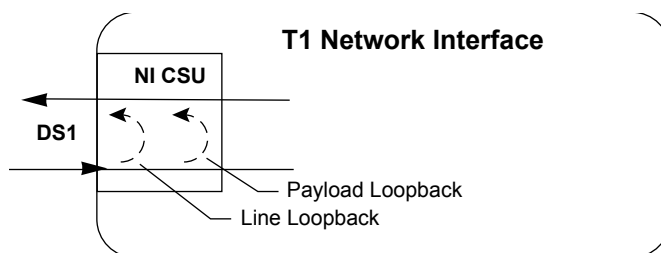
fdl	Uses the facility data link (FDL) to initiate a full 1.544 Mbps physical (metallic) loopback of the signal received by the remote unit from the network.
inband	Uses the inband channel to initiate a full 1.544 Mbps physical (metallic) loopback of the signal received by the remote unit from the network.

Default Values

No defaults necessary for this command.

Functional Notes

The following diagram depicts the difference between a line and payload loopback.



Usage Examples

The following example initiates a remote line loopback using the FDL:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#loopback remote line fdl
```

loopback remote payload

Use the **loopback remote payload** command to send a loopback code to the remote unit to initiate a payload loopback. A payload loopback is a 1.536 Mbps loopback of the payload data received from the network maintaining bit-sequence integrity for the information bits by synchronizing (regenerating) the timing. Use the **no** form of this command to send a loopdown code to the remote unit to deactivate the loopback.

Syntax Description

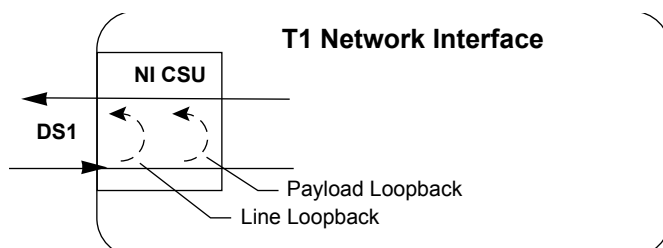
No subcommands.

Default Values

No defaults necessary for this command.

Functional Notes

The following diagram depicts the difference between a line and payload loopback.



Usage Examples

The following example initiates a remote payload loopback:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#loopback remote payload
```

remote-alarm [rai]

The **remote-alarm** command selects the alarm signaling type to be sent when a loss of frame is detected on the T1 receive signal. Use the **no** form of this command to disable all transmitted alarms.

Syntax Description

rai	Specifies sending a remote alarm indication (RAI) in response to a loss of frame. Also prevents a received RAI from causing a change in interface operational status.

Default Values

The default for this command is rai.

Usage Examples

The following example enables transmission of RAI in response to a loss of frame:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#remote-alarm rai
```

remote-loopback

Use the **remote-loopback** command to configure the interface to respond to loopbacks initiated by a remote unit (or the service provider). Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

By default, all interfaces respond to remote loopbacks.

Usage Examples

The following example enables remote loopbacks on the T1 interface:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#remote-loopback
```

show test-pattern

Use the **show test-pattern** command to display results from test patterns inserted using the **test-pattern** command (see *test-pattern [clear | insert | ones | p215 | p220 | p511 | qrss | zeros]* on page 593 for more information).

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following is sample output from this command:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#show test-pattern
Qrss Errored Seconds: 6
```

snmp trap link-status

Use the **snmp trap link-status** to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the T1 interface:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#no snmp trap link-status
```

tdm-group <group number> **timeslots** <1-24> **speed** [56 | 64]

Use the **tdm-group** command to create a group of contiguous DS0s on this interface to be used during the **bind** process. See *crypto map* <mapname> on page 875 for related information.

Caution *Changing **tdm-group** settings could result in service interruption.*

Syntax Description

<group number>	Identifies the created tdm-group by numerical label (valid range: 1 to 255).
timeslots <1-24>	Specifies the DS0s to be used in the TDM group. This can be entered as a single number representing one of the 24 T1 channel timeslots or as a contiguous group of DS0s. (For example, 1-10 specifies the first 10 channels of the T1.)
speed [56 64]	Optional. Specifies the individual DS0 rate on the T1 interface to be 56 or 64 kbps. The default speed is 64 kbps. 56 kbps operation is not available on all T1 interfaces. Refer to the Quick Start Guide provided with your T1 module to determine whether 56 kbps is valid.

Default Values

By default, there are no configured TDM group.

Usage Examples

The following example creates a TDM group (labeled **5**) of 10 DS0s at 64 kbps each:

```
ProCurve(config)#interface t1 1/1
```

```
ProCurve(config-t1 1/1)#tdm-group 5 timeslots 1-10 speed 64
```


test-pattern [clear | insert | ones | p215 | p220 | p511 | qrss | zeros]

Use the **test-pattern** command to activate the built-in pattern generator and begin sending the specified test pattern. This pattern generation can be used to verify a data path when used in conjunction with an active loopback. Use the **no** form of this command to cease pattern generation.

Syntax Description

clear	Clears the test pattern error count. Display the error count using the show test-pattern command.
insert	Inserts an error into the currently active test pattern. Display the injected error result by using the show test pattern command.
ones	Generates a test pattern of continuous ones.
p215	Generates a pseudorandom test pattern sequence based on a 15-bit shift register.
p220	Generates a pseudorandom test pattern sequence based on a 20-bit shift register.
p511	Generates a repeating test pattern of ones and zeros.
qrss	Generates a random test pattern of ones and zeroes.
zeros	Generates a test pattern of continuous zeros.

Default Values

No defaults necessary for this command.

Usage Examples

The following example activates the pattern generator for a stream of continuous ones:

```
ProCurve(config)#interface t1 1/1
ProCurve(config-t1 1/1)#test-pattern ones
```

DSX-1 INTERFACE CONFIGURATION COMMAND SET

To activate the DSX-1 Interface Configuration mode, enter the **interface t1** command (and specify the DSX-1 port) at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface t1 1/2
ProCurve(config-t1 1/2)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

coding [ami | b8zs] [on page 595](#)

framing [d4 | esf] [on page 596](#)

line-length <value> [on page 597](#)

loopback network [line | payload] [on page 598](#)

loopback remote line inband [on page 599](#)

remote-loopback [on page 600](#)

signaling-mode [message-oriented | none | robbed-bit] [on page 601](#)

snmp trap link-status [on page 602](#)

test-pattern [ones | zeros] [on page 603](#)

coding [ami | b8zs]

Use the **coding** command to configure the line coding for a DSX-1 physical interface. This setting must match the line coding supplied on the circuit by the external device.

Syntax Description

ami	Configures the line coding for alternate mark inversion (AMI).
b8zs	Configures the line coding for bipolar eight zero substitution (B8ZS).

Default Values

By default, all DSX-1 interfaces are configured with B8ZS line coding.

Functional Notes

The line coding configured in the unit must match the line coding of the DSX-1 circuit. A mismatch will result in line errors (e.g., BPVs).

Usage Examples

The following example configures the DSX-1 interface for AMI line coding:

```
ProCurve(config)#interface t1 1/2
ProCurve(config-t1 1/2)#coding ami
```

framing [d4 | esf]

Use the **framing** command to configure the framing format for the DSX-1 interface. This parameter should match the framing format set on the external device. Use the **no** form of this command to return to the default value.

Syntax Description

d4	Specifies D4 superframe (SF) format.
esf	Specifies extended superframe (ESF) format.

Default Values

By default, the framing format is set to **esf**.

Functional Notes

A frame is comprised of a single byte from each of the T1's timeslots; there are 24 timeslots on a single T1 circuit. Framing bits are used to separate the frames and indicate the order of information arriving at the receiving equipment. D4 and ESF are two methods of collecting and organizing frames over the circuit.

Usage Examples

The following example configures the DSX-1 interface for D4 framing:

```
ProCurve(config)#interface t1 1/2  
ProCurve(config-t1 1/2)#framing d4
```

line-length <value>

Use the **line-length** command to set the line build out (in feet or dB) for the DSX-1 interface. Use the **no** form of this command to return to the default value.

Syntax Description

<value>	Configures the line build out for the DSX-1 interface.

Valid options include: -7.5 dB or <0 to 655> feet.

Default Values

By default, the line build out is set to 0 feet.

Functional Notes

The **line-length** value represents the physical distance between DSX equipment (measured in cable length). Based on this setting, the Secure Router OS device increases signal strength to compensate for the distance the signal must travel. Valid distance ranges are listed below:

- 0 to 133 feet
- 134 to 265 feet
- 266 to 399 feet
- 400 to 533 feet
- 534 to 655 feet

Use the **-7.5 dB** setting for maximum signal attenuation.

Usage Examples

The following example configures the DSX-1 interface **line-length** for 300 feet:

```
ProCurve(config)#interface t1 1/2
ProCurve(config-t1 1/2)#line-length 300
```

loopback network [line | payload]

Use the **loopback network** command to initiate a loopback on the interface toward the network. Use the **no** form of this command to deactivate the loopback.

Syntax Description

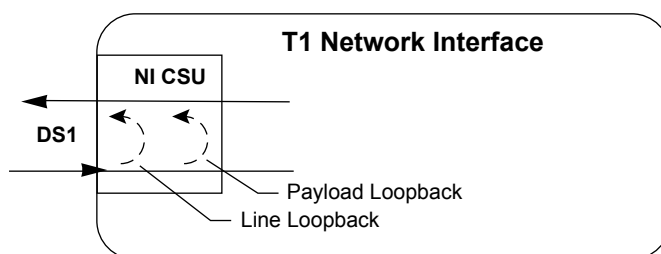
line	Initiates a metallic loopback of the physical DSX-1 network interface.
payload	Initiates a loopback of the T1 framer (CSU portion) of the DSX-1 network interface.

Default Values

No default necessary for this command.

Functional Notes

The following diagram depicts the difference between a line and payload loopback.



Usage Examples

The following example initiates a payload loopback of the DSX-1 interface:

```
ProCurve(config)#interface t1 1/2
ProCurve(config-t1 1/2)#loopback network payload
```

loopback remote line inband

Use the **loopback remote line inband** command to send a loopback code to the remote unit to initiate a line loopback. Use the **no** form of this command to send a loopdown code to the remote unit to deactivate the loopback.

Syntax Description

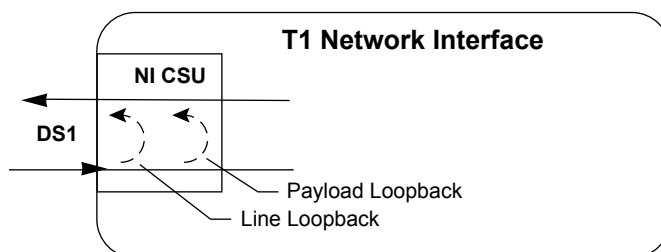
inband	Uses the inband channel to initiate a full 1.544 Mbps physical (metallic) loopback of the signal received by the remote unit from the network.

Default Values

No defaults necessary for this command.

Functional Notes

A remote loopback can only be issued if a bind does not exist on the interface and if the signaling mode is set to **none**. The following diagram depicts the difference between a line and payload loopback.



Usage Examples

The following example initiates a remote line loopback using the inband channel:

```
ProCurve(config)#interface t1 1/2
```

```
ProCurve(config-t1 1/2)#loopback remote line inband
```

remote-loopback

Use the **remote-loopback** command to configure the interface to respond to loopbacks initiated by a remote unit (or the service provider). Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

By default, all interfaces respond to remote loopbacks.

Usage Examples

The following example enables remote loopbacks on the DSX-1 interface:

```
ProCurve(config)#interface t1 1/2  
ProCurve(config-t1 1/2)#remote-loopback
```


signaling-mode [message-oriented | none | robbed-bit]

Use the **signaling-mode** command to configure the signaling type (robbed bit for voice or clear channel for data) for the DS0s mapped to the DSX-1 port.

Syntax Description

message-oriented	Specifies clear channel signaling on Channel 24 only. Use this signaling type with QSIG installations.
none	Specifies clear channel signaling on all 24 DS0s. Use this signaling type with data-only or PRI DSX-1 installations.
robbed-bit	Specifies robbed bit signaling on all DS0s. Use this signaling type for voice-only DSX-1 applications.

Default Values

By default, the signaling mode is set to robbed-bit.

Usage Examples

The following example configures the DSX-1 port for PRI compatibility:

```
ProCurve(config)#interface t1 1/2  
ProCurve(config-t1 1/2)#signaling-mode none
```

snmp trap link-status

Use the **snmp trap link-status** to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the DSX-1 interface:

```
ProCurve(config)#interface t1 1/2  
ProCurve(config-t1 1/2)#no snmp trap link-status
```

test-pattern [ones | zeros]

Use the **test-pattern** command to activate the built-in pattern generator and begin sending the specified test pattern. This pattern generation can be used to verify a data path when used in conjunction with an active loopback. Use the **no** form of this command to cease pattern generation.

Syntax Description

ones	Generates a test pattern of continuous ones.
zeros	Generates a test pattern of continuous zeros.

Default Values

No defaults necessary for this command.

Usage Examples

The following example activates the pattern generator for a stream of continuous ones:

```
ProCurve(config)#interface t1 1/2  
ProCurve(config-t1 1/2)#test-pattern ones
```

E1 INTERFACE CONFIGURATION COMMAND SET

To activate the E1 Interface Configuration mode, enter the **interface e1** command (and specify the E1 port) at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

clock source [*internal* | *line* | *through*] [on page 605](#)

coding [*ami* | *hdb3*] [on page 606](#)

framing [*crc4*] [on page 607](#)

loop-alarm-detect [on page 608](#)

loopback network [*line* | *payload*] [on page 609](#)

loopback remote v54 [on page 610](#)

remote-alarm [*rai* | *ais*] [on page 611](#)

remote-loopback [on page 612](#)

sa4tx-bit [*0* | *1*] [on page 613](#)

show test-pattern [on page 614](#)

snmp trap link-status [on page 615](#)

tdm-group <group number> timeslots <1-31> speed [*56* | *64*] [on page 616](#)

test-pattern [*clear* | *insert* | *ones* | *p215* | *p220* | *p511* | *qrss* | *zeros*] [on page 617](#)

ts16 [on page 618](#)

clock source [internal | line | through]

Use the **clock source** command to configure the source timing used for the interface. Use the **no** form of this command to return to the default value.

Syntax Description

internal	Configures the unit to provide clocking using the internal oscillator.
line	Configures the unit to recover clocking from the E1 circuit.
through	Configures the unit to recover clocking from the circuit connected to the G.703 interface.

Default Values

By default, the unit is configured to recover clocking from the primary circuit.

Functional Notes

When operating on a circuit that is providing timing, setting the **clock source** to **line** can avoid errors such as Clock Slip Seconds (CSS).

Usage Examples

The following example configures the unit to recover clocking from the primary circuit:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#clock source line
```

coding [ami | hdb3]

Use the **coding** command to configure the line coding for the E1 physical interface. This setting must match the line coding supplied on the circuit by the service provider.

Syntax Description

ami	Configures the line coding for alternate mark inversion (AMI).
hdb3	Configures the line coding for high-density bipolar 3 (HDB3).

Default Values

By default, all E1 interfaces are configured with HDB3 line coding.

Functional Notes

The line coding configured in the unit must match the line coding of the E1 circuit. A mismatch will result in line errors (e.g., BPVs).

Usage Examples

The following example configures the E1 interface for AMI line coding:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#coding ami
```

framing [crc4]

Use the **framing** command to configure the framing format for the E1 interface. This parameter should match the framing format set on the external device. Use the **no** form of this command to return to the default value.

Syntax Description

crc4	Enables CRC-4 bits to be transmitted in the outgoing data stream. Also, the received signal is checked for CRC-4 errors.

Default Values

By default, CRC4 is disabled.

Functional Notes

The framing value must match the configuration of the E1 circuit. A mismatch will result in a loss of frame alarm.

Usage Examples

The following example configures the E1 interface for CRC-4 framing:

```
ProCurve(config)#interface e1 1/1  
ProCurve(config-e1 1/1)#framing crc4
```

loop-alarm-detect

The **loop-alarm-detect** command enables detection of a loop alarm on the E1 interface. Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Functional Notes

This command enables the detection of a loopback alarm. This alarm works in conjunction with the **sa4tx-bit** command setting. The loopback condition is detected by comparing the transmitted **sa4tx-bit** value to the received Sa4 bit value. If the bits match, a loopback is assumed. This detection method only works with a network in which the far end is transmitting the opposite value for Sa4.

Usage Examples

The following example enables detection of a loop alarm on the E1 interface:

```
ProCurve(config)#config e1 1/1
```

```
ProCurve(config-e1 1/1)#loop-alarm-detect
```


loopback network [line | payload]

Use the **loopback network** command to initiate a loopback on the interface toward the network. Use the **no** form of this command to deactivate the loopback.

Syntax Description

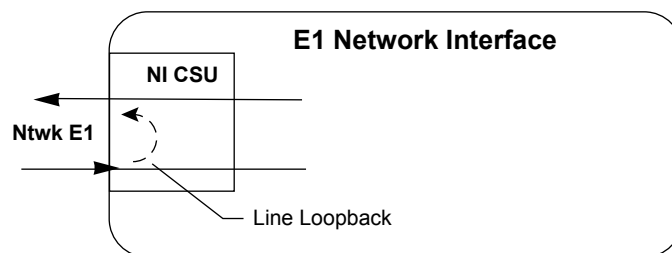
line	Initiates a metallic loopback of the physical E1 network interface.
payload	Initiates a loopback of the E1 framer (CSU portion) of the E1 network interface.

Default Values

No default necessary for this command.

Functional Notes

The following diagram depicts a line loopback.



Usage Examples

The following example initiates a line loopback of the E1 interface:

```
ProCurve(config)#interface e1 1/1  
ProCurve(config-e1 1/1)#loopback network line
```

loopback remote v54

The **loopback remote v54** command initiates an E1 remote loopback test (with a V.54 loopback pattern). Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

No default value is necessary for this command.

Functional Notes

This command causes a V.54 inband loop code to be sent in the payload towards the far end.

Usage Examples

The following example sends a V.54 inband loop code to the far end:

```
ProCurve(config)#interface e1 1/1  
ProCurve(config-e1 1/1)#loopback remote v54
```

remote-alarm [rai | ais]

The **remote-alarm** command selects the alarm signaling type to be sent when a loss of frame is detected on the E1 receive signal. Use the **no** form of this command to disable all transmitted alarms.

Syntax Description

rai	Specifies sending a remote alarm indication (RAI) in response to a loss of frame. Also prevents received RAI from causing a change in interface operational status.
ais	Specifies choosing to send an alarm indication signal (AIS) as an unframed all-ones signal.

Default Values

The default for this command is rai.

Functional Notes

An E1 will respond to a loss of frame on the receive signal by transmitting a remote alarm to the far end to indicate the error condition. TS0 of an E1 contains the Frame Alignment Signal (FAS) in the even-numbered frames. The odd-numbered frames are not used for frame alignment, and some of those bits are labeled as spare bits (SA bits) in bit positions 4 through 8.

Usage Examples

The following example enables transmission of AIS in response to a loss of frame:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#remote alarm ais
```

remote-loopback

Use the **remote-loopback** command to configure the interface to accept loopback requests initiated by a remote unit (or the service provider). Use the **no** form of this command to disable this feature.

Syntax Description

No subcommands.

Default Values

By default, all interfaces respond to remote loopbacks.

Functional Notes

This controls the acceptance of any remote loopback requests. When enabled, remote loopbacks are detected and cause a loopback to be applied. When disabled, remote loopbacks are ignored.

Usage Examples

The following example enables remote loopbacks on the E1 interface:

```
ProCurve(config)#interface e1 1/1
```

```
ProCurve(config-e1 1/1)#remote-loopback
```

sa4tx-bit [0 | 1]

The **sa4tx-bit** command selects the Tx value of Sa4 in this E1 interface. Use the **no** form of this command to return to the default value of 1.

Syntax Description

No subcommands.

Default Values

The default value for this command is 1.

Functional Notes

This command assigns a value to the Tx spare bit in position 4. The odd-numbered frames of TS0 are not used for frame alignment. Bits in position 4 through 8 are called spare bits. Values of 0 or 1 are accepted.

TS0 odd frame

Bit position	1	2	3	4	5	6	7	8
Bit use	0	1	RAI = 1	S	S	S	S	S

Usage Examples

The following example sets the Tx value of Sa4 to 0:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#sa4tx-bit 0
```

show test-pattern

Use the **show test-pattern** command to display results from test patterns inserted using the **test-pattern** command (see *test-pattern [clear | insert | ones | p215 | p220 | p511 | qrss | zeros]* on page 617 for more information).

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following is sample output from this command:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#show test-pattern
Qrss Errored Seconds: 6
```

snmp trap link-status

Use the **snmp trap link-status** to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the E1 interface:

```
ProCurve(config)#interface e1 1/1  
ProCurve(config-e1 1/1)#no snmp trap link-status
```

tdm-group <group number> **timeslots** <1-31> **speed** [56 | 64]

Use the **tdm-group** command to create a group of contiguous channels on this interface to be used during the **bind** process. See *crypto map* <mapname> on page 875 for related information.

Caution *Changing **tdm-group** settings could result in service interruption.*

Syntax Description

<group number>	Identifies the created TDM group by numerical label (Range: 1 to 255).
timeslots <1-31>	Specifies the channels to be used in the TDM group. This can be entered as a single number representing one of the 31 E1 channel timeslots or as a contiguous group of channels. (For example, 1-10 specifies the first 10 channels of the E1.)
speed [56 64]	Optional. Specifies the individual channel rate on the E1 interface to be 56 or 64 kbps. The default speed is 64 kbps. 56 kbps operation is not available on all E1 interfaces. Refer to the Quick Start Guide provided with your E1 module to determine whether 56 kbps is valid.

Default Values

By default, there are no configured TDM groups.

Usage Examples

The following example creates a TDM group (labeled **5**) of 10 channels at 64 kbps each:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#tdm-group 5 timeslots 1-10 speed 64
```


test-pattern [clear | insert | ones | p215 | p220 | p511 | qrss | zeros]

Use the **test-pattern** command to activate the built-in pattern generator and begin sending the specified test pattern. This pattern generation can be used to verify a data path when used in conjunction with an active loopback. Use the **no** form of this command to cease pattern generation.

Syntax Description

clear	Clears the test pattern error count. Display the error count using the show test-pattern command.
insert	Inserts an error into the currently active test pattern. Display the injected error result using the show test-pattern command.
ones	Generates a test pattern of continuous ones.
p215	Generates a pseudorandom test pattern based on a 15-bit shift register.
p220	Generates a pseudorandom test pattern based on a 20-bit shift register.
p511	Generates a test pattern of repeating ones and zeros.
qrss	Generates a test pattern of random ones and zeros.
zeros	Generates a test pattern of continuous zeros.

Default Values

No defaults necessary for this command.

Usage Examples

The following example activates the pattern generator for a stream of continuous ones:

```
ProCurve(config)#interface e1 1/1  
ProCurve(config-e1 1/1)#test-pattern ones
```

ts16

Use the **ts16** command to enable timeslot 16 multiframe to be checked on the receive signal. Use the **no** form of this command to disable timeslot 16.

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Functional Notes

If timeslot 16 is used on the incoming E1, do not map timeslot 16 using the **tdm-group** command. By default, all timeslots not physically mapped using the **tdm-group** command are passed through to the G.703 interface. Leaving timeslot 16 unmapped makes it available for multiframe signaling by the connected E1 device.

Usage Examples

The following example enables timeslot 16 multiframe:

```
ProCurve(config)#interface e1 1/1
ProCurve(config-e1 1/1)#ts16
```

G.703 INTERFACE CONFIGURATION COMMAND SET

To activate the G.703 Interface Configuration mode, enter the **interface e1** command (and specify the G.703 port) at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface e1 1/2
ProCurve(config-e1 1/2)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

coding [ami | hdb3] [on page 620](#)

framing [crc4] [on page 621](#)

loopback network [line | payload] [on page 622](#)

snmp trap link-status [on page 623](#)

test-pattern [ones | zeros] [on page 624](#)

ts16 [on page 625](#)

coding [ami | hdb3]

Use the **coding** command to configure the line coding for the G.703 physical interface. This setting must match the line coding supplied on the circuit by the PBX.

Syntax Description

ami	Configures the line coding for alternate mark inversion (AMI).
hdb3	Configures the line coding for high-density bipolar 3 (HDB3).

Default Values

By default, all E1 interfaces are configured with HDB3 line coding.

Functional Notes

The line coding configured in the unit must match the line coding of the E1 circuit. A mismatch will result in line errors (e.g., BPVs).

Usage Examples

The following example configures the G.703 interface for AMI line coding:

```
ProCurve(config)#interface e1 1/2
ProCurve(config-e1 1/2)#coding ami
```

framing [crc4]

Use the **framing** command to configure the framing format for the G.703 interface. This parameter should match the framing format set on the external device. Use the **no** form of this command to return to the default value.

Syntax Description

crc4	Enables CRC4 bits to be transmitted in the outgoing data stream. Also, the received signal is checked for CRC4 errors.

Default Values

By default, CRC4 is enabled.

Functional Notes

The framing value must match the configuration of the E1 circuit. A mismatch will result in a loss of frame alarm.

Usage Examples

The following example configures the G.703 interface for CRC4 framing:

```
ProCurve(config)#interface e1 1/2  
ProCurve(config-e1 1/2)#framing crc4
```

loopback network [line | payload]

Use the **loopback network** command to initiate a loopback on the interface toward the network. Use the **no** form of this command to deactivate the loopback.

Syntax Description

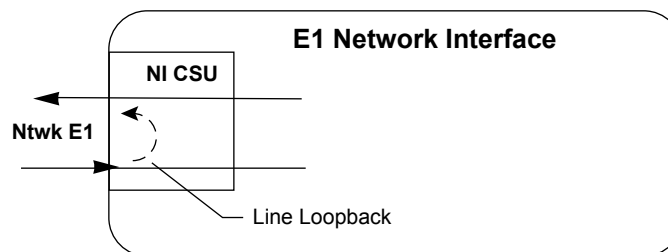
line	Initiates a metallic loopback of the physical E1 network interface.
payload	Initiates a loopback of the E1 framer (CSU portion) of the E1 network interface.

Default Values

No default necessary for this command.

Functional Notes

The following diagram depicts a line loopback.



Usage Examples

The following example initiates a line loopback of the G.703 interface:

```
ProCurve(config)#interface e1 1/2
ProCurve(config-e1 1/2)#loopback network line
```

snmp trap link-status

Use the **snmp trap link-status** to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the G.703 interface:

```
ProCurve(config)#interface e1 1/2  
ProCurve(config-e1 1/2)#no snmp trap link-status
```

test-pattern [ones | zeros]

Use the **test-pattern** command to activate the built-in pattern generator and begin sending the specified test pattern. This pattern generation can be used to verify a data path when used in conjunction with an active loopback. Use the **no** form of this command to cease pattern generation.

Syntax Description

ones	Generates a test pattern of continuous ones.
zeros	Generates a test pattern of continuous zeros.

Default Values

No defaults necessary for this command.

Usage Examples

The following example activates the pattern generator for a stream of continuous ones:

```
ProCurve(config)#interface e1 1/2
ProCurve(config-e1 1/2)#test-pattern ones
```


ts16

Use the **ts16** command to enable timeslot 16 multiframe to be checked on the receive signal. Use the **no** form of this command to disable timeslot 16.

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Functional Notes

If timeslot 16 is used on the incoming E1, do not map timeslot 16 using the **tdm-group** command. By default, all timeslots not physically mapped using the **tdm-group** command are passed through to the G.703 interface. Leaving timeslot 16 unmapped makes it available for multiframe signaling by the connected E1 device.

Usage Examples

The following example enables timeslot 16 multiframe:

```
ProCurve(config)#interface e1 1/2
ProCurve(config-e1 1/2)#ts16
```

MODEM INTERFACE CONFIGURATION COMMAND SET

To activate the Modem Interface Configuration mode, enter the **interface modem** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface modem 1/2
ProCurve(config-modem 1/2)#
```

Note *The modem interface number in the example above is shown as **modem 1/2**. This number is based on the interface's location (slot/port) and could vary depending on the unit's configuration. Use the **do show interfaces** command to determine the appropriate interface number.*

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

caller-id override [*always* <number> | *if-no-cid* <number>] [on page 627](#)

dialin [on page 628](#)

init-string <string> [on page 629](#)

caller-id override [always <number> | if-no-cid <number>]

Use the **caller-id override** command to configure the unit to replace caller ID information with a user-specified number. Use the **no** form of this command to disable any caller ID overrides.

Syntax Description

always <number>	Always forces replacement of the incoming caller ID number with the number given.
if-no-cid <number>	Replaces the incoming caller ID number with the number given only if there is no caller ID information available for the incoming call.

Default Values

By default, this command is disabled.

Functional Notes

Forces a replacement of the incoming caller ID number with the number given. The received caller ID, if any, is discarded, and the given override number is used to connect the incoming call to a circuit of the same number.

Usage Examples

The following example configures the unit to always provide the given number as the caller ID number:

```
ProCurve(config)#interface modem 1/2
```

```
ProCurve(config-modem 1/2)#caller-id override always 5555555
```

dialin

Use the **dialin** command to enable the modem for remote console dial-in, disabling the use of the modem for backup.

Syntax Description

No subcommands.

Default Values

By default, **dialin** is disabled.

Usage Examples

The following example enables remote console dial-in:

```
ProCurve(config)#interface modem 1/2
```

```
ProCurve(config-modem 1/2)#dialin
```

init-string <string>

Use the **init-string** command to specify an initialization string for the modem using standard AT commands. Use the **no** form of this command to return to the default initialization string.

Syntax Description

<string>	Specifies an initialization string using standard AT commands. This string must start with AT and cannot contain spaces.

Default Values

<string>	ate0q0v1x4\n0
at	All initialization strings must begin with AT.
e0	Disables command echo.
q0	Response messages on.
v1	Formats result codes in long word form.
x4	Specifies extended response set, dial tone, and busy signal detection for result codes following modem operations.
\n0	Selects standard buffered connection only.

Usage Examples

The following example configures the modem to perform a hang-up at each initialization (to verify that the line is free) and maintains the default initialization:

```
ProCurve(config)#interface modem 1/2
```

```
ProCurve(config-modem 1/2)#init-string ate0h0q0v1x4\n0
```

BRI INTERFACE CONFIGURATION COMMAND SET

To activate the BRI Interface Configuration mode, enter the **interface bri** command at the Global Configuration mode prompt. For example:

```
Router>enable
Router#configure terminal
RouterProCurve(config)#interface bri 1/2
Router(config-bri 1/2)#
```

Note *The BRI interface number in the example above is shown as **bri 1/2**. This number is based on the interface's location (slot/port) and could vary depending on the unit's configuration. Use the **do show interfaces** command to determine the appropriate interface number.*

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

bonding commands [begin on page 631](#)

caller-id override [*always* <number> | *if-no-cid* <number>] [on page 637](#)

isdn ldn1 <ldn> [on page 638](#)

isdn ldn2 <ldn> [on page 639](#)

isdn spid1 <spid> [on page 640](#)

isdn spid2 <spid> [on page 641](#)

isdn switch-type [*basic-5ess* | *basic-dms* | *basic-net3* | *basic-ni*] [on page 642](#)

bonding txadd-timer <seconds>

Use the **bonding txadd-timer** command to specify the value (in seconds) for the aggregate call connect timeout. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the number of seconds the endpoint will wait for additional channels (to add to the bonded aggregate) before considering the bonding negotiation a failure.

Default Values

By default, the **bonding txadd-timer** value is set to 50 seconds.

Functional Notes

Specifies the length of time both endpoints wait for additional calls to be connected at the end of negotiation before deciding that the bonding call has failed. The factory default setting is sufficient for most calls to connect, although when dialing overseas it may be necessary to lengthen this timer to allow for slower call routing.

Usage Examples

The following example defines a **bonding txadd-timer** value of 95 seconds:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#bonding txadd-timer 95
```

bonding txcid-timer <seconds>

Use the **bonding txcid-timer** command to specify the value (in seconds) for the bearer channel (B-channel) negotiation timeout. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the number of seconds the endpoint allots for negotiating data rates and channel capacities before considering the bonding negotiation a failure.

Default Values

By default, the **bonding txcid-timer** value is set to 5 seconds.

Functional Notes

Specifies the length of time both endpoints attempt to negotiate an agreeable value for bearer channels and channel capacities before deciding the bonding call has failed.

Usage Examples

The following example defines a **bonding txcid-timer** value of 8 seconds:

```
ProCurve(config)#interface bri 1/2  
(config-bri 1/2)#bonding txcid-timer 8
```


bonding txdeq-timer <seconds>

Use the **bonding txdeq-timer** command to specify the value (in seconds) for the network delay equalization timeout. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the number of seconds the endpoint allots for attempting to equalize the network delay between bearer channels before considering the bonding negotiation a failure.

Default Values

By default, the **bonding txdeq-timer** value is set to 50 seconds.

Functional Notes

Specifies the length of time both endpoints allot to attempt to equalize the network delay between the bearer channels before deciding the bonding call has failed.

Usage Examples

The following example defines a **bonding txdeq-timer** value of 80 seconds:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#bonding txdeq-timer 80
```

bonding txf-timer <seconds>

Use the **bonding txf-timer** command to specify the value (in seconds) for the frame pattern detection timeout. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the number of seconds the endpoint allots for attempting to detect the bonding frame pattern (when a call is connected) before considering the bonding negotiation a failure.

Default Values

By default, the **bonding txf-timer** value is set to 10 seconds.

Functional Notes

Specifies the length of time both endpoints attempt to detect the bonding frame pattern when a call is connected before deciding the bonding call has failed. When operating with other manufacturers' bonding equipment, it may be necessary to change this time so that it matches TXADD01.

Usage Examples

The following example defines a **bonding txf-timer** value of 15 seconds:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#bonding txf-timer 15
```

bonding txinit-timer <seconds>

Use the **bonding txinit-timer** command to specify the value (in seconds) for the originating endpoint negotiation timeout. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the number of seconds the endpoint waits to detect the bonding negotiation frame pattern from the remote endpoint (when a call is connected) before considering the bonding negotiation a failure.

Default Values

By default, the **bonding txinit-timer** value is set to 10 seconds.

Functional Notes

Specifies the length of time the originating endpoint attempts to detect the bonding negotiation pattern from the answering endpoint before deciding the bonding call has failed.

Usage Examples

The following example defines a **bonding txinit-timer** value of 15 seconds:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#bonding txinit-timer 15
```

bonding txnull-timer <seconds>

Use the **bonding txnull-timer** command to specify the value (in seconds) for the answering endpoint negotiation timeout. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the number of seconds the endpoint waits to detect the bonding negotiation frame pattern from the originating endpoint (after answering a call) before considering the bonding negotiation a failure.

Default Values

By default, the **bonding txnull-timer** value is set to 10 seconds.

Functional Notes

Specifies the length of time the answering endpoint attempts to detect the bonding negotiation pattern from the originating endpoint before deciding the bonding call has failed. It may be necessary to shorten this timer if the DTE equipment using the bonding module also has timer constraints for completing non-bonding parameter negotiation.

Usage Examples

The following example defines a **bonding txnull-timer** value of 8 seconds:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#bonding txnull-timer 8
```

caller-id override [always <number> | if-no-cid <number>]

Use the **caller-id override** command to configure the unit to replace caller ID information with a user-specified number. Use the **no** form of this command to disable any caller ID overrides.

Syntax Description

always <number>	Always forces replacement of the incoming caller ID number with the number given.
if-no-cid <number>	Replaces the incoming caller ID number with the number given only if there is no caller ID information available for the incoming call.

Default Values

By default, this command is disabled.

Functional Notes

Forces a replacement of the incoming caller ID number with the number given. The received caller ID, if any, is discarded, and the given override number is used to connect the incoming call to a circuit of the same number.

Usage Examples

The following example configures the unit to always provide the given number as the caller ID number:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#caller-id override always 5551000
```

isdn ldn1 <ldn>

Use the **isdn ldn1** command to specify the Local Directory Numbers (LDNs) associated with the Service Profile Identifier (SPID) entered using the **isdn spid1** command. Use the **no** form of this command to remove a configured LDN.

Note *The BRI module requires all incoming calls to be directed to the Local Directory Number (LDN) associated with the SPID programmed using the **isdn spid1** command. All calls to the LDN associated with SPID 2 will be rejected (unless part of a bonding call).*

Syntax Description

<ldn>	Specifies the LDN assigned to the circuit by the service provider. The LDN is the number used by remote callers to dial into the ISDN circuit. If the <ldn> field is left blank, the Secure Router OS will not accept inbound backup calls to the BRI module.

Default Values

By default, there are no configured LDNs.

Functional Notes

The BRI module requires all incoming calls to be directed to the LDN (programmed using the **isdn ldn1** command) associated with the SPID programmed using the **isdn spid1** command. All calls to the LDN associated with SPID 2 will be rejected (unless part of a bonding call).

Usage Examples

The following example defines an LDN of 555-1111:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#isdn ldn1 5551111
```

isdn ldn2 <ldn>

Use the **isdn ldn2** command to specify the Local Directory Numbers (LDNs) associated with the Service Profile Identifier (SPID) entered using the **isdn spid2** command. Use the **no** form of this command to remove a configured LDN.

Note *The BRI module requires all incoming calls to be directed to the Local Directory Number (LDN) associated with the SPID programmed using the **isdn spid1** command. All calls to the LDN associated with SPID 2 will be rejected (unless part of a bonding call).*

Syntax Description

<ldn>	Specifies the LDN assigned to the circuit by the service provider. The LDN is the number used by remote callers to dial into the ISDN circuit. If the <ldn> field is left blank, the Secure Router OS will not accept inbound backup calls to the BRI module.

Default Values

By default, there are no configured LDNs.

Functional Notes

The BRI module requires all incoming calls to be directed to the LDN (programmed using the **isdn ldn1** command) associated with the SPID programmed using the **isdn spid1** command. All calls to the LDN associated with SPID 2 will be rejected (unless part of a bonding call).

Usage Examples

The following example defines an LDN of 555-1112:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#isdn ldn2 5551112
```

isdn spid1 <spid>

Use the **isdn spid1** command to specify the Service Profile Identifiers (SPIDs). Use the **no** form of this command to remove a configured SPID.

Note *The BRI module requires all incoming calls to be directed to the Local Directory Number (LDN) associated with the SPID programmed using the **isdn spid1** command. All calls to the LDN associated with SPID 2 will be rejected (unless part of a bonding call).*

Syntax Description

<spid>	Specifies the 8 to 14 digit number identifying your Basic Rate ISDN (BRI) line in the Central Office switch. A SPID is generally created using the area code and phone number associated with the line and a four-digit suffix. For example, the following SPIDs may be provided on a BRI line with phone numbers 555-1111 and 555-1112:

SPID1: 701 555 1111 0101

SPID2: 701 555 1112 0101

Default Values

By default, there are no configured SPIDs.

Functional Notes

The Secure Router OS does not support "SPID-less" 5ESS signaling. SPIDs are required for all configured BRI endpoints using 5ESS signaling.

Usage Examples

The following example defines a SPID of 704 555 1111 010:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#isdn spid1 70455511110101
```


isdn spid2 <spid>

Use the **isdn spid2** command to specify the Service Profile Identifiers (SPIDs). Use the **no** form of this command to remove a configured SPID.

Note *The BRI module requires all incoming calls to be directed to the Local Directory Number (LDN) associated with the SPID programmed using the **isdn spid1** command. All calls to the LDN associated with SPID 2 will be rejected (unless part of a bonding call).*

Syntax Description

<spid>	Specifies the 8 to 14 digit number identifying your Basic Rate ISDN (BRI) line in the Central Office switch. A SPID is generally created using the area code and phone number associated with the line and a four-digit suffix. For example, the following SPIDs may be provided on a BRI line with phone numbers 555-1111 and 555-1112:

SPID1: 701 555 1111 0101

SPID2: 701 555 1112 0101

Default Values

By default, there are no configured SPIDs.

Functional Notes

The Secure Router OS does not support "SPID-less" 5ESS signaling. SPIDs are required for all configured BRI endpoints using 5ESS signaling.

Usage Examples

The following example defines a SPID of 704 555 1111 0101:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#isdn spid2 70455511110101
```

isdn switch-type [basic-5ess | basic-dms | basic-net3 | basic-ni]

Use the **isdn switch-type** command to specify the ISDN signaling type configured on the Basic Rate ISDN (BRI) interface. The type of ISDN signaling implemented on the BRI interface does not always match the manufacturer of the Central Office switch. Use the **no** form of this command to return to the default value.

Syntax Description

basic-5ess	Specifies Lucent/AT&T 5ESS signaling.
basic-dms	Specifies Nortel DMS-100 custom signaling. The basic-dms signaling type is not compatible with proprietary SL-1 DMS signaling.
basic-net3	Specifies Net3 Euro-ISDN signaling.
basic-ni	Specifies National ISDN-1 signaling.

Default Values

By default, the ISDN signaling type is set to National ISDN-1.

Functional Notes

The **isdn switch-type** command specifies the type of ISDN signaling implemented on the BRI interface, not the manufacturer of the Central Office switch. It is quite possible to have a Lucent Central Office switch providing National ISDN signaling on the BRI interface.

Usage Examples

The following example configures a BRI interface for a circuit with Lucent 5ESS (custom) signaling:

```
ProCurve(config)#interface bri 1/2
(config-bri 1/2)#isdn switch-type basic-5ess
```

FRAME RELAY INTERFACE CONFIG COMMAND SET

To activate the Frame Relay Interface Configuration mode, enter the **interface frame-relay** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface frame-relay 1
ProCurve(config-fr 1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

bandwidth <value> [on page 644](#)

encapsulation frame-relay ietf [on page 645](#)

fair-queue <threshold> [on page 646](#)

frame-relay commands [begin on page 647](#)

hold-queue <queue size> out [on page 659](#)

qos-policy out <mapname> [on page 660](#)

snmp trap [on page 661](#)

snmp trap link-status [on page 662](#)

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Specifies bandwidth in kbps.

Default Values

No default value is necessary for this command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the Frame Relay interface to 10 Mbps:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#bandwidth 10000
```

encapsulation frame-relay ietf

Use the **encapsulation frame-relay ietf** command to configure the encapsulation on a virtual Frame Relay interface as IETF (RFC 1490). Currently, this is the only encapsulation setting. Settings for this option must match the far-end router's settings in order for the Frame Relay interface to become active.

Syntax Description

No subcommands.

Default Values

By default, all Frame Relay interfaces use IETF encapsulation.

Usage Examples

The following example configures the endpoint for IETF encapsulation:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#encapsulation frame-relay ietf
```

fair-queue <threshold>

Use the **fair-queue** command to enable weighted fair queuing (WFQ) on an interface. Use the **no** form of this command to disable WFQ and enable FIFO (first-in-first-out) queueing for an interface. WFQ is enabled by default for WAN interfaces.

Syntax Description

<threshold>	Optional. Specifies the maximum number of packets that can be present in each conversation sub-queue. Packets received for a conversation after this limit is reached are discarded. Range: 16 to 512.

Default Values

By default, **fair-queue** is enabled with a threshold of 64 packets.

Usage Examples

The following example enables WFQ on the interface with a threshold set at 100 packets:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#fair-queue 100
```

frame-relay intf-type [dce | dte | nni]

Use the **frame-relay intf-type** command to define the Frame Relay signaling role needed for the endpoint. Use the **no** form of this command to return to the default value.

Syntax Description

dce	Specifies DCE or network signaling role. Use this interface type when you need the unit to emulate the frame switch.
dte	Specifies DTE or user signaling role. Use this interface type when connecting to a Frame Relay switch (or piece of equipment emulating a frame switch).
nni	Configures the interface to support both network and user signaling (DTE or DCE) when necessary.

Default Values

By default, **frame-relay intf-type** is set to **dte**.

Usage Examples

The following example configures the Frame Relay endpoint for DCE signaling:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay intf-type dce
```

frame-relay lmi-n391dce <polls>

Use the **frame-relay lmi-n391dce** command to set the N391 full status polling counter for the DCE endpoint. Typical applications should leave the default value for this timer. Use the **no** form of this command to return to the default value.

Syntax Description

<polls>	Sets the counter value (Range: 1 to 255).

Default Values

By default, the polling counter for the DCE endpoint is set to six polls.

Functional Notes

The N391 counter determines how many link integrity polls occur in between full status polls. The number of link integrity polls between full status polls is $n - 1$, where n represents the full status poll. n can be set to any number between 1 and 255, but the default is used for most applications.

Usage Examples

The following example sets the N391 counter for three polls:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-n391dce 3
```


frame-relay lmi-n391dte <polls>

Use the **frame-relay lmi-n391dte** command to set the n391 full status polling counter for the DTE endpoint. Typical applications should leave the default value for this timer. Use the **no** form of this command to return to the default value.

Syntax Description

<polls>	Sets the counter value (Range: 1 to 255).

Default Values

By default, the polling counter for the DTE endpoint is set to six polls.

Functional Notes

The N391 counter determines how many link integrity polls occur in between full status polls. The number of link integrity polls between full status polls is $n - 1$, where n represents the full status poll. n can be set to any number between 1 and 255, but the default is used for most applications.

Usage Examples

The following example sets the N391 counter for three polls:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-n391dte 3
```

frame-relay lmi-n392dce <threshold>

Use the **frame-relay lmi-n392dce** command to set the N392 error threshold for the DCE endpoint. Typical applications should leave the default value for this setting. Use the **no** form of this command to return to the default value.

Syntax Description

<threshold>	Sets the threshold value (Range: 1 to 10).

Default Values

By default, the error threshold for the DCE endpoint is set to three errors.

Functional Notes

If the error threshold is met, the signaling state status is changed to down, indicating a service-affecting condition. This condition is cleared once N393 consecutive error-free events are received. N392 defines the number of errors required in a given event window, while N393 defines the number of polling events in each window.

For example:

If N392=3 and N393=4, then if three errors occur within any four events, the interface is determined inactive.

Usage Examples

The following example sets the N392 threshold for 5 seconds:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-n392dce 5
```

frame-relay lmi-n392dte <threshold>

Use the **frame-relay lmi-n392dte** command to set the N392 error threshold for the DTE endpoint. Typical applications should leave the default value for this setting. Use the **no** form of this command to return to the default value.

Syntax Description

<threshold>	Sets the threshold value (Range: 1 to 10).

Default Values

By default, the error threshold for the DTE endpoint is set to three errors.

Functional Notes

If the error threshold is met, the signaling state status is changed to down, indicating a service-affecting condition. This condition is cleared once N393 consecutive error-free events are received. N392 defines the number of errors required in a given event window, while N393 defines the number of polling events in each window.

For example:

If N392=3 and N393=4, then if three errors occur within any four events, the interface is determined inactive.

Usage Examples

The following example sets the N392 threshold for five errors:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-n392dte 5
```

frame-relay lmi-n393dce <counter>

Use the **frame-relay lmi-n393dce** to set the N393 LMI monitored event counter for the DCE endpoint. Typical applications should leave the default value for this counter. Use the **no** form of this command to return to the default value.

Syntax Description

<counter>	Sets the counter value (Range: 1 to 10).

Default Values

By default, the LMI monitored event counter for the DCE endpoint is set to four events.

Usage Examples

The following example sets the N393 threshold for five events:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-n393dce 5
```

frame-relay lmi-n393dte *<counter>*

Use the **frame-relay lmi-n393dte** command to set the N393 LMI monitored event counter for the DTE endpoint. Typical applications should leave the default value for this counter. Use the **no** form of this command to return to the default value.

Syntax Description

<i><counter></i>	Sets the counter value (Range: 1 to 10).

Default Values

By default, the LMI monitored event counter for the DTE endpoint is set to four events.

Usage Examples

The following example sets the N393 threshold for five events:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-n393dte 5
```

frame-relay lmi-t391dte <seconds>

Use the **frame-relay lmi-t391dte** command to set the T391 signal polling timer for the DTE endpoint. Typical applications should leave the default value for this timer. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Sets the timer value in seconds (Range: 5 to 30).

Default Values

By default, the signal polling timer for the DTE endpoint is set to 10 seconds.

Functional Notes

The T391 timer sets the time (in seconds) between polls to the Frame Relay network.

Usage Examples

The following example sets the T391 timer for 15 seconds:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-t391dte 15
```

frame-relay lmi-t392dce <seconds>

Use the **frame-relay lmi-t392dce** command to set the T392 polling verification timer for the DCE endpoint. Typical applications should leave the default value for this timer. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Sets the timer value in seconds (Range: 5 to 30).

Default Values

By default, the polling verification timer for the DCE endpoint is set to 10 seconds.

Functional Notes

The T392 sets the timeout (in seconds) between polling intervals. This parameter needs to be a few seconds longer than the T391 setting of the attached Frame Relay device.

Usage Examples

The following example sets the T392 timer for 15 seconds:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-t392dce 15
```

frame-relay lmi-type [ansi | auto | cisco | none | q933a]

Use the **frame-relay lmi-type** command to define the Frame Relay signaling (LMI) type. Use the **no** form of the command to return to the default value.

Syntax Description

ansi	Specifies Annex D signaling method (based on ANSI T1.617 standard for Frame Relay).
auto	Automatically determines signaling type by messages received on the frame circuit.
cisco	Specifies Cisco LMI signaling method (reserves DLCI 1023).
none	Turns off signaling on the endpoint. This is used for backup connections.
q933a	Specifies Annex A signaling method (based on the ITU-T Q.933A frame format for Frame Relay).

Default Values

By default, the Frame Relay signaling type is set to **ansi**.

Usage Examples

The following example sets the signaling method for the endpoint to **cisco**:

```
ProCurve(config)#interface frame-relay 1
ProCurve(config-fr 1)#frame-relay lmi-type cisco
```


frame-relay multilink [ack <seconds> | bandwidth-class <class> <threshold> | hello <seconds> | retry <number>]

Use the **frame-relay multilink** command to enable the Frame Relay multilink interface. When the **no** form of this command is issued, all configuration options associated with this command and binds made to this interface are removed.

Syntax Description

ack <seconds>	Optional. Specifies a wait for acknowledgement time (in seconds) for every bundle link in the bundle. Range: 1 to 180 seconds.
bandwidth-class	Optional. Specifies the class of operation, placing a minimum limit on the acceptable amount of bandwidth required for a bundle to up.
<class>	Optional. Specifies the class of operation. Range: a to c. Class A A single active link is sufficient for the bundle to be up. Class B All defined bundle links must be active for the bundle to be up. Class C A minimum threshold of links must be active for the bundle to be up.
<threshold>	Optional. Specifies the minimum number of active bundle links required for a class C bundle to be in the up state. This option will not be available unless Class C is specified. Range: 1 to 65,535 links.
hello <seconds>	Optional. Specifies the time (in seconds) between hello messages for every bundle link in the bundle. Range: 1 to 180 seconds.
retry <number>	Optional. Specifies the number of times a bundle link will retransmit a message while waiting for acknowledgement. Range: 1 to 5 times.

Default Values

The default **ack** value is 4 seconds. The default **hello** value is 10 seconds. The default <class> value is a. The default **retry** value is 2.

Functional Note

This command is different from **ppp multilink**. In **ppp multilink**, if multiple binds are configured for the PPP interface without multilink PPP being enabled, the first link to bring up LCP will be the only link to actually bind. In Frame Relay multilink, since there is no protocol corresponding to LCP, all binds will be removed and the user will be free to re-issue any bind.

Usage Examples

The following example enables the Frame Relay multilink interface and sets the time between **hello** messages to 45 seconds:

```
ProCurve(config)#interface frame-relay 1
ProCurve(config-fr 1)#frame-relay multilink hello 45
```

The following example specifies Class B operation:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay multilink bandwidth-class b
```

The following example specifies Class C operation with a threshold of 5:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay multilink bandwidth-class c 5
```

hold-queue <queue size> out

Use the **hold-queue out** command to change the overall size of an interface's WAN output queue.

Syntax Description

<queue size>	The total number of packets the output queue can contain before packets are dropped. Range: 16 to 1000.

Default Values

The default queue size for WFQ is 400. The default queue size for PPP FIFO and Frame Relay round robin is 200.

Usage Examples

The following example sets the overall output queue size to 700:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#hold-queue 700 out
```

qos-policy out <mapname>

Use the **qos-policy out** command to apply a previously-configured QoS map to an interface. Use the **no** form of this command to remove the map from the interface. The **out** keyword specifies that this policy will be applied to outgoing packets.

Syntax Description

<map name>	Specifies the name of a previously-created QoS map (see <i>qos map</i> <mapname> <sequence number> on page 363 for more information).

Default Values

No default value is necessary for this command.

Usage Examples

The following example applies the QoS map **VOICEMAP** to the Frame Relay 1 interface:

```
ProCurve(config)#interface frame-relay 1
ProCurve(config-fr 1)#qos-policy out VOICEMAP
```

snmp trap

Use the **snmp trap** command to enable all supported Simple Network Management Protocol (SNMP) traps on the interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces (except virtual Frame Relay interfaces and sub-interfaces) have SNMP traps enabled.

Usage Examples

The following example enables SNMP on the virtual Frame Relay interface:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#snmp trap
```

snmp trap link-status

Use the **snmp trap link-status** command to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the Frame Relay interface:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#no snmp trap link-status
```

FRAME RELAY SUB-INTERFACE CONFIG COMMAND SET

To activate the Frame Relay Interface Configuration mode, enter the **interface frame-relay** command (and specify a sub-interface) at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)
shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycyname> [on page 664](#)
backup commands [begin on page 667](#)
bandwidth <value> [on page 683](#)
bridge-group <group#> [on page 684](#)
crypto map <mapname> [on page 685](#)
dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password> [on page 687](#)
frame-relay bc <committed burst value> [on page 689](#)
frame-relay be <excessive burst value> [on page 690](#)
frame-relay fragment <threshold> [on page 691](#)
frame-relay interface-dlci <dlci> [on page 692](#)
ip commands [begin on page 693](#)
lldp receive [on page 712](#)
lldp send [management-address l port-description l system-capabilities l system-description l system-name] [on page 713](#)
lldp send-and-receive [on page 714](#)
mtu <size> [on page 715](#)
spanning-tree commands [begin on page 716](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic on an interface. Use the **no** form of this command to remove an access policy association.

Note *Configured access policies will only be active if the **ip firewall** command has been entered at the Global Configuration mode prompt to enable the Secure Router OS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.*

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor (all access policy descriptors are case-sensitive).

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access-policy <policy name>**.

Usage Examples

The following example associates the access policy **UnTrusted** (to allow inbound traffic to the Web server) to the Frame Relay sub-interface labeled 1.16:

Enable the Secure Router OS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list **InWeb**:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access list with the Frame Relay sub-interface labeled 1.16:

```
ProCurve(config)#interface frame-relay 1.16
```

```
ProCurve(config-fr 1.16)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an access policy that uses a configured access list. Secure Router OS access policies are used to permit, deny, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list <access list names> **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet and the Web GUI.

discard list <access list names> self

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list <access list names> address <IP address> overload

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list <access list names> interface <interface> overload

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list <access list names> address <IP address>

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**. The following example assigns access policy **MatchAll** to the Frame Relay sub-interface labeled 1.16:

```
ProCurve(config)#interface frame-relay 1.16
```

```
ProCurve(config-fr 1.16)#access-policy MatchAll
```

backup auto-backup

Use the **backup auto-backup** command to configure the sub-interface to automatically attempt a backup upon failure.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically attempt backup upon a failure.

Usage Examples

The following enables automatic backup on the endpoint:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup auto-backup
```

backup auto-restore

Use the **backup auto-restore** command to configure the sub-interface to automatically discontinue backup when all network conditions are operational. Use the **no** form of this command to disable the auto-restore feature.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically restore the primary connection when the failure condition clears.

Usage Examples

The following configures the Secure Router OS to automatically restore the primary connection when the failure condition clears:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup auto-restore
```

backup backup-delay <seconds>

Use the **backup backup-delay** command to configure the amount of time the router will wait after the failure condition is recognized before attempting to backup the link. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the delay period (in seconds) a failure must be active before the Secure Router OS will enter backup operation on the interface (valid range: 10 to 86400 seconds).

Default Values

By default, the **backup backup-delay** is set to 10 seconds.

Usage Examples

The following configures the Secure Router OS to wait 60 seconds (on an endpoint with an active alarm condition) before attempting backup operation:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup backup-delay 60
```

backup call-mode <role>

Use the **backup call-mode** command to specify whether the configured backup interfaces answers or originates (or a combination of both) backup calls. Use the **no** form of this command to return to the default value.

Syntax Description

<role>	Selects the role the router will take in backup of this sub-interface.
answer	Answers and backs up primary link on failure.
answer-always	Answers and backs up regardless of primary link state.
originate	Originates backup call on primary link failure.
originate-answer	Originates or answers call on primary link failure.
originate-answer-always	Originates on failure; answers and backs up always.

Default Values

By default, the **backup call-mode** is set to **originate-answer**.

Functional Notes

The majority of the configuration for Frame Relay backup is configured in the Frame Relay sub-interface configuration commands. However, the numbers dialed are configured in the primary interface. Full sample configurations follow:

Sample config for remote router (dialing out)

```
hostname "Remote7203dl"
enable password password
!
interface eth 0/1
 ip address 192.168.1.254 255.255.255.0
 no shutdown
!
interface modem 1/3
 no shutdown
!
interface t1 1/1
 coding b8zs
 framing esf
 clock source line
 tdm-group 1 timeslots 1-24
 no shutdown
!
```

```
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  no shutdown
  bind 1 t1 1/1 1 fr 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  ip address 10.1.1.2 255.255.255.252
  backup call-mode originate
  backup number 5551111 analog
  backup number 5552222 analog
!
ip route 0.0.0.0 0.0.0.0 10.1.1.1
!
line telnet 0 4
  password password
```

Sample config for central router (dialing in)

```
hostname "Central7203dl"
enable password password
!
interface eth 0/1
  ip address 192.168.100.254 255.255.255.0
  no shutdown
!
interface modem 1/3
  no shutdown
!
interface t1 1/1
  coding b8zs
  framing esf
  clock source line
  tdm-group 1 timeslots 1-24
  no shutdown
!
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  no shutdown
  bind 1 t1 1/1 1 fr 1
!
interface fr 1.100 point-to-point
```

```
frame-relay interface-dlci 100
ip address 10.1.1.1 255.255.255.252
backup call-mode answer
backup number 555-8888 analog
!
line telnet 0 4
password password
```

Usage Examples

The following configures the Secure Router OS to answer backup calls on this endpoint but never generate calls:

```
ProCurve(config)#interface frame-relay 1.1
ProCurve(config-fr 1.1)#backup call-mode answer-always
```


backup connect-timeout <seconds>

Use the **backup connect-timeout** command to specify the number of seconds to wait for a connection after a call is attempted before trying to call again or dialing a different number. It is recommended this number be greater than 60.

Syntax Description

<seconds>	Selects the amount of time in seconds that the router will wait for a connection before attempting another call (valid range: 10 to 300).

Default Values

By default, **backup connect-timeout** is set to 60 seconds.

Usage Examples

The following configures the Secure Router OS to wait 120 seconds before retrying a failed backup call:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup connect-timeout 120
```

backup force <state>

Use the **backup force** command to manually override the automatic backup feature. This can be used to force a link into backup to allow maintenance to be performed on the primary link without disrupting data. Use the **no** form of this command to return to the normal backup operation state.

Syntax Description

<state>	Selects the forced backup state of the sub-link.
backup	Forces backup regardless of primary link state.
primary	Forces primary link regardless of its state.

Default Values

By default, this feature is disabled.

Usage Examples

The following configures the Secure Router OS to force this endpoint into backup:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup force backup
```

backup maximum-retry <attempts>

Use the **backup maximum-retry** command to select the number of calls the router will make when attempting to backup a link. Use the **no** form of this command to return to the default state. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes and Technology Review* sections of the command *backup call-mode <role>* [on page 670](#).

Syntax Description

<attempts>	Selects the number of call retries that will be made after a link failure (valid range: 0 to 15).

Setting this value to 0 will allow unlimited retries during the time the network is failed.

Default Values

By default, **backup maximum-retry** is set to 0 attempts.

Usage Examples

The following example configures the Secure Router OS to retry a backup call four times before considering backup operation not available:

```
ProCurve(config)#interface frame-relay 1.1
ProCurve(config-fr 1.1)#backup maximum-retry 4
```

backup number <digits> [**analog** | **digital-56k** | **digital 64k**] <isdn min chan> <isdn max chan> <interface>

Use the **backup number** command to configure the phone number and the call type the router will dial upon network failure. Multiple entries can be made for an interface to allow alternate sites to be dialed. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode* <role> [on page 670](#).

Syntax Description

<digits>	Specifies the phone numbers to call when the backup is initiated.
analog	Indicates number connects to an analog modem.
digital-56k	Indicates number belongs to a digital 56 kbps per DS0 connection.
digital-64k	Indicates number belongs to a digital 64 kbps per DS0 connection.
<isdn min chan>	Specifies the minimum number of DS0s required for a digital 56 or 64 kbps connection (Range: 1 to 24).
<isdn mas chan>	Specifies the maximum number of DS0s desired for a digital 56 or 64 kbps connection (Range: 1 to 24).
<interface>	Specifies the Frame Relay sub-interface (e.g., fr 1.16) to use when originating or answering using this number.

Default Values

By default, there are no configured **backup numbers**.

Usage Examples

The following example configures the Secure Router OS to dial 704-555-1212 (digital 64 kbps connection) to initiate backup operation on this endpoint using the Frame Relay sub-interface fr 1.1:

```
ProCurve(config)#interface frame-relay 1.1
```

```
ProCurve(config-fr 1.1)#backup number 7045551212 digital-64k 1 1 frame-relay 1.1
```

backup priority <value>

Use the **backup priority** command to select the backup priority for this interface. This allows the user to establish the highest priority backup link and ensure that link will override backups attempted by lower priority links. Use the **no** form of this command to return to the default value. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 670](#).

Syntax Description

<value>	Sets the relative priority of this link (valid range: 0 to 100). A value of 100 designates the highest priority.

Default Values

By default, **backup priority** is set to 50.

Usage Examples

The following example assigns the highest priority to this endpoint:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#backup priority 100
```

backup randomize-timers

Use the **backup randomize-timers** command to randomize the call timers to minimize potential contention for resources. Use the **no** form of this command to return to the default value. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 670](#).

Syntax Description

No subcommands.

Default Values

By default, the Secure Router OS does not randomize the **backup call timers**.

Usage Examples

The following example configures the Secure Router OS to randomize the backup timers associated with this endpoint:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup randomize-timers
```

backup redial-delay <seconds>

Use the **backup redial-delay** command to configure the delay after an unsuccessful call until the call will be re-tried. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 670](#).

Syntax Description

<seconds>	Specifies the delay in seconds between attempting to re-dial a failed backup attempt. Range: 10 to 3600.

Default Values

By default, **backup redial-delay** is set to 10 seconds.

Usage Examples

The following example configures a redial delay of 25 seconds on this endpoint:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup redial-delay 25
```

backup restore-delay <seconds>

Use the **backup restore-delay** command to configure the amount of time the router will wait after the network is restored before disconnecting the backup link and reverting to the primary. This setting is used to prevent disconnecting the backup link if the primary link is “bouncing” in and out of alarm. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 670](#).

Syntax Description

<seconds>	Specifies the number of seconds the Secure Router OS will wait (after a primary link is restored) before disconnecting backup operation. Range: 10 to 86,400.

Default Values

By default, **backup restore-delay** is set to 10 seconds.

Usage Examples

The following example configures the Secure Router OS to wait 30 seconds before disconnecting backup operation and restoring the primary connection for this endpoint:

```
ProCurve(config)#interface frame-relay 1.1  
ProCurve(config-fr 1.1)#backup restore-delay 30
```


backup schedule [day | enable-time | disable-time]

Use the **backup schedule** command to set the time of day that backup will be enabled. Use this command if backup is desired only during normal business hours and on specific days of the week. Use the **no** form of this command to disable backup (as specified). For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 670](#).

Syntax Description

day	Sets the days to allow backup (valid range: Monday through Sunday).
enable-time	Sets the time of day to enable backup. Time is entered in 24-hour format (00:00).
disable-time	Sets the time of day to disable backup.

Default Values

By default, backup is enabled for all days and times if the **backup auto-backup** command has been issued and the **backup schedule** has not been entered.

Usage Examples

The following example enables backup Monday through Friday 8:00 am to 7:00 pm:

```
ProCurve(config)#interface frame-relay 1.1
ProCurve(config-fr 1.1)#backup schedule enable-time 08:00
ProCurve(config-fr 1.1)#backup schedule disable-time 19:00
ProCurve(config-fr 1.1)#no backup schedule day Saturday
ProCurve(config-fr 1.1)#no backup schedule day Sunday
```

backup shutdown

Use the **backup shutdown** command to deactivate all backup functionality in the unit. Backup configuration parameters are kept intact, but the unit will not initiate (or respond) to backup sequences in the event of a network outage. Use the **no** form of this command to reactivate the backup interface. For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command.

Syntax Description

No subcommands.

Default Values

By default, all Secure Router OS interfaces are disabled.

Usage Examples

The following example deactivates the configured backup interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#backup shutdown
```

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Specifies bandwidth in kbps.

Default Values

To view default values use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the Frame Relay interface to 10 Mbps:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#bandwidth 10000
```

bridge-group <group#>

Use the **bridge-group** command to assign an interface to the specified bridge group. This command is supported on all Ethernet interfaces, PPP virtual interfaces, and Frame Relay virtual sub-interfaces. Use the **no** form of this command to remove the interface from the bridge group.

Syntax Description

<group#>	Specifies the bridge group number (1 to 255).

Default Values

By default, there are no configured bridge groups.

Functional Notes

A bridged network can provide excellent traffic management to reduce collisions and limit the amount of bandwidth wasted with unnecessary transmissions when routing is not necessary. Any two interfaces can be bridged (Ethernet to T1 bridge, Ethernet to Frame Relay sub-interface).

Usage Examples

The following example assigns the Frame Relay sub-interface labeled 1.16 to bridge group 1:

```
ProCurve(config)#interface frame-relay 1.16
```

```
ProCurve(config-fr 1.16)#bridge-group 1
```

crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<mapname> Specifies the crypto map name that you wish to assign to the interface.

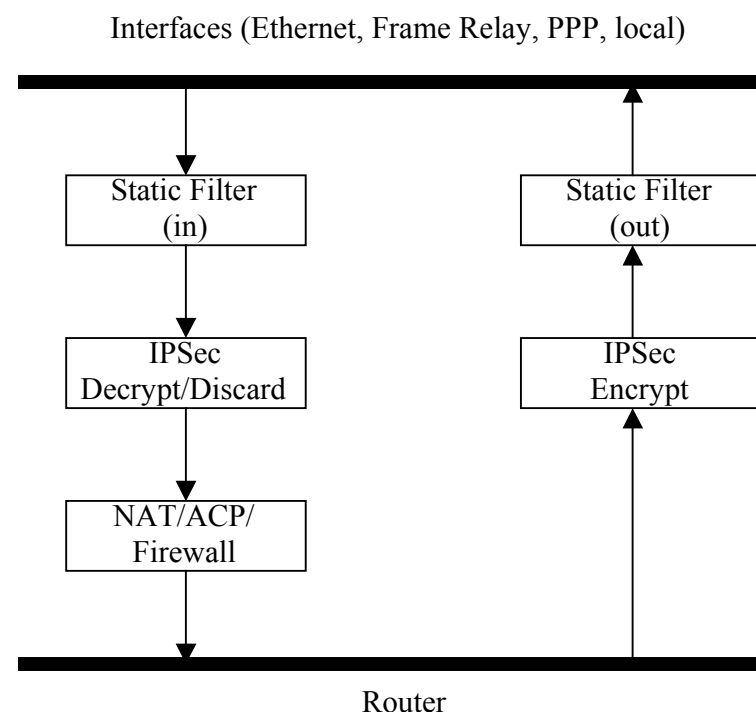
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following notes in mind.

When defining the policy-class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the unencrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical Secure Router OS data-flow logic.



As shown in the diagram above, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only unencrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy-class on an interface. When specifying the ACLs for a crypto map, the source information is the private local side, unencrypted source of the data. The destination information will be the far end, unencrypted destination of the data. However, ACLs for a policy-class work in reverse. The source information for the ACL in a policy class is the far end. The destination information is the local side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the Frame Relay interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#crypto map MyMap
```

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password>

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

See **Functional Notes** below for argument descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A web-based interface provides two levels of control over your domain, catering to average or power users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs, and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service, in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

When to Use Dynamic DNS

If your IP address does not change often (or at all) but you still want an easy name to remember it by (without having to purchase your own domain name), Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com), you need Custom DNS service, which also provides full dynamic and static IP address support.

Dynamic DNS service can be extremely helpful for site-to-site VPN connections using dynamic IP addressing. Instead of creating a tunnel to the IP-address, a tunnel to the DNS-name is established.

Usage Examples

The following example sets the Dynamic DNS to **dyndns custom** with hostname host, username user, and password **pass**:

```
ProCurve(config)#interface frame-relay 1.16
```

```
ProCurve(config-fr 1.16)#dynamic-dns dyndns-custom host user pass
```


frame-relay bc *<committed burst value>*

Use the **frame-relay bc** command to set the b_c (committed burst) value for a Frame Relay sublink. The value is in bits. Use the **no** form of this command to return to default.

Syntax Description

<committed burst value> Enter the committed burst value (in bits) for the sublink.

Default Values

The default is 0 (no limit).

Functional Notes

The time interval is always one second, so this can also be considered bits per second. Shaping is performed on a sliding one-second window to make maximum use of configured bandwidth. Note that when both b_c and b_e are non-zero, shaping is performed on the virtual circuit. The circuit is limited to the sum of b_c and b_e , and it is recommended that the sum always be greater than 8000.

Usage Examples

The following example configures sublink fr 1.16 with a committed burst value of 128000 bits:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#frame-relay bc 128000
```

frame-relay be *<excessive burst value>*

Use the **frame-relay be** command to set the b_e (excessive burst) value for a Frame Relay sublink. The value is in bits. Use the **no** form of this command to return to default.

Syntax Description

<i><excessive burst value></i>	Specifies the excessive burst value (in bits) for the sublink.

Default Values

The default is 0 (no limit).

Functional Notes

The time interval is always one second, so this can also be considered bits per second. Shaping is performed on a sliding one-second window to make maximum use of configured bandwidth. Note that when both b_c and b_e are non-zero, shaping is performed on the virtual circuit. The circuit is limited to the sum of b_c and b_e , and it is recommended that the sum always be greater than 8000.

Usage Examples

The following example configures the sublink fr 1.16 with an excessive burst value of 64000 bits:

```
ProCurve(config)#interface frame relay 1.16  
ProCurve(config-fr 1.16)#frame-relay be 64000
```

frame-relay fragment <threshold>

Use the **frame-relay fragment** command to set the FRF.12 fragmentation threshold. Use the **no** form of this command to erase the configured threshold.

Syntax Description

<threshold>	Specifies the fragmentation threshold. Valid fragmentation thresholds are greater than or equal to 64 and less than or equal to 1600.

Default Values

No default value is necessary for this command.

Functional Notes

For Frame Relay fragmentation to take effect, rate-limiting must be enabled by setting the committed burst rate and excessive burst rate. See *frame-relay bc <committed burst value>* [on page 689](#) and *frame-relay be <excessive burst value>* [on page 690](#) for more information.

Usage Examples

The following example enables FRF.12 fragmentation on a sublink:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#frame-relay bc 64000
ProCurve(config-fr 1.16)#frame-relay be 16
ProCurve(config-fr 1.16)#frame-relay fragmentation 100
```

The following example disables FRF.12 fragmentation on a sublink:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#no frame-relay fragment
```

frame-relay interface-dlci <dlci>

Use the **frame-relay interface-dlci** command to configure the Data Link Connection Identifier (DLCI) for the Frame Relay sub-interface. This setting should match the DLCI supplied by your Frame Relay service provider. Use the **no** form of this command to remove the configured DLCI.

Syntax Description

<dlci>	Specifies numeric value supplied by your provider.

Default Values

By default, the DLCI is populated with the sub-interface identifier. For example, if configuring the virtual Frame Relay sub-interface labeled **fr 1.20**, the default DLCI is **20**.

Usage Examples

The following example configures a DLCI of 72 for this Frame Relay endpoint:

```
ProCurve(config)#interface frame relay 1.16  
ProCurve(config-fr 1.16)#frame-relay interface-dlci 72
```

ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<listname>	Specifies the IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the router to only allow Telnet traffic into the Frame Relay sub-interface:

```
ProCurve(config)#interface frame relay 1.16
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#int frame-relay 1.16
ProCurve(config-fr 1.16)#ip access-group TelnetOnly in
```

ip address dhcp

Use the **ip address dhcp** command to use Dynamic Host Configuration Protocol (DHCP) to obtain an address on the interface. Use the **no** form of this command to remove a configured IP address (using DHCP) and disable DHCP operation on the interface.

ip address dhcp [client-id [*<interface>* | *<identifier>*] hostname "*<string>*"]

Syntax Description

client-id	Optional. Specifies the client identifier used when obtaining an IP address from a DHCP server.
<i><interface></i>	Specifying an interface defines the client identifier as the hexadecimal MAC address of the specified interface (including a hexadecimal number added to the front of the MAC address to identify the media type). For example, specifying the client-id ethernet 0/1 (where the Ethernet interface has a MAC address of d217.0491.1150) defines the client identifier as 01:d2:17:04:91:11:50 (where 01 defines the media type as Ethernet). Refer to <i>hardware-address <hardware-address> <type></i> on page 283 for a detailed listing of media types.
<i><identifier></i>	Specifies a custom client-identifier using a text string (that is converted to a hexadecimal equivalent) or 7 to 28 hexadecimal numbers (with colon delimiters). For example, a custom client identifier of 0f:ff:ff:ff:51:04:99:a1 may be entered using the <i><identifier></i> option.
host-name	Optional. Specifies a text string (to override the global router name) to use as the name in the DHCP option 12 field.
<i>"<string>"</i>	String (encased in quotation marks) of up to 35 characters to use as the name of the host for DHCP operation.
no-default-route	Keyword used to specify that the OS not install the default-route obtained via DHCP.
no-domain-name	Keyword used to specify that the OS not install the domain-name obtained via DHCP.
no-nameservers	Keyword used to specify that the OS not install the DNS servers obtained via DHCP.

Default Values

client-id Optional. By default, the client identifier is populated using the following formula:

TYPE: INTERFACE SPECIFIC INFO : MAC ADDRESS

Where TYPE specifies the media type in the form of one hexadecimal byte (refer to *hardware-address* <*hardware-address*> <*type*> [on page 283](#) for a detailed listing of media types), and the MAC ADDRESS is the Media Access Control (MAC) address assigned to the first Ethernet interface in the unit in the form of six hexadecimal bytes. (For units with a single Ethernet interface, the MAC ADDRESS assigned to ethernet 0/1 is used in this field).

INTERFACE SPECIFIC INFO is only used for Frame Relay interfaces and can be determined using the following:

FR_PORT# : Q.922 ADDRESS

Where the FR_PORT# specifies the label assigned to the virtual Frame Relay interface using four hexadecimal bytes. For example, a virtual Frame Relay interface labeled 1 would have a FR_PORT# of 00:00:00:01.

The Q.922 ADDRESS field is populated using the following:

8	7	6	5	4	3	2	1
DLCI (high order)						C/R	EA
DLCI (lower)		FECN		BECN	DE	EA	

Where the FECN, BECN, C/R, DE, and high order EA bits are assumed to be 0 and the lower order extended address (EA) bit is set to 1.

The following list provides a few example DLCIs and associated Q.922 addresses: DLCI (decimal) / Q.922 address (hex)

16 / 0x0401
 50 / 0x0C21
 60 / 0x0CC1
 70 / 0x1061
 80 / 0x1401

hostname Optional. By default, the hostname is the name configured using the Global Configuration **hostname** command.

<"string"> By default, the hostname is the name configured using the Global Configuration **hostname** command.

Functional Notes

DHCP allows interfaces to acquire a dynamically assigned IP address from a configured DHCP server on the network. Many Internet Service Providers (ISPs) require the use of DHCP when connecting to their services. Using DHCP reduces the number of dedicated IP addresses the ISP must obtain. Consult your ISP to determine the proper values for the **client-id** and **hostname** fields.

Usage Examples

The following example enables DHCP operation on the virtual Frame Relay interface (labeled 1.16):

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip address dhcp
```


ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface. Use the optional **secondary** keyword to define a secondary IP address. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional keyword used to configure a secondary IP address for the specified interface.

Default Values

By default, there are no assigned IP addresses.

Functional Notes

Use secondary IP addresses to allow dual subnets on a single interface (when you need more IP addresses than the primary subnet can provide). When using secondary IP addresses, avoid routing loops by verifying that all devices on the network segment are configured with secondary IP addresses on the secondary subnet.

Usage Examples

The following example configures a secondary IP address of **192.22.72.101/30**:

```
(ProCurve(config)#interface frame-relay 1.16
```

```
ProCurve(config-fr 1.16)#ip address 192.22.72.101 255.255.255.252 secondary
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip dhcp [release | renew]

Use the **ip dhcp** command to release or renew the DHCP IP address. This command is only applicable when using DHCP for IP address assignment.

Syntax Description

release	Releases DHCP IP address.
renew	Renews DHCP IP address.

Default Values

No default values required for this command.

Usage Examples

The following example releases the IP DHCP address for the virtual interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip dhcp release
```

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note *The **ip helper** command must be used in conjunction with the **ip forward-protocol** command to configure the Secure Router OS to forward UDP broadcast packets. Refer to [ip forward-protocol udp <port number>](#) on page 319 for more information.*

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign a helper-address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is User Datagram Protocol (UDP).
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip helper-address 192.33.5.99
```

ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

immediate-leave	If only one host (or IGMP snooping switch) is connected to the interface, when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	Controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer, if no receiver responds, the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65535 ms. Default: 1000 ms.
querier-timeout <seconds>	Specifies the number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60 to 300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Specifies the maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically-connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip igmp last-member-query-interval 200
```

ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface and place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding. Refer to *ip mcast-stub helper-address <ip address>* [on page 327](#) and *ip mcast-stub upstream* [on page 703](#) for more information.

Usage Examples

The following example enables multicast forwarding and IGMP on the interface:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip mcast-stub downstream
```

ip mcast-stub helper-enable

Use the **ip mcast-stub helper-enable** command to assign the **ip mcast-stub helper-address** as the IGMP proxy. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address**, **ip mcast-stub upstream**, and **ip mcast-stub downstream** commands. When enabled, the interface becomes a helper forwarding interface. The IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the unit to perform as an IGMP proxy. Refer to *ip mcast-stub helper-address <ip address>* [on page 389](#), *ip mcast-stub downstream* [on page 701](#), and *ip mcast-stub upstream* [on page 703](#) for more information.

Usage Examples

The following example sets the helper address as the IGMP proxy:

```
ProCurve(config)#interface frame relay 1.16  
ProCurve(config-fr 1.16)#ip mcast-stub helper-enable
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface. Refer to *ip mcast-stub helper-address <ip address>* [on page 327](#) and *ip mcast-stub downstream* [on page 701](#) for more information.

Usage Examples

The following example enables multicast forwarding on the interface:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip mcast-stub upstream
```

ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key <password>	Assigns a simple-text authentication password to be used by other routers using the OSPF simple password authentication.
cost <value>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65535.
dead-interval <seconds>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, neighboring devices will determine that the device is down. Range: 0 to 32767.
hello-interval <seconds>	Specifies the interval between hello packets sent on the interface. Range: 0 to 32767.
message-digest-key <keyid> md5 <key>	Configures OSPF Message Digest 5 (MD5) authentication (16-byte maximum) keys.
priority <value>	Sets the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <seconds>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32767.
transmit-delay <seconds>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32767.

Default Values

retransmit-interval <seconds> 5 seconds
transmit-delay <seconds> 1 second
hello-interval <seconds> <seconds><seconds> 40 seconds

Usage Example

The following example sets the maximum number of seconds allowed between hello packets to 25,000:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip ospf dead-interval 25000
```


ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Optional. Selects message-digest authentication type.
null	Optional. Specifies that no authentication be used.

Default Values

By default, this is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the Frame Relay interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip ospf authentication null
```

ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Sets the network type for broadcast.
point-to-point	Sets the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP and Frame Relay default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip ospf network broadcast
```

ip proxy-arp <address> <subnet mask>

Use the **ip proxy-arp** to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, proxy ARP is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the Secure Router OS will respond to all proxy ARP requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the Frame Relay sub-interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip proxy-arp
```

ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Accepts only received RIP version 1 packets on the interface.
2	Accepts only received RIP version 2 packets on the interface.

Default Values

By default, all interfaces implement RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip receive version** command to specify a RIP version that will override the **version** (in the Router RIP) configuration.

The Secure Router OS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures a Frame Relay sub-interface to accept only RIP version 2 packets:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip rip receive version 2
```

ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Transmits only RIP version 1 packets on the interface.
2	Transmits only RIP version 2 packets on the interface.

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** to specify a RIP version that will override the **version** (in the Router RIP) configuration.

The Secure Router OS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures a Frame Relay sub-interface to transmit only RIP version 2 packets:

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip rip send version 2
```

ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note *Using Network Address Translation (NAT) or the Secure Router OS firewall capabilities on an interface requires process switching mode (using the **no ip route-cache** command).*

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on a Frame Relay sub-interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#ip route-cache
```

ip unnumbered <interface>

Use the **ip unnumbered** command to use the IP address assigned to the specified interface for all IP processing on the active interface. Use the **no** form of this command to remove the unnumbered configuration.

Syntax Description

<interface>	Specifies the interface (in the format type slot/port) that contains the IP address to use as the source address for all packets transmitted on this interface.

Default Values

By default, all interfaces are configured to use a specified IP address (using the **ip address** command).

Functional Notes

If **ip unnumbered** is enabled on an interface, all IP traffic from the interface will use a source IP address taken from the specified interface. For example, specifying **ip unnumbered eth 0/1** while in the Frame Relay Sub-Interface Configuration mode configures the Frame Relay sub-interface to use the IP address assigned to the Ethernet interface for all IP processing. In addition, the Secure Router OS uses the specified interface information when sending route updates over the unnumbered interface.

Usage Examples

The following example configures the Frame Relay interface (labeled **frame-relay 1.16**) to use the IP address assigned to the Ethernet interface (**eth 0/1**):

```
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#ip unnumbered eth 0/1
```

lldp receive

Use the **lldp receive** command to allow LLDP packets to be received on this interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to send and receive LLDP packets.

Usage Examples

The following example configures the Frame Relay sub-interface to receive LLDP packets:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#lldp receive
```


lldp send [management-address | port-description | system-capabilities | system-description | system-name]

Use the **lldp send** command to configure this interface to transmit LLDP packets or to control the types of information contained in the LLDP packets transmitted by this interface.

Syntax Description

management-address	Enables transmission of management address information on this interface.
port-description	Enables transmission of port description information on this interface.
system-capabilities	Enables transmission of this device's system capabilities on this interface.
system-description	Enables transmission of this device's system description on this interface.
system-name	Enables transmission of this device's system name on this interface.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the Frame Relay sub-interface to transmit LLDP packets containing all enabled information types:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#lldp send
```

lldp send-and-receive

Use the **lldp send-and-receive** command to configure this interface to transmit and receive LLDP packets.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the Frame Relay sub-interface (fr 1.16) to transmit and receive LLDP packets containing all information types:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#lldp send-and-receive
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit (MTU) size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size> Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:

ATM interfaces	64 to 1520
Ethernet interfaces	64 to 1500
FDL interfaces	64 to 256
HDLC interfaces	64 to 1520
Loopback interfaces	64 to 1500
Tunnel interfaces	64 to 18,190
Virtual Frame Relay sub-interfaces	64 to 1520
Virtual PPP interfaces	64 to 1500

Default Values

<size> The default values for the various interfaces are listed below:

ATM interfaces	1500
Ethernet interfaces	1500
FDL interfaces	256
HDLC interfaces	1500
Loopback interfaces	1500
Tunnel interfaces	1500
Virtual Frame Relay sub-interfaces	1500
Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the Frame Relay interface:

```
ProCurve(config)#interface frame relay 1.16
ProCurve(config-fr 1.16)#mtu 1200
```

spanning-tree bpdudfilter [enable | disable]

Use the **spanning-tree bpdudfilter** command to block BPDUs from being transmitted and received on this interface. To return to the default value, use the **no** form of this command.

Syntax Description

enable	Enables the BPDU filter.
disable	Disables the BPDU filter.

Default Values

By default, this command is set to disable.

Functional Notes

The purpose of this command is to remove a port from participation in the spanning tree. This might be beneficial while debugging a network setup. It normally should not be used in a live network.

Usage Examples

The following example enables the BPDU filter on the interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#spanning-tree bpdudfilter enable
```

spanning-tree bpduguard [enable | disable]

Use the **spanning-tree bpduguard** command to block BPDUs from being received on this interface. To return to the default value, use the **no** form of this command.

Syntax Description

enable	Enables the BPDU block.
disable	Disables the BPDU block.

Default Values

By default, this command is set to disable.

Usage Examples

The following example enables the BPDU guard on the interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#spanning-tree bpduguard enable
```

spanning-tree edgeport

Use the **spanning-tree edgeport** command to set this interface to be an edgeport. This command overrides the Global setting. Use the **no** form of this command to return the interface to normal operation (non-edgeport).

Syntax Description

No subcommands.

Default Values

By default, this command is set to disable.

Usage Examples

The following example configures the interface to be an edgeport:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#spanning-tree edgeport
```

An individual interface can be configured to not be considered an edgeport. For example:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#no spanning-tree edgeport
```

spanning-tree link-type [auto | point-to-point | shared]

Use the **spanning-tree link-type** command to configure the spanning-tree protocol link type for an interface. To return to the default value, use the **no** form of this command.

Syntax Description

auto	Link type is determined by the port's duplex settings.
point-to-point	Link type is manually set to point-to-point, regardless of duplex settings.
shared	Link type is manually set to shared, regardless of duplex settings.

Default Values

By default, a port is set to auto.

Functional Notes

This command overrides the default link-type setting determined by the duplex of the individual port. By default a port configured for half-duplex is set to **shared** link type, and a port configured for full-duplex is set to **point-to-point** link type. Setting the link type manually overrides the default and forces the port to use the specified link type. Using the **link-type auto** command, restores the convention of determining link type based on duplex settings.

Usage Examples

The following example forces the link type to point-to-point, even if the port is configured to be half-duplex:

```
ProCurve(config)#bridge 1 protocol ieee
ProCurve(config)#interface frame-relay 1.16
ProCurve(config-fr 1.16)#spanning-tree link-type point-to-point
```

Technology Review

Rapid transitions are possible in rapid spanning-tree protocol (RSTP) by taking advantage of point-to-point links (a port is connected to exactly one other bridge) and edge-port connections (a port is not connected to any additional bridges). Setting the link type to **auto** allows the spanning tree to automatically configure the link type based on the duplex of the link. Setting the link type to **point-to-point** allows a half-duplex link to act as if it were a point-to-point link.

spanning-tree path-cost <value>

Use the **spanning-tree path-cost** command to assign a cost to a bridge group that is used when computing the spanning-tree root path. To return to the default path-cost value, use the **no** form of this command.

Syntax Description

<value>	Assigns the bridge interface a path cost value to be used in spanning calculations (valid range: 0 to 65,535).

Default Values

By default, the path-cost value is set at 19.

Functional Notes

The specified value is inversely proportional to the likelihood the bridge interface will be chosen as the root path. Set the path-cost value lower to increase the chance the interface will be the root. To obtain the most accurate spanning-tree calculations, develop a system for determining path costs for links and apply it to all bridged interfaces.

Usage Examples

The following example assigns a path cost of 100 for bridge group 17 on a Frame Relay sub-interface:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#spanning-tree path-cost 100
```

Technology Review

Spanning-tree protocol provides a way to prevent loopback or parallel paths in bridged networks. Using the priority values and path costs assigned to each bridging interface, the spanning-tree protocol determines the root path and identifies whether to block or allow other paths.

spanning-tree port-priority <value>

Use the **spanning-tree port-priority** command to select the priority level of a port associated with a bridge. To return to the default bridge-group priority value, use the **no** version of this command.

Syntax Description

<value>	Priority value for the bridge group; the lower the value, the higher the priority (valid range: 0 to 255).

Default Values

By default, the bridge-group priority value is set at 128.

Functional Notes

The only time that this priority level is used is when two interfaces with a path to the root have equal cost. At that point, the level set in this command will determine which port the bridge will use. Set the priority value lower to increase the chance the interface will be used.

Usage Examples

The following example sets the maximum priority on the Frame Relay sub-interface labeled 1.16 in bridge group 17:

```
ProCurve(config)#interface frame-relay 1.16  
ProCurve(config-fr 1.16)#spanning-tree port-priority 0
```

ATM INTERFACE CONFIG COMMAND SET

To activate the ATM Interface Configuration mode, enter the **interface atm** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface atm 1
ProCurve(config-atm 1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

snmp trap [on page 723](#)

snmp trap link-status [on page 724](#)

snmp trap

Use the **snmp trap** command to enable all supported Simple Network Management Protocol (SNMP) traps on the interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces (except virtual Frame Relay interfaces and sub-interfaces) have SNMP traps enabled.

Usage Examples

The following example enables SNMP on the ATM interface:

```
ProCurve(config)#interface atm 1  
ProCurve(config-atm 1)#snmp trap
```

snmp trap link-status

Use the **snmp trap link-status** command to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the ATM interface:

```
ProCurve(config)#interface atm 1
ProCurve(config-atm 1)#no snmp trap link-status
```

ATM SUB-INTERFACE CONFIG COMMAND SET

To activate the ATM Interface Configuration mode, enter the **interface atm** command (and specify a sub-interface) at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycname> [on page 726](#)

atm routed-bridged ip [on page 729](#)

backup commands [begin on page 730](#)

bandwidth <value> [on page 738](#)

bridge-group <group#> [on page 739](#)

crypto map <mapname> [on page 740](#)

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password> [on page 742](#)

encapsulation [aal5mux | aal5snap] [on page 744](#)

fair-queue <threshold> [on page 745](#)

hold-queue <queue size> out [on page 746](#)

ip commands [begin on page 747](#)

mtu <size> [on page 765](#)

oam commands [begin on page 766](#)

pvc <VPI/VCI> [on page 768](#)

qos-policy out <mapname> [on page 769](#)

spanning-tree commands [begin on page 771](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic on an interface. Use the **no** form of this command to remove an access policy association.

Note *Configured access policies will only be active if the **ip firewall** command has been entered at the Global Configuration mode prompt to enable the Secure Router OS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.*

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor for (all access policy descriptors are case-sensitive).

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**.

Usage Examples

The following example associates the access policy **UnTrusted** (to allow inbound traffic to the Web server) to the ATM sub-interface labeled 1.1:

Enable the Secure Router OS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list **InWeb**:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access list with the ATM 1.1 interface:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an access policy that uses a configured access list. Secure Router OS access policies are used to permit, deny, or manipulate (using **NAT**) data for each physical interface. Each ACP consists of a selector (**access list**) and an action (**allow**, **discard**, **NAT**). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

nat source list <access list names> **address** <IP address> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list *<access list names>* **interface** *<interface>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter interface configuration mode for the desired interface and enter **access policy** *<policy name>*. The following example assigns access policy **MatchAll** to the ATM sub-interface:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#access-policy MatchAll
```


atm routed-bridged ip

Use the **atm routed-bridged ip** command to enable routed IP bridge encapsulation (RBE) on an interface. Use the **no** form of this command to disable RBE operation.

Syntax Description>

No subcommands.

Default Values

By default, routed bridge encapsulation is disabled.

Usage Examples

The following example enables routed bridge encapsulation:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#atm routed-bridged ip
```

backup auto-backup

Use the **backup auto-backup** command to configure the sub-interface to automatically attempt a backup upon failure.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically attempt backup upon a failure.

Usage Examples

The following enables automatic backup on the endpoint:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#backup auto-backup
```

backup auto-restore

Use the **backup auto-restore** command to configure the sub-interface to automatically discontinue backup when all network conditions are operational. Use the **no** form of this command to disable the auto-restore feature.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically restore the primary connection when the failure condition clears.

Usage Examples

The following configures the Secure Router OS to automatically restore the primary connection when the failure condition clears:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#backup auto-restore
```

backup backup-delay <seconds>

Use the **backup backup-delay** command to configure the amount of time the router will wait after the failure condition is recognized before attempting to backup the link. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the delay period (in seconds) a failure must be active before the Secure Router OS will enter backup operation on the interface (Range: 10 to 86,400 seconds)

Default Values

By default, the backup delay period is set to 10 seconds.

Usage Examples

The following configures the Secure Router OS to wait 60 seconds (on an endpoint with an active alarm condition) before attempting backup operation:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#backup backup-delay 60
```

backup call-mode <role>

Use the **backup call-mode** command to specify whether the configured backup interfaces answers or originates (or a combination of both) backup calls. Use the **no** form of this command to return to the default value.

Syntax Description

<role>	Selects the role the router will take in backup of this sub-interface.
answer	Answers and backs up primary link on failure.
answer-always	Answers and backs up regardless of primary link state.
originate	Originates backup call on primary link failure.
originate-answer	Originates or answers call on primary link failure.
originate-answer-always	Originates on failure; answers and backs up always.

Default Values

By default, **backup call-mode** is set to **originate-answer**.

Functional Notes

The majority of the configuration for Frame Relay backup is configured in the Frame Relay interface. However, the numbers dialed are configured in the primary interface. Full sample configurations follow:

Sample configuration for remote router (dialing out)

```
hostname "Remote7203dl"
enable password password
!
interface eth 0/1
 ip address 192.168.1.254 255.255.255.0
 no shutdown
!
interface modem 1/3
 no shutdown
!
interface t1 1/1
 coding b8zs
 framing esf
 clock source line
 tdm-group 1 timeslots 1-24
 no shutdown
!
interface fr 1 point-to-point
```

```
frame-relay lmi-type ansi
no shutdown
bind 1 t1 1/1 1 fr 1
!
interface fr 1.16 point-to-point
frame-relay interface-dlci 16
ip address 10.1.1.2 255.255.255.252
backup call-mode originate
backup number 5551111 analog
backup number 5552222 analog
!
ip route 0.0.0.0 0.0.0.0 10.1.1.1
!
line telnet 0 4
password password
```

Sample configuration for central router (dialing in)

```
hostname "Central7203dl"
enable password password
!
interface eth 0/1
ip address 192.168.100.254 255.255.255.0
no shutdown
!
interface modem 1/3
no shutdown
!
interface t1 1/1
coding b8zs
framing esf
clock source line
tdm-group 1 timeslots 1-24
no shutdown
!
interface fr 1 point-to-point
frame-relay lmi-type ansi
no shutdown
bind 1 t1 1/1 1 fr 1
!
interface fr 1.100 point-to-point
frame-relay interface-dlci 100
```

```
ip address 10.1.1.1 255.255.255.252
backup call-mode answer
backup number 555-8888 analog
!
line telnet 0 4
password password
```

Usage Examples

The following configures the Secure Router OS to answer backup calls on this endpoint but never generate calls:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#backup call-mode answer-always
```

backup connect-timeout <seconds>

Use the **backup connect-timeout** command to specify the number of seconds to wait for a connection after a call is attempted before trying to call again or dialing a different number. It is recommended this number be greater than 60.

Syntax Description

<seconds>	Selects the amount of time in seconds that the router will wait for a connection before attempting another call (valid range: 10 to 300).

Default Values

By default, the backup connect-timeout period is set to 60 seconds.

Usage Examples

The following configures the Secure Router OS to wait 120 seconds before retrying a failed backup call:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#backup connect-timeout 120
```


backup force <state>

Use the **backup force** command to manually override the automatic backup feature. This can be used to force a link into backup to allow maintenance to be performed on the primary link without disrupting data. Use the **no** form of this command to return to the normal backup operation state.

Syntax Description

<state>	Selects the forced backup state of the sub-link.
backup	Forces backup regardless of primary link state.
primary	Forces primary link regardless of its state.

Default Values

By default, this feature is disabled.

Usage Examples

The following configures the Secure Router OS to force this endpoint into backup:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#backup force backup
```

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Enter bandwidth in kbps.

Default Values

To view default values, use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the ATM sub-interface to 10 Mbps:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#bandwidth 10000
```

bridge-group <group#>

Use the **bridge-group** command to assign an interface to the specified bridge group. Use the **no** form of this command to remove the interface from the bridge group.

Syntax Description

<group#>	Assigns a bridge group number to the interface (range is 1 to 255).

Default Values

By default, there are no configured bridge groups.

Functional Notes

A bridged network can provide excellent traffic management to reduce collisions and limit the amount of bandwidth wasted with unnecessary transmissions when routing is not necessary. Any two interfaces can be bridged (Ethernet to T1 bridge, Ethernet to Frame Relay sub-interface).

Usage Examples

The following example assigns the ATM sub-interface labeled 1.1 to bridge group 1:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#bridge-group 1
```

crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<mapname> Specifies the crypto map name that you wish to assign to the interface.

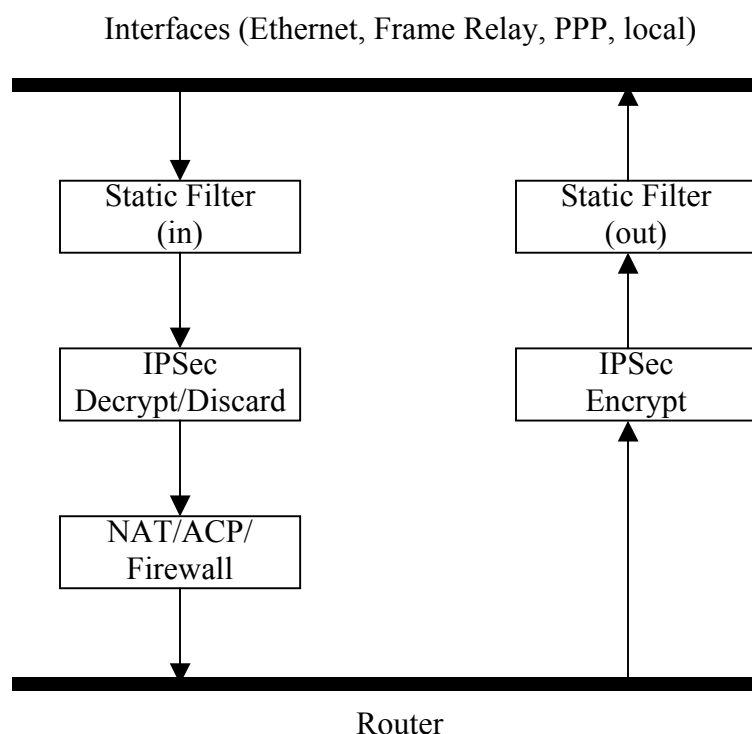
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following notes in mind.

When defining the policy class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the unencrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical Secure Router OS data-flow logic.



As shown in the previous diagram, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only unencrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy class on an interface. When specifying the ACLs for a crypto map, the source information is the private local side, unencrypted source of the data. The destination information will be the far end, unencrypted destination of the data. However, ACLs for a policy class work in reverse. The source information for the ACL in a policy class is the far end. The destination information is the local side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the ATM sub-interface:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#crypto map MyMap
```

**dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname>
<username> <password>**

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

See **Functional Notes**, below, for argument descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A web-based interface provides two levels of control over your domain, catering to average or power users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs, and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service, in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

If your IP address doesn't change often or at all, but you still want an easy name to remember it by (without having to purchase your own domain name) Static DNS service is ideal for you.

When to Use Dynamic DNS

If your IP address does not change often (or at all) but you still want an easy name to remember it by (without having to purchase your own domain name), Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com), you need Custom DNS service, which also provides full dynamic and static IP address support.

Dynamic DNS service can be extremely helpful for site-to-site VPN connections using dynamic IP addressing. Instead of creating a tunnel to the IP-address, a tunnel to the DNS-name is established.

Usage Examples

The following example sets the dynamic-dns to dyndns-custom with hostname host, username user, and password pass:

```
ProCurve(config-atm 1.1)#dynamic-dns dyndns-custom host user pass
```

encapsulation [aal5mux | aal5snap]

Use the **encapsulation** command to configure the encapsulation type for the ATM adaption Layer (AAL) of the ATM Protocol Reference model.

Variations of this command include the following:

encapsulation aal5mux [ip | ppp]

encapsulation aal5snap

Syntax Description

aal5mux	Specifies encapsulation type for multiplexed virtual circuits. A protocol must be specified.
aal5snap	Specifies encapsulation type that supports LLC/SNAP protocols.
[ip ppp]	Specifies protocol type used for multiplexed virtual circuits (aal5mux).

Default Values

By default, the encapsulation type is **aal5snap**.

Functional Notes

For PPP and PPPoE, the encapsulation type can be **aal5snap** or **aal5mux ppp**.

For IP with no bridging, the encapsulation type can be **aal5snap** or **aal5mux ip**.

For IP with bridging, the encapsulation type can only be **aal5snap**.

For bridging, the encapsulation type can only be **aal5snap**.

Usage Examples

The following example sets the encapsulation type to **aal5snap**:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#encapsulation aal5snap
```


fair-queue *<threshold>*

Use the **fair-queue** command to enable weighted fair queuing (WFQ) on an interface. Use the **no** form of this command to disable WFQ and enable first-in-first-out (FIFO) queueing for an interface. WFQ is enabled by default for WAN interfaces.

Syntax Description

<i><threshold></i>	Optional. Specifies the maximum number of packets that can be present in each conversation sub-queue. Packets received for a conversation after this limit is reached are discarded. Range: 16 to 512.

Default Values

By default, **fair-queue** is enabled with a threshold of 64 packets.

Usage Examples

The following example enables WFQ on the interface with a threshold set at 100 packets:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#fair-queue 100
```

hold-queue <queue size> out

Use the **hold-queue out** command to change the overall size of an interface's WAN output queue.

Syntax Description

<queue size>	Specifies the total number of packets the output queue can contain before packets are dropped. Range: 16 to 1000.

Default Values

The default queue size for WFQ is 400. The default queue size for PPP FIFO and Frame Relay round-robin is 200.

Usage Examples

The following example sets the overall output queue size to 700:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#hold-queue 700 out
```

ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<i>listname</i>	Specifies the assigned IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the router to only allow Telnet traffic into the ATM sub-interface:

```
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#interface atm 1.1
ProCurve(config-atm 1.1)#ip access-group TelnetOnly in
```

ip address dhcp

Use the **ip address dhcp** command to use Dynamic Host Configuration Protocol (DHCP) to obtain an address on the interface. Use the **no** form of this command to remove a configured IP address (using DHCP) and disable DHCP operation on the interface.

ip address dhcp [**client-id** [*<interface>* | *<identifier>*] **hostname** *<"string">*]

Syntax Description

client-id	Specifies the client identifier used when obtaining an IP address from a DHCP server.
<i><interface></i>	Specifies an interface to define the client identifier as the hexadecimal MAC address of the specified interface (including a hexadecimal number added to the front of the MAC address to identify the media type). For example, specifying the client-id ethernet 0/1 (where the Ethernet interface has a MAC address of d217.0491.1150) defines the client identifier as 01:d2:17:04:91:11:50 (where 01 defines the media type as Ethernet). Refer to <i>hardware-address <hardware-address> <type></i> on page 283 for a detailed listing of media types.
<i><identifier></i>	Specifies a custom client-identifier using a text string (that is converted to a hexadecimal equivalent) or 7 to 28 hexadecimal numbers (with colon delimiters). For example, a custom client identifier of 0f:ff:ff:ff:ff:51:04:99:a1 may be entered using the <i><identifier></i> option.
host name	Specifies a text string (to override the global router name) to use as the name in the DHCP option 12 field.
<i><"string"></i>	String (encased in quotation marks) of up to 35 characters to use as the name of the host for DHCP operation.
no-default-route	Specifies that the Secure Router OS not install the default route obtained via DHCP.
no-domain-name	Specifies that the Secure Router OS not install the domain name obtained via DHCP.
no-nameservers	Specifies that the Secure Router OS not install the DNS servers obtained via DHCP.

Default Values

client-id	By default, the client identifier is populated using the following formula: TYPE: INTERFACE SPECIFIC INFO : MAC ADDRESS Where TYPE specifies the media type in the form of one hexadecimal byte (refer to <i>hardware-address</i> < <i>hardware-address</i> > < <i>type</i> > on page 283 for a detailed listing of media types), and the MAC ADDRESS is the Media Access Control (MAC) address assigned to the first Ethernet interface in the unit in the form of six hexadecimal bytes. (For units with a single Ethernet interface, the MAC ADDRESS assigned to Ethernet 0/1 is used in this field).
hostname	By default, the hostname is the name configured using the Global Configuration hostname command.
<"string">	By default, the hostname is the name configured using the Global Configuration hostname command.

Functional Notes

DHCP allows interfaces to acquire a dynamically assigned IP address from a configured DHCP server on the network. Many Internet Service Providers (ISPs) require the use of DHCP when connecting to their services. Using DHCP reduces the number of dedicated IP addresses the ISP must obtain. Consult your ISP to determine the proper values for the **client-id** and **hostname** fields.

Usage Examples

The following example enables DHCP operation on the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip address dhcp
```

ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface. Use the optional **secondary** keyword to define a secondary IP address. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional. Defines a secondary IP address for the specified interface.

Default Values

By default, there are no assigned IP addresses.

Functional Notes

Use secondary IP addresses to allow dual subnets on a single interface (when you need more IP addresses than the primary subnet can provide). When using secondary IP addresses, avoid routing loops by verifying that all devices on the network segment are configured with secondary IP addresses on the secondary subnet.

Usage Examples

The following example configures a secondary IP address of **192.22.72.101/30**:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#ip address 192.22.72.101 255.255.255.252 secondary
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip dhcp [release | renew]

Use the **ip dhcp** command to release or renew the DHCP IP address. This command is only applicable when using DHCP for IP address assignment.

Syntax Description

release	Releases DHCP IP address.
renew	Renews DHCP IP address.

Default Values

No default values required for this command.

Usage Examples

The following example releases the IP DHCP address for the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip dhcp release
```

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note *The **ip helper** command must be used in conjunction with the **ip forward-protocol** command to configure the Secure Router OS to forward UDP broadcast packets. See **ip forward-protocol udp <port number>** on page 319 for more information.*

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is UDP.
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip helper-address 192.33.5.99
```


ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

helper-enable	Tells this downstream interface to use the global helper address.
immediate-leave	Specifies that if only one host (or IGMP snooping switch) is connected to the interface, when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	Controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave-group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer, if no receiver responds, the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65535 ms. Default: 1000 ms.
querier-timeout <seconds>	Specifies the number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60 to 300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Specifies the interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Specifies the maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically-connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip igmp last-member-query-interval 200
```

ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface and place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding. Refer to *ip mcast-stub helper-address <ip address>* on page 327 and *ip mcast-stub upstream* on page 756 for more information.

Usage Examples

The following example enables multicast forwarding and IGMP on the interface:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip mcast-stub downstream
```

ip mcast-stub helper-enable

Use the **ip mcast-stub helper-enable** command to assign the **ip mcast-stub helper-address** as the IGMP proxy. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address**, **ip mcast-stub upstream**, and **ip mcast-stub downstream** commands. When enabled, the interface becomes a helper forwarding interface. The IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Refer to *ip mcast-stub helper-address <ip address>* on page 327, *ip mcast-stub downstream* on page 754, and *ip mcast-stub upstream* on page 756 for more information.

Usage Examples

The following example sets the helper address as the IGMP proxy:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip mcast-stub helper-enable
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface. Refer to *ip mcast-stub helper-address <ip address>* on page 327 and *ip mcast-stub downstream* on page 754 for more information.

Usage Examples

The following example enables multicast forwarding on the interface:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip mcast-stub upstream
```

ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key <password>	Assigns a simple text authentication password to be used by other routers using the OSPF simple password authentication.
cost <value>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65,535.
dead-interval <seconds>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, neighboring devices will determine that the device is down. Range: 0 to 32,767.
hello-interval <seconds>	Specifies the interval between hello packets sent on the interface. Range: 0-32,767.
message-digest-key <keyid> md5 <key>	Configures OSPF Message Digest 5 (MD5) authentication (16-byte maximum) keys.
priority <value>	Sets the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <seconds>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32,767.
transmit-delay <seconds>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32,767.

Default Values

retransmit-interval <seconds>	5 seconds
transmit-delay <seconds>	1 second
hello-interval <seconds>	10 seconds: Ethernet, point-to-point, Frame Relay, and PPP
dead-interval <seconds>	40 seconds

Usage Example

The following example sets the maximum number of seconds allowed between hello packets to 25,000:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip ospf dead-interval 25000
```

ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Specifies message-digest authentication type.
null	Specifies that no authentication be used.

Default Values

By default, this is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip ospf authentication null
```

ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Sets the network type for broadcast.
point-to-point	Sets the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP and Frame Relay default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#ip ospf network broadcast
```

ip proxy-arp <address> <subnet mask>

Use the **ip proxy-arp** to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, proxy ARP is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the Secure Router OS will respond to all proxy ARP requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip proxy-arp
```


ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Accepts only RIP version 1 packets received on the interface.
2	Accepts only RIP version 2 packets received on the interface.

Default Values

By default, all interfaces implement RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip receive version** to specify a RIP version that will override the **version** (in the Router RIP) configuration.

The Secure Router OS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the ATM sub-interface 1.1 to accept only RIP version 2 packets:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip rip receive version 2
```

ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Transmits only RIP version 1 packets on the interface
2	Transmits only RIP version 2 packets on the interface

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** to specify a RIP version that will override the **version** (in the Router RIP) configuration.

The Secure Router OS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the ATM sub-interface 1.1 to transmit only RIP version 2 packets:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip rip send version 2
```

ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note *Using Network Address Translation (NAT) or the Secure Router OS firewall capabilities on an interface requires process switching mode (using the **no ip route-cache** command).*

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#ip route-cache
```

ip unnumbered <interface>

Use the **ip unnumbered** command to use the IP address assigned to the specified interface for all IP processing on the active interface. Use the **no** form of this command to remove the unnumbered configuration.

Syntax Description

<interface>	Specifies the interface (in the format type slot/port) that contains the IP address to use as the source address for all packets transmitted on this interface.

Default Values

By default, all interfaces are configured to use a specified IP address (using the **ip address** command).

Functional Notes

If **ip unnumbered** is enabled on an interface, all IP traffic from the interface will use a source IP address taken from the specified interface. For example, specifying **ip unnumbered eth 0/1** while in the Frame Relay Sub-Interface Configuration mode configures the Frame Relay sub-interface to use the IP address assigned to the Ethernet interface for all IP processing. In addition, the Secure Router OS uses the specified interface information when sending route updates over the unnumbered interface.

Usage Examples

The following example configures the ATM sub-interface 1.1 to use the IP address assigned to the Ethernet interface (**eth 0/1**):

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#ip unnumbered eth 0/1
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit (MTU) size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size> Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:

ATM interfaces	64 to 1520
Ethernet interfaces	64 to 1500
FDL interfaces	64 to 256
HDLC interfaces	64 to 1520
Loopback interfaces	64 to 1500
Tunnel interfaces	64 to 18,190
Virtual Frame Relay sub-interfaces	64 to 1520
Virtual PPP interfaces	64 to 1500

Default Values

<size> The default values for the various interfaces are listed below:

ATM interfaces	1500
Ethernet interfaces	1500
FDL interfaces	256
HDLC interfaces	1500
Loopback interfaces	1500
Tunnel interfaces	1500
Virtual Frame Relay sub-interfaces	1500
Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#mtu 1200
```

oam-pvc managed *<frequency>*

Use the **oam-pvc managed** command to enable end-to-end F5 Operation, Administration, and Maintenance (OAM) loopback cell generation and OAM management for an ATM interface. Use the **no** form of this command to disable generation of OAM loopback cells.

Syntax Description

<i><frequency></i>	Specifies the time delay between transmitting OAM loopback cells. The range is from 0 to 600 seconds.

Default Values

By default, the frequency is 1 second.

Usage Examples

The following example enables OAM loopback cell generation with a frequency of 5 seconds:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#oam-pvc manage 5
```

oam retry <up-count> <down-count> <retry-frequency>

Use the **oam retry** command to configure parameters related to Operation, Administration, and Maintenance (OAM) management for an ATM interface. Use the **no** form of this command to disable OAM management parameters.

Syntax Description

<up-count>	Specifies the number of consecutive end-to-end F5 OAM loopback cell responses that must be received in order to change a PVC connection state to up. The range is from 1 to 255.
<down-count>	Specifies the number of consecutive end-to-end F5 OAM loopback cell responses that are not received in order to change a PVC state to down. The range is from 1 to 255.
<retry-frequency>	Specifies the frequency (in seconds) that end-to-end F5 OAM loopback cells are transmitted when a change in the up/down state of a PVC is being verified. The range is from 1 to 600.

Default Values

By default, the up-count is set to 3, the down-count is set to 5, and the retry frequency is 1.

Usage Examples

The following example configures the OAM parameters with an up-count of 2, down-count of 2, and retry frequency of 10:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#oam retry 2 2 10
```

pvc <VPI/VCI>

Use the **pvc** command to select the ATM virtual link for this sub-interface. Use the **no** form of this command to remove the link.

Syntax Description

<VPI/VCI>	Specifies the ATM network virtual path identifier (VPI) for this PVC and the ATM network virtual path identifier (VPI) for this PVC. The VPI value range is 0 to 255, and the VCI value range is 32 to 65,535.

Default Values

No default value is necessary for this command.

Usage Examples

The following example sets the VPI to 8 and the VCI to 35:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#pvc 8/35
```


qos-policy out <mapname>

Use the **qos-policy out** command to apply a previously-configured QoS map to an interface. Use the **no** form of this command to remove the map from the interface. The keyword **out** specifies that this policy will be applied to outgoing packets.

Syntax Description

<mapname>	Specifies the name of a previously-created QoS map (refer to <i>qos map</i> <mapname> <sequence number> on page 363 for more information).

Default Values

No default value is necessary for this command.

Usage Examples

The following example applies the QoS map **VOICEMAP** to the ATM sub-interface 1.1:

```
ProCurve(config)#interface atm 1.1
```

```
ProCurve(config-atm 1.1)#qos-policy out VOICEMAP
```

spanning-tree bpdudfilter [enable | disable]

Use the **spanning-tree bpdudfilter** command to block BPDUs from being transmitted and received on this interface. To return to the default value, use the **no** form of this command.

Syntax Description

enable	Enables the BPDU filter.
disable	Disables the BPDU filter.

Default Values

By default, this command is set to **disable**.

Functional Notes

The purpose of this command is to remove a port from participation in the spanning tree. This might be beneficial while debugging a network setup. It normally should not be used in a live network.

Usage Examples

The following example enables the BPDU filter on the interface:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#spanning-tree bpdudfilter enable
```

spanning-tree bpduguard [enable | disable]

Use the **spanning-tree bpduguard** command to block BPDUs from being received on this interface. To return to the default value, use the **no** form of this command.

Syntax Description

enable	Enables the BPDU block.
disable	Disables the BPDU block.

Default Values

By default, this command is set to **disable**.

Usage Examples

The following example enables the BPDU guard on the interface:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#spanning-tree bpduguard enable
```

spanning-tree edgeport

Use the **spanning-tree edgeport** command to set this interface to be an edgeport. This command overrides the Global setting. Use the **no** form of this command to return the interface to normal operation (non-edgeport).

Syntax Description

No subcommands.

Default Values

By default, this command is set to disable.

Usage Examples

The following example configures the interface to be an edgeport:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#spanning-tree edgeport
```

An individual interface can be configured to not be considered an edgeport. For example:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#no spanning-tree edgeport
```

spanning-tree link-type [auto | point-to-point | shared]

Use the **spanning-tree link-type** command to configure the spanning-tree protocol link type for an interface. To return to the default value, use the **no** form of this command.

Syntax Description

auto	Determines link type by the port's duplex settings.
point-to-point	Manually sets link type to point-to-point, regardless of duplex settings.
shared	Manually sets link type to shared, regardless of duplex settings.

Default Values

By default, a port is set to **auto**.

Functional Notes

This command overrides the default link-type setting determined by the duplex of the individual port. By default a port configured for half-duplex is set to **shared** link type, and a port configured for full-duplex is set to **point-to-point** link type. Setting the link type manually overrides the default and forces the port to use the specified link type. Using the **link-type auto** command, restores the convention of determining link type based on duplex settings.

Usage Examples

The following example forces the link type to point-to-point, even if the port is configured to be half-duplex:

```
ProCurve(config)#bridge 1 protocol ieee
ProCurve(config)#interface atm 1.1
ProCurve(config-atm 1.1)#spanning-tree link-type point-to-point
```

Technology Review

Rapid transitions are possible in rapid spanning-tree protocol (RSTP) by taking advantage of point-to-point links (a port is connected to exactly one other bridge) and edge-port connections (a port is not connected to any additional bridges). Setting the link type to **auto** allows the spanning tree to automatically configure the link type based on the duplex of the link. Setting the link type to **point-to-point** allows a half-duplex link to act as if it were a point-to-point link.

spanning-tree path-cost <value>

Use the **spanning-tree path-cost** command to assign a cost to a bridge group that is used when computing the spanning-tree root path. To return to the default path-cost value, use the **no** form of this command.

Syntax Description

<value>	Assigns a number to the bridge interface to be used as the path cost in spanning calculations (valid range: 0 to 65,535).

Default Values

By default, the path-cost value is set to 19.

Functional Notes

The specified value is inversely proportional to the likelihood the bridge interface will be chosen as the root path. Set the path-cost value lower to increase the chance the interface will be the root. To obtain the most accurate spanning tree calculations, develop a system for determining path costs for links and apply it to all bridged interfaces.

Usage Examples

The following example assigns a path cost of 100 for bridge group 17 on an ATM sub-interface:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#spanning-tree path-cost 100
```

Technology Review

Spanning-tree protocol provides a way to prevent loopback or parallel paths in bridged networks. Using the priority values and path costs assigned to each bridging interface, the spanning-tree protocol determines the root path and identifies whether to block or allow other paths.

spanning-tree port-priority <value>

Use the **spanning-tree port-priority** command to select the priority level of a port associated with a bridge. To return to the default bridge-group priority value, use the **no** version of this command.

Syntax Description

<value>	Assigns a priority value for the bridge group; the lower the value, the higher the priority (valid range: 0 to 255).

Default Values

By default, the priority value is set to 128.

Functional Notes

The only time that this priority level is used is when two interfaces with a path to the root have equal cost. At that point, the level set in this command will determine which port the bridge will use. Set the priority value lower to increase the chance the interface will be used.

Usage Examples

The following example sets the maximum priority on the ATM sub-interface labeled 1.1 in bridge group 17:

```
ProCurve(config)#interface atm 1.1  
ProCurve(config-atm 1.1)#spanning-tree priority 0
```

ADSL INTERFACE CONFIG COMMAND SET

To activate the ADSL Interface Configuration mode, enter the **interface adsl** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface adsl 1/1
ProCurve(config-adsl 1/1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

retrain [on page 777](#)

snr-margin [*showtime monitor* | *training monitor*] <margin> [on page 778](#)

training-mode [*ADSL2* | *ADSL2+* | *G.DMT* | *G.LITE* | *Multi-Mode* | *READSL2* | *T1.413*] [on page 779](#)

retrain

Use the **retrain** command to force the modem to retrain.

Syntax Description

No subcommands.

Default Values

No default is necessary for this command.

Usage Examples

The following example forces a modem retrain:

```
ProCurve(config)#interface adsl 0/1  
ProCurve(config-adsl 1/1)#retrain
```

snr-margin [showtime monitor | training monitor] <margin>

Use the **snr-margin** command to enable monitoring and set the minimum signal-to-noise (SNR) ratio during training and showtime. Use the **no** form of this command to disable monitoring.

Syntax Description

showtime monitor	Enables margin monitoring to retrain the ADSL interface if the specified minimum margin is violated during showtime.
training monitor	Enables margin monitoring to retrain the ADSL interface if the specified minimum margin is violated during training.
<margin>	Sets the minimum SNR margin in dB. The range is from 1 to 15.

Default Values

By default, SNR margin monitoring is disabled.

Usage Examples

The following example enables SNR margin monitoring during showtime with a minimum level of 7 dB:

```
ProCurve(config)#interface adsl 0/1  
ProCurve(config-adsl 1/1)#snr-margin showtime monitor 7
```

training-mode [ADSL2 | ADSL2+ | G.DMT | G.LITE | Multi-Mode | READSL2 | T1.413]

Use the **training mode** command to configure the ADSL training mode.

Syntax Description

ADSL2	Specifies ITU G.992.3 Annex A mode.
ADSL2+	Specifies ITU G.992.5 ADSL2+ mode.
G.DMT	Specifies ANSI full-rate mode.
G.LITE	Specifies ANSI splitterless mode.
Multi-Mode	Specifies auto detect mode. When set to multi-mode , the ADSL interface attempts to train to the DSLAM using each of the supported training modes until a match is found.
READLS2	Specifies ITU G.992.3 Annex L mode.
T1.413	Specifies ANSI T1.413 mode.

Default Values

By default, the training mode is set to **Multi-Mode**.

Usage Examples

The following example sets the training mode to T1.413:

```
ProCurve(config)#interface adsl 0/1
ProCurve(config-adsl 1/1)#training-mode T1.413
```

BGP CONFIGURATION COMMAND SET

To activate the BGP Configuration mode, enter the **bgp** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

bgp fast-external-fallover [on page 781](#)

bgp log-neighbor-changes [on page 782](#)

bgp router-id <ip address> [on page 783](#)

distance bgp <external> <internal> <local> [on page 784](#)

hold-timer <hold time> [on page 785](#)

bgp fast-external-fallover

Use the **bgp fast-external-fallover** command to enable the fast-external-fallover feature.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Functional Notes

When enabled, if the link interface over which the router is communicating with a BGP peer goes down, the BGP session with that peer is immediately cleared. When fallover is disabled and the link goes down, the session is maintained until the BGP hold timer expires.

Usage Examples

The following example enables this option:

```
ProCurve(config)#router bgp 65000
```

```
ProCurve(config-bgp)#bgp fast-external-fallover
```

bgp log-neighbor-changes

Use the **bgp log-neighbor-changes** command to control the logging of neighbor state changes. Use the **no** form of this command to return to the default setting.

Syntax Description

No subcommands.

Default Values

By default, neighbor changes are not logged.

Functional Notes

This command controls logging of BGP neighbor state changes (up/down) and resets. This information is useful for troubleshooting and determining network stability.

Usage Examples

The following example enables logging of BGP neighbor state changes:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#bgp log-neighbor-changes
```

bgp router-id <ip address>

Use the **bgp router-id** command to specify the IP address that the router should use as its BGP router ID. Use the **no** form of this command to return to the default setting.

Syntax Description

<ip address>	Designates the IP address this router should use as its BGP router ID.

Default Values

By default, no router ID is configured. The default action is detailed in *Functional Notes*, below.

Functional Notes

This command allows an IP address to be specified for use as the BGP router ID. If no IP address is configured at BGP startup, it uses the highest IP address configured on a loopback interface. If no loopback interfaces are configured, it uses the highest IP address configured on any interface that is active. If the specified router ID is changed, existing sessions with BGP neighbors are reset.

Usage Examples

The following example configures IP address 10.0.0.1 as the BGP router ID:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#bgp router-id 10.0.0.1
```

distance bgp *<external>* *<internal>* *<local>*

Use the **distance bgp** command to set the administrative distance for BGP routes. Use the **no** form of this command to return to the default setting.

Syntax Description

<i><external></i>	Sets the administrative distance for BGP routes learned via eBGP sessions. A value of 255 means the route is not installed. Range: 1 to 254.
<i><internal></i>	Sets the administrative distance for BGP routes learned via iBGP sessions. A value of 255 means the route is not installed. Range: 1 to 254.
<i><local></i>	Sets the administrative distance for BGP routes learned via the network command and redistribution. A value of 255 means the route is not installed. Range: 1 to 254.

Default Values

By default external is set to 20, internal to 200, and local to 200. Normally, these default settings should not be changed.

Functional Notes

This command sets the administrative distance for BGP routes. The administrative distance is a local variable that allows a router to choose the best route when there are multiple paths to the same network. Routes with smaller administrative distances are favored.

Usage Examples

The following example gives external BGP routes an administrative distance of 30, internal BGP routes an administrative distance of 200, and local routes an administrative distance of 240:

```
ProCurve(config)#router bgp 65000
```

```
ProCurve(config-bgp)#distance bgp 30 200 240
```


hold-timer *<hold time>*

Use the **hold-timer** command to set the default hold time for all neighbors in the BGP process.

Syntax Description

<i><hold time></i>	Specifies a time interval (in seconds) within which a keepalive must be received from a peer before it is declared dead peer. Range: 0 to 65535

Default Values

By default, the hold time is 90 seconds.

Functional Notes

Using the **hold-timer** command in BGP configuration mode sets the default hold time for all neighbors in that BGP process. Using the **hold-timer** command in BGP neighbor configuration mode sets the hold time for only that neighbor. The peers will negotiate and use the lowest configured setting. The keepalive interval will be set to one third of the negotiated hold time.

Usage Examples

The following example sets a hold time of 120 seconds for a specific neighbor, with an understood keepalive interval of 40 seconds:

```
ProCurve(config)#router bgp 65000
ProCurve(config-bgp)#hold-timer 120
```

BGP NEIGHBOR CONFIGURATION COMMAND SET

To activate the BGP Neighbor Configuration mode, enter the **bgp-neighbor** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#router bgp 65000
ProCurve(config-bgp)#neighbor 65.4.3.192
ProCurve(config-bgp-neighbor)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
exit [on page 1003](#)
shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

advertisement-interval <seconds> [on page 787](#)
as-path-list <listname> [in | out] [on page 788](#)
ebgp-multihop <hop count> [on page 789](#)
hold-timer <hold time> [on page 790](#)
local-as <as-number> [on page 791](#)
next-hop-self [on page 792](#)
password <password> [on page 793](#)
prefix-list <listname> [in | out] [on page 794](#)
remote-as <as-number> [on page 795](#)
route-map <map-name> [in | out] [on page 796](#)
send-community standard [on page 797](#)
soft-reconfiguration inbound [on page 798](#)
update-source <interface> [on page 799](#)

advertisement-interval <seconds>

Use the **advertisement-interval** command to configure the Secure Router OS to specify how long the BGP process waits before sending updates to the neighbor.

Syntax Description

<seconds>	Specifies the advertisement interval in seconds. Range: 0 to 600.

Default Values

By default, the advertisement interval is 30 seconds for external neighbors and 5 seconds for internal neighbors.

Functional Notes

This command sets the minimum interval between sending updates to the specified neighbor.

Usage Examples

The following example configures the BGP process to wait at least 100 seconds before sending updates to the neighbor:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#advertisement-interval 100
```

as-path-list <listname> [in | out]

Use the **as-path-list** command to assign a predefined autonomous system (AS) path list to a BGP neighbor. This list is then used to filter inbound and/or outbound BGP route updates. Use the **no** form of this command to discontinue use of the list.

Syntax Description

<listname>	Assigns an AS path list to this BGP neighbor.
in	Specifies the filtering of all inbound BGP route updates.
out	Specifies the filtering of all outbound BGP route updates.

Default Values

By default, no AS path lists are specified for filtering.

Functional Notes

Before they can be assigned to a neighbor, AS path lists must first be defined using the **ip as-path-list** command. See *ip as-path-list <listname>* [on page 295](#) for more information.

Usage Examples

The following example uses the **no15** AS path list to filter all inbound BGP route updates:

```
ProCurve(config)#router bgp 65000
ProCurve(config-bgp)#neighbor 65.4.3.192
ProCurve(config-bgp-neighbor)#as-path-list no15 in
```

ebgp-multihop *<hop count>*

Use the **ebgp-multihop** command to configure the maximum hop count of BGP messages to a neighbor. Use the **no** form of this command to return to the default setting.

Syntax Description

<i><hop count></i>	Specifies the maximum hop count of BGP messages to a neighbor. Range: 1 to 254.

Default Values

By default, eBGP multihop is set to 1.

Functional Notes

This command allows an eBGP neighbor to be on a network that is not directly connected. Normally, eBGP peers are directly connected. In certain applications, a non-BGP device such as a firewall or router may reside between eBGP peers. In this case, the eBGP multihop command is required to allow updates to have a TTL greater than 1 and to allow received BGP updates to be added to the BGP table when the next hop address is not directly connected.

Usage Examples

The following example allows a BGP message to travel 10 hops to a neighbor:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#ebgp-multihop 10
```

hold-timer *<hold time>*

Use the **hold-timer** command to set the default hold time for all neighbors in the BGP process.

Syntax Description

<i><hold time></i>	Specifies a time interval (in seconds) within which a keepalive must be received from a peer before it is declared dead peer. Range: 0 to 65,535.

Default Values

By default, the hold time is 90 seconds.

Functional Notes

Using the **hold-timer** command in BGP configuration mode sets the default hold time for all neighbors in that BGP process. Using the **hold-timer** command in BGP neighbor configuration mode sets the hold time for only that neighbor. The peers will negotiate and use the lowest configured setting. The keepalive interval will be set to one-third of the negotiated hold time.

Usage Examples

The following example sets a hold time of 120 seconds for a specific neighbor, with an understood keepalive interval of 40 seconds:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#hold-timer 120
```

local-as <as-number>

Use the **local-as** command to specify an autonomous system (AS) number for the unit to use when communicating with this BGP neighbor. Use the **no** form of this command to return to default settings.

Syntax Description

<as-number>	Specifies the AS number to use when communicating with this neighbor. Must be different than the AS number for this router and the peer router. Only valid for eBGP connections. Range is 0 to 65,535.

Default Values

By default, no local AS number is defined. The router's BGP AS number is used.

Functional Notes

This command substitutes a different AS number to be used for communicating with this BGP neighbor. (other than the one the router is actually a member of). This can be used to satisfy network designs requiring a customer to appear as one AS number when communicating with one Internet service provider (ISP) and another when communicating with another ISP.

Usage Examples

The following example configures this BGP neighbor's AS number to be **300**:

```
ProCurve(config)#router bgp 65000
ProCurve(config-bgp)#neighbor 65.4.3.192
ProCurve(config-bgp-neighbor)#local-as 300
```

Technology Review

This router appears (to the peer router) to be in the AS specified with the **local-as** command. Therefore all routes learned from the peer have this number prepended to the AS path. In network advertisements from routers using the **local-as** command, the router's true AS number (the number specified using the **router bgp as-number** command) is prepended to the AS path attribute, and the local AS (the number specified using the **local-as** command) is prepended to the AS path attribute. This makes it appear that the path to the network is first through the local AS, and then through the true AS. To further illustrate, consider the following example network.

In this network:

- Router A is in AS 100.
- Router B is in AS 300.
- Router A is an eBGP peer with Router B.
- Router A's connection to Router B specifies a **local-as** of 200.
- Router B is configured to connect to Router A in AS 200.

Therefore:

- To Router B, all aspects of Router A appear as AS 200.
- Networks advertised from Router A to Router B will have the AS path **200 100** prepended to the AS path attribute.
- Router A will add AS 200 to the AS path of networks learned from Router B.

next-hop-self

Use the **next-hop-self** command to force the next hop attribute to be changed to this unit's address when advertising networks that would not have the next hop changed under normal rules. Normal next hop rules are described in the *Functional Notes* section below. Use the **no** form of this command to cause normal next hop rules to apply.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled and normal next hop rules apply.

Functional Notes

In eBGP, routes are normally advertised with a next hop set to the IP address that the receiving peer has configured in its neighbor statement for this router. In the eBGP case where the receiving router is in the same subnet as the current next hop, the current next hop is not changed.

For broadcast multiaccess networks (Ethernet), this provides more efficient routing. For non-broadcast multiaccess networks (NBMA) such as Frame Relay with a partial mesh using point-to-multipoint circuits, this rule can cause significant problems. Since the partial mesh is on the same subnet, BGP applies the rule of not changing the next hop address, rendering invalid routes in certain topologies. This is one case where this command is necessary to solve a problem.

Usage Examples

The following example enables **next-hop-self**:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#next-hop-self
```


password <password>

Use the **password** command to enable MD5 password authentication on TCP. Use the **no** form of this command to disable authentication.

Syntax Description

<password>	Specifies the password string to be used for authentication. The password is case-sensitive and must not exceed 80 characters.

Default Values

By default, authentication is disabled.

Functional Notes

Authentication must be configured on both peers using the same password. Every BGP TCP segment sent is authenticated. Configuring authentication causes an existing session to be torn down and re-established using the currently specified authentication.

Usage Examples

The following example enables authentication for this BGP neighbor and sets a password of **user1**:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#password user1
```

prefix-list <listname> [in | out]

Use the **prefix-list** command to assign a predefined prefix list to a BGP neighbor. The list is then used to filter BGP route updates received and/or sent from/by the specified peer. Use the **no** form of this command to discontinue use of the prefix list.

Syntax Description

<listname>	Assigns a prefix list to this BGP neighbor.
in	Specifies the filtering of all inbound BGP route updates received from the specified peer.
out	Specifies the filtering of all outbound BGP route updates being sent to the specified peer.

Default Values

By default, no prefix lists are specified for filtering.

Functional Notes

Before they can be assigned to a BGP neighbor, prefix lists must first be defined using the **ip prefix-list** command. See *ip prefix-list <listname> description <"text">* [on page 335](#) for more information.

Usage Examples

The following example uses the **MyList** prefix list to filter all BGP updates received from the specified peer:

```
ProCurve(config)#router bgp 65000
ProCurve(config-bgp)#neighbor 65.4.3.192
ProCurve(config-bgp-neighbor)#prefix-list MyList in
```

remote-as <as-number>

Use the **remote-as** command to specify the BGP autonomous system (AS) to which the neighbor belongs, adding an entry to the BGP neighbor table. Use the **no** form of this command to return to default settings.

Syntax Description

<as-number>	Specifies the AS number. This number must be different from the AS number of the local router (which is defined using the router bgp command). Range: 1 to 65,535. See <i>router bgp <AS number></i> on page 368 for more information.

Default Values

By default, no BGP neighbors are defined.

Usage Examples

The following example configures a remote AS number of **200** for this neighbor:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#remote-as 200
```

route-map <map-name> [in | out]

Use the **route-map** command to assign a route map to this BGP neighbor. The route map is then used to filter or modify inbound and/or outbound BGP route updates. Use the **no** form of this command to return to default settings.

Syntax Description

<map-name>	Assigns a route map to this BGP neighbor.
in	Specifies the filtering/modification of all inbound BGP route updates.
out	Specifies the filtering/modification of all outbound BGP route updates.

Default Values

By default, no route map is assigned.

Functional Notes

Before a route map can be assigned to a BGP neighbor, it must first be defined using the **route-map** command. See *route-map <map-name> [permit | deny] <sequence number>* [on page 367](#) for more information.

Usage Examples

The following example assigns a route map to this neighbor for outbound filtering:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#route-map MapName out
```

send-community standard

Use the **send-community standard** command to insert a standard BGP community attribute into all outgoing route updates for this neighbor. Use the **no** form of this command to return to default settings.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Usage Examples

The following example inserts a standard BGP community attribute into all outgoing route updates for the specified neighbor:

```
ProCurve(config)#router bgp 65000
```

```
ProCurve(config-bgp)#neighbor 65.4.3.192
```

```
ProCurve(config-bgp-neighbor)#send-community standard
```

soft-reconfiguration inbound

Use the **soft-reconfiguration inbound** command to enable this unit to store BGP updates for the specified neighbor. Use the **no** form of this command to return to default settings.

Syntax Description

No subcommands.

Default Values

By default, this command is enabled.

Functional Notes

BGP updates are stored prior to filtering, thus allowing the **clear ip bgp soft** command to be used in the absence of route refresh (RFC2918) capability. This command affects all neighbors. See *clear ip bgp [* | <as-number> | <ip address>] [in | out | soft]* [on page 35](#) for more information.

Usage Examples

The following example enables the unit to store BGP updates for the specified neighbor:

```
ProCurve(config)#router bgp 65000
```

```
ProCurve(config-bgp)#neighbor 65.4.3.192
```

```
ProCurve(config-bgp-neighbor)#soft-reconfiguration inbound
```

update-source *<interface>*

Use the **update-source** command to specify which virtual interface's IP address will be used as the source IP address for the BGP TCP connection (when connecting to this peer). Use the **no** form of this command to return to default settings.

Syntax Description

<i><interface></i>	Specifies the interface ID (e.g., loopback 1) of the virtual interface to be used as the source IP address.

Default Values

By default, the outbound interface's IP address is used for BGP updates.

Functional Notes

This is most often configured as a loopback interface that is reachable by the peer router. The peer will specify this address in its neighbor commands for this router.

Usage Examples

The following example configures the **loopback 1** interface as the source IP:

```
ProCurve(config)#router bgp 65000  
ProCurve(config-bgp)#neighbor 65.4.3.192  
ProCurve(config-bgp-neighbor)#update-source loopback 1
```

TUNNEL CONFIGURATION COMMAND SET

To activate the Tunnel Configuration mode, enter the **interface tunnel** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycyname> [on page 801](#)

backup commands [begin on page 804](#)

bandwidth <value> [on page 820](#)

crypto map <mapname> [on page 821](#)

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password> [on page 823](#)

ip commands [begin on page 826](#)

keepalive <period> <retries> [on page 840](#)

lldp receive [on page 841](#)

lldp send [management-address | port-description | system-capabilities | system-description | system-name] [on page 842](#)

lldp send-and-receive [on page 843](#)

mtu <size> [on page 844](#)

tunnel commands [begin on page 845](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic on an interface. Use the **no** form of this command to remove an access policy association.

Note	<i>Configured access policies will only be active if the ip firewall command has been entered at the Global Configuration mode prompt to enable the SROS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.</i>

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor (all access policy descriptors are case-sensitive).

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access-policy <policy name>**.

Usage Examples

The following example associates the access policy **UnTrusted** (to allow inbound traffic to the Web server) to the tunnel 1 interface:

Enable the SROS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list **InWeb**:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access policy with the tunnel 1 interface:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an IP policy class that uses a configured access list. SROS access policies are used to permit, deny, or manipulate (using **NAT**) data for each physical interface. Each ACP consists of a selector (**access list**) and an action (**allow**, **discard**, **NAT**). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet or the Web GUI.

discard list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list *<access list names>* **address** *<IP address>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list *<access list names>* **interface** *<interface>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy** *<policy name>*. The following example assigns access policy **MatchAll** to the tunnel 1 interface:

```
ProCurve(config)#interface tunnel 1
```

```
ProCurve(config-tunnel 1)#access-policy MatchAll
```

backup auto-backup

Use the **backup auto-backup** command to configure the sub-interface to automatically attempt a backup upon failure.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically attempt backup upon a failure.

Usage Examples

The following enables automatic backup on the endpoint:

```
ProCurve(config)#interface tunnel 1
```

```
ProCurve(config-tunnel 1)#backup auto-backup
```

backup auto-restore

Use the **backup auto-restore** command to configure the sub-interface to automatically discontinue backup when all network conditions are operational. Use the **no** form of this command to disable the auto-restore feature.

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically restore the primary connection when the failure condition clears.

Usage Examples

The following configures the Secure Router OS to automatically restore the primary connection when the failure condition clears:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#backup auto-restore
```

backup backup-delay <seconds>

Use the **backup backup-delay** command to configure the amount of time the router will wait after the failure condition is recognized before attempting to backup the link. Use the **no** form of this command to return to the default value.

Syntax Description

<seconds>	Specifies the delay period (in seconds) a failure must be active before the Secure Router OS will enter backup operation on the interface (valid range: 10 to 86400 seconds).

Default Values

By default, the **backup backup-delay** is set to 10 seconds.

Usage Examples

The following configures the Secure Router OS to wait 60 seconds (on an endpoint with an active alarm condition) before attempting backup operation:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#backup backup-delay 60
```

backup call-mode <role>

Use the **backup call-mode** command to specify whether the configured backup interfaces answers or originates (or a combination of both) backup calls. Use the **no** form of this command to return to the default value.

Syntax Description

<role>	Selects the role the router will take in backup of this sub-interface.
answer	Answers and backs up primary link on failure.
answer-always	Answers and backs up regardless of primary link state.
originate	Originates backup call on primary link failure.
originate-answer	Originates or answers call on primary link failure.
originate-answer-always	Originates on failure; answers and backs up always.

Default Values

By default, the **backup call-mode** is set to **originate-answer**.

Functional Notes

The majority of the configuration for tunnel backup is configured via the tunnel interface configuration commands. However, the numbers dialed are configured in the primary interface. Full sample configurations follow:

Sample config for remote router (dialing out)

```
hostname "Remote7203dl"
enable password password
!
interface eth 0/1
 ip address 192.168.1.254 255.255.255.0
 no shutdown
!
interface modem 1/3
 no shutdown
!
interface t1 1/1
 coding b8zs
 framing esf
 clock source line
 tdm-group 1 timeslots 1-24
 no shutdown
!
```

```
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  no shutdown
  bind 1 t1 1/1 1 fr 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  ip address 10.1.1.2 255.255.255.252
  backup call-mode originate
  backup number 5551111 analog
  backup number 5552222 analog
!
ip route 0.0.0.0 0.0.0.0 10.1.1.1
!
line telnet 0 4
  password password
```

Sample config for central router (dialing in)

```
hostname "Central7203dl"
enable password password
!
interface eth 0/1
  ip address 192.168.100.254 255.255.255.0
  no shutdown
!
interface modem 1/3
  no shutdown
!
interface t1 1/1
  coding b8zs
  framing esf
  clock source line
  tdm-group 1 timeslots 1-24
  no shutdown
!
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  no shutdown
  bind 1 t1 1/1 1 fr 1
!
interface fr 1.100 point-to-point
```



```
frame-relay interface-dlci 100
ip address 10.1.1.1 255.255.255.252
backup call-mode answer
backup number 555-8888 analog
!
line telnet 0 4
password password
```

Usage Examples

The following configures the Secure Router OS to answer backup calls on this endpoint but never generate calls:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#backup call-mode answer-always
```

backup connect-timeout <seconds>

Use the **backup connect-timeout** command to specify the number of seconds to wait for a connection after a call is attempted before trying to call again or dialing a different number. Recommended value is greater than 60.

Syntax Description

<seconds>	Selects the amount of time in seconds that the router will wait for a connection before attempting another call (valid range: 10 to 300).

Default Values

By default, **backup connect-timeout** is set to 60 seconds.

Usage Examples

The following configures the Secure Router OS to wait 120 seconds before retrying a failed backup call:

```
ProCurve(config)#interface tunnel 1
```

```
ProCurve(config-tunnel 1)#backup connect-timeout 120
```

backup force <state>

Use the **backup force** command to manually override the automatic backup feature. This can be used to force a link into backup to allow maintenance to be performed on the primary link without disrupting data. Use the **no** form of this command to return to the normal backup operation state.

Syntax Description

<state>	Selects the forced backup state of the sub-link.
backup	Forces backup regardless of primary link state.
primary	Forces primary link regardless of its state.

Default Values

By default, this feature is disabled.

Usage Examples

The following configures the Secure Router OS to force this endpoint into backup:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#backup force backup
```

backup maximum-retry <attempts>

Use the **backup maximum-retry** command to select the number of calls the router will make when attempting to backup a link. Use the **no** form of this command to return to the default state. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

<attempts>	Selects the number of call retries that will be made after a link failure (valid range: 0 to 15).
	Setting this value to 0 will allow unlimited retries during the time the network is failed.

Default Values

By default, **backup maximum-retry** is set to 0 attempts.

Usage Examples

The following example configures the Secure Router OS to retry a backup call four times before considering backup operation not available:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#backup maximum-retry 4
```

backup number <digits> [analog | digital-56k | digital 64k] <isdn min chan> <isdn max chan> <interface>

Use the **backup number** command to configure the phone number and the call type the router will dial upon network failure. Multiple entries can be made for an interface to allow alternate sites to be dialed. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

<digits>	Specifies the phone numbers to call when the backup is initiated.
analog	Indicates number connects to an analog modem.
digital-56k	Indicates number belongs to a digital 56 kbps per DS0 connection.
digital-64k	Indicates number belongs to a digital 64 kbps per DS0 connection.
<isdn min chan>	Specifies the minimum number of DS0s required for a digital 56 or 64 kbps connection (Range: 1 to 24).
<isdn mas chan>	Specifies the maximum number of DS0s desired for a digital 56 or 64 kbps connection (Range: 1 to 24).
<interface>	Specifies the Frame Relay sub-interface (e.g., tunnel 1) to use when originating or answering using this number.

Default Values

By default, there are no configured **backup numbers**.

Usage Examples

The following example configures the Secure Router OS to dial 704-555-1212 (digital 64 kbps connection) to initiate backup operation on this endpoint using the configured tunnel interface tunnel 1:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#backup number 7045551212 digital-64k 1 1 tunnel 1
```

backup priority <value>

Use the **backup priority** command to select the backup priority for this interface. This allows the user to establish the highest priority backup link and ensure that link will override backups attempted by lower priority links. Use the **no** form of this command to return to the default value. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

<value>	Sets the relative priority of this link (valid range: 0 to 100). A value of 100 designates the highest priority.

Default Values

By default, **backup priority** is set to 50.

Usage Examples

The following example assigns the highest priority to this endpoint:

```
ProCurve(config)#interface tunnel 16
ProCurve(config-tunnel 16)#backup priority 100
```

backup randomize-timers

Use the **backup randomize-timers** command to randomize the call timers to minimize potential contention for resources. Use the **no** form of this command to return to the default value. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

No subcommands.

Default Values

By default, the Secure Router OS does not randomize the backup call timers.

Usage Examples

The following example configures the Secure Router OS to randomize the backup timers associated with this endpoint:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#backup randomize-timers
```

backup redial-delay <seconds>

Use the **backup redial-delay** command to configure the delay after an unsuccessful call until the call will be re-tried. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode* <role> [on page 807](#).

Syntax Description

<seconds>	Specifies the delay in seconds between attempting to re-dial a failed backup attempt. Range: 10 to 3600.

Default Values

By default, **backup redial-delay** is set to 10 seconds.

Usage Examples

The following example configures a redial delay of 25 seconds on this endpoint:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#backup redial-delay 25
```


backup restore-delay <seconds>

Use the **backup restore-delay** command to configure the amount of time the router will wait after the network is restored before disconnecting the backup link and reverting to the primary. This setting is used to prevent disconnecting the backup link if the primary link is “bouncing” in and out of alarm. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

<seconds>	Specifies the number of seconds the Secure Router OS will wait (after a primary link is restored) before disconnecting backup operation. Range: 10 to 86,400.

Default Values

By default, **backup restore-delay** is set to 10 seconds.

Usage Examples

The following example configures the Secure Router OS to wait 30 seconds before disconnecting backup operation and restoring the primary connection for this endpoint:

```
ProCurve(config)#interface tunnel 1
```

```
ProCurve(config-tunnel 1)#backup restore-delay 30
```

backup schedule [day | enable-time | disable-time]

Use the **backup schedule** command to set the time of day that backup will be enabled. Use this command if backup is desired only during normal business hours and on specific days of the week. Use the **no** form of this command to disable backup (as specified). For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

day	Sets the days to allow backup (valid range: Monday through Sunday).
enable-time	Sets the time of day to enable backup. Time is entered in 24-hour format (00:00).
disable-time	Sets the time of day to disable backup.

Default Values

By default, backup is enabled for all days and times if the **backup auto-backup** command has been issued and the backup schedule has not been entered.

Usage Examples

The following example enables backup Monday through Friday 8:00 am to 7:00 pm:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#backup schedule enable-time 08:00
ProCurve(config-tunnel 1)#backup schedule disable-time 19:00
ProCurve(config-tunnel 1)#no backup schedule day Saturday
ProCurve(config-tunnel 1)#no backup schedule day Sunday
```

backup shutdown

Use the **backup shutdown** command to deactivate all backup functionality in the unit. Backup configuration parameters are kept intact, but the unit will not initiate (or respond) to backup sequences in the event of a network outage. Use the **no** form of this command to reactivate the backup interface. For more detailed information on tunnel backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 807](#).

Syntax Description

No subcommands.

Default Values

By default, all Secure Router OS interfaces are disabled.

Usage Examples

The following example deactivates the configured backup interface:

```
ProCurve(config)#interface tunnel 16  
ProCurve(config-tunnel 16)#backup shutdown
```

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Specifies bandwidth in kbps.

Default Values

To view default values, use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the tunnel 1 interface to 10 Mbps:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#bandwidth 10000
```

crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the technical support note **Configuring VPN** located on the ProCurve SROS documentation CD provided with your unit.*

Syntax Description

<mapname> Enter the crypto map name that you wish to assign to the interface.

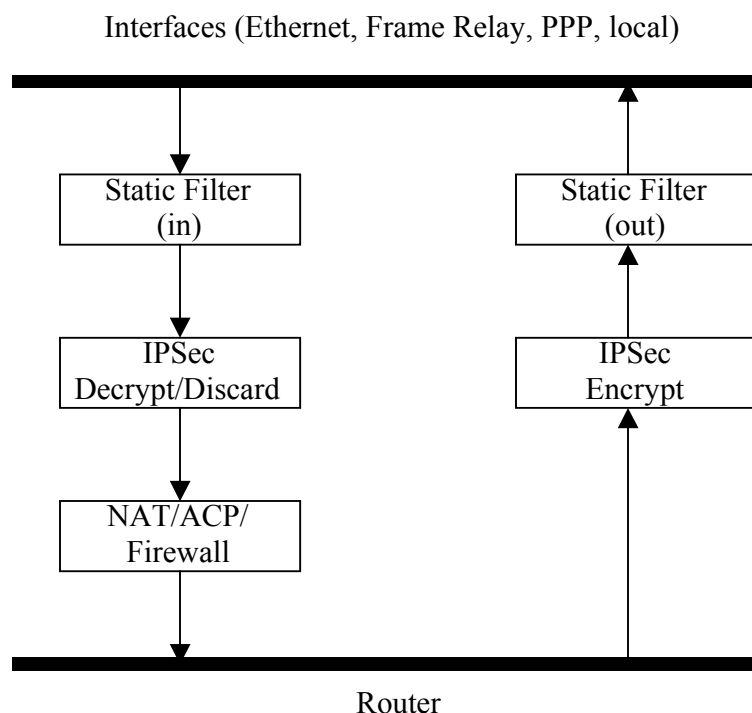
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following notes in mind.

When defining the policy-class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the un-encrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical Secure Router OS data-flow logic.



As shown in the previous diagram, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access-group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only un-encrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy-class on an interface. When specifying the ACLs for a crypto map, the source information is the private local-side, un-encrypted source of the data. The destination information will be the far-end, un-encrypted destination of the data. However, ACLs for a policy-class work in reverse. The source information for the ACL in a policy-class is the far-end. The destination information is the local-side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the tunnel interface:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#crypto map MyMap
```

**dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname>
<username> <password>**

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

Refer to Functional Notes below for argument descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A web-based interface provides two levels of control over your domain, catering to average or advanced users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

When to Use Dynamic DNS

If your IP address does not change often (or at all) but you still want an easy name to remember it by (without having to purchase your own domain name), Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com), you need Custom DNS service, which also provides full dynamic and static IP address support.

Dynamic DNS service can be extremely helpful for site-to-site VPN connections using dynamic IP addressing. Instead of creating a tunnel to the IP-address, a tunnel to the DNS-name is established.

Usage Examples

The following example sets the dynamic-dns to **dyndns-custom** with hostname **host**, username **user**, and password **pass**:

```
ProCurve(config)#interface tunnel 1
```

```
ProCurve(config-tunnel 1)#dynamic-dns dyndns-custom host user pass
```


ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<listname>	Assigns an IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the unit to only allow Telnet traffic (as defined in the user-configured **TelnetOnly** IP access list) into the tunnel interface:

```
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#interface tunnel 1
ProCurve(config-tunnel 1)#ip access-group TelnetOnly in
```

ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional. Configures a secondary IP address for the specified interface.

Default Values

By default, there are no assigned IP addresses.

Usage Examples

The following example configures an IP address of **192.22.72.101/30**:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#ip address 192.22.72.101 255.255.255.252
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note *The **ip helper-address** command must be used in conjunction with the **ip forward-protocol** command to configure the SROS to forward UDP broadcast packets. Refer to [ip forward-protocol udp <port number>](#) on page 318 for more information.*

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign helper address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is User Datagram Protocol (UDP).
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain  
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#ip helper-address 192.33.5.99
```

ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

immediate-leave	Specifies that if only one host (or IGMP snooping switch) is connected to the interface, when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	Controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave-group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer, if no receiver responds, the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65,535 ms. Default: 1000 ms.
querier-timeout <seconds>	Specifies the number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60 to 300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Specifies the interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65,535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Specifies the maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically-connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip igmp last-member-query-interval 200
```

ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface, and to place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding. Refer to *ip mcast-stub helper-address <ip address>* [on page 327](#) and *ip mcast-stub upstream* [on page 832](#) for more information.

Usage Examples

The following example enables multicast forwarding and IGMP on the interface:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip mcast-stub downstream
```

ip mcast-stub helper-enable

Use the **ip mcast-stub helper-enable** command to assign the **ip mcast-stub helper-address** as the IGMP proxy. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address**, **ip mcast-stub upstream**, and **ip mcast-stub downstream** commands. When enabled, the interface becomes a helper forwarding interface. The IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the unit to perform as an IGMP proxy. Refer to *ip mcast-stub helper-address <ip address>* [on page 389](#), *ip mcast-stub downstream* [on page 830](#), and *ip mcast-stub upstream* [on page 832](#) for more information.

Usage Examples

The following example sets the helper address as the IGMP proxy:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip mcast-stub helper-enable
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface. Refer to *ip mcast-stub helper-address <ip address>* [on page 327](#) and *ip mcast-stub downstream* [on page 830](#) for more information.

Usage Examples

The following example enables multicast forwarding on the interface:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip mcast-stub upstream
```


ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key	Specifies a simple-text authentication password to be used by other routers using <i><password></i> the OSPF simple password authentication.
cost <i><value></i>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65,535.
dead-interval <i><seconds></i>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, neighboring devices will determine that the device is down. Range: 0 to 32,767.
hello-interval <i><seconds></i>	Specifies the interval between hello packets sent on the interface. Range: 0 to 32,767.
message-digest-key <i><keyid></i> md5 <i><key></i>	Configures OSPF Message Digest 5 (MD5) authentication (16-byte max) keys.
priority <i><value></i>	Set the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <i><seconds></i>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32,767.
transmit-delay <i><seconds></i>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32,767.

Default Values

retransmit-interval <i><seconds></i>	5 seconds
transmit-delay <i><seconds></i>	1 second
hello-interval <i><seconds></i>	10 seconds: Ethernet, point-to-point, Frame Relay, Tunnel, and PPP
dead-interval <i><seconds></i>	40 seconds

Usage Example

The following example sets the maximum number of seconds allowed between hello packets to 25,000:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip ospf dead-interval 25000
```

ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Optional. Selects message-digest authentication type.
null	Optional. Specifies that no authentication is used.

Default Values

By default, this is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the tunnel interface:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#ip ospf authentication null
```

ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Sets the network type for broadcast.
point-to-point	Sets the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP, Frame Relay, and tunnel default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#ip ospf network broadcast
```

ip proxy-arp <ip address> <subnet mask>

Use the **ip proxy-arp** command to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<ip address>	Defines the proxy ARP IP address in dotted decimal notation (for example: 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, proxy-arp is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the SROS will respond to all proxy ARP requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the tunnel interface:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip proxy-arp
```

ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Only accept received RIP version 1 packets on the interface.
2	Only accept received RIP version 2 packets on the interface.

Default Values

By default, all interfaces implement RIP version 1 (the default value for the version command).

Functional Notes

Use the **ip rip receive version** command to specify a RIP version that overrides the **version** (in the Router RIP) configuration. See *version [1 | 2]* [on page 969](#) for more information.

The SROS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the tunnel interface to accept only RIP version 2 packets:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip rip receive version 2
```

ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Only transmits RIP version 1 packets on the interface.
2	Only transmits RIP version 2 packets on the interface.

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** command to specify a RIP version that overrides the **version** (in the Router RIP) configuration. See *version [1 | 2]* [on page 969](#) for more information.

The SROS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the tunnel interface to transmit only RIP version 2 packets:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip rip send version 2
```

ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note *Using Network Address Translation (NAT) or the SROS firewall capabilities on an interface requires process switching mode (using the **no ip route-cache** command).*

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on the tunnel interface:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#ip route-cache
```

keepalive *<period>* *<retries>*

Use the **keepalive** command to periodically send keepalive packets to verify the integrity of the tunnel from end to end. Use the **no** form of this command to disable keepalives.

Syntax Description

<i><period></i>	Defines the time interval (in seconds) between transmitted keepalive packets (valid range: 1 to 32,767 seconds).
<i><retries></i>	Defines the number of times to retry after failed keepalives before determining that the tunnel endpoint is down (valid range: 1 to 255 times).

Default Values

By default, keepalives are disabled. When enabled, the keepalive period defaults to 10 seconds and the retry count defaults to 3 times.

Functional Notes

Keepalives do not have to be configured on both ends of the tunnel in order to work. A tunnel is not aware of incoming keepalive packets.

Usage Examples

The following example enables **keepalive** with a period of 30 seconds and a retry count of 5 times:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#keepalive 30 5
```


lldp receive

Use the **lldp receive** command to allow LLDP packets to be received on this interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to send and receive LLDP packets.

Usage Examples

The following example configures the tunnel interface to receive LLDP packets:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#lldp receive
```

lldp send [management-address | port-description | system-capabilities | system-description | system-name]

Use the **lldp send** command to configure this interface to transmit LLDP packets or to control the types of information contained in the LLDP packets transmitted by this interface.

Syntax Description

management-address	Enables transmission of management address information on this interface.
port-description	Enables transmission of port description information on this interface.
system-capabilities	Enables transmission of this device's system capabilities on this interface.
system-description	Enables transmission of this device's system description on this interface.
system-name	Enables transmission of this device's system name on this interface.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the tunnel interface to transmit LLDP packets containing all enabled information types:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#lldp send
```

lldp send-and-receive

Use the **lldp send-and-receive** command to configure this interface to transmit and receive LLDP packets.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send-and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the tunnel interface (tunnel 16) to transmit and receive LLDP packets containing all information types:

```
ProCurve(config)#interface tunnel 16  
ProCurve(config-tunnel 16)#lldp send-and-receive
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit (MTU) size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size> Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:

ATM interfaces	64 to 1520
Ethernet interfaces	64 to 1500
FDL interfaces	64 to 256
HDLC interfaces	64 to 1520
Loopback interfaces	64 to 1500
Tunnel interfaces	64 to 18,190
Virtual Frame Relay sub-interfaces	64 to 1520
Virtual PPP interfaces	64 to 1500

Default Values

<size> The default values for the various interfaces are listed below:

ATM interfaces	1500
Ethernet interfaces	1500
FDL interfaces	256
HDLC interfaces	1500
Loopback interfaces	1500
Tunnel interfaces	1500
Virtual Frame Relay sub-interfaces	1500
Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the Tunnel interface:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#mtu 1200
```

tunnel checksum

Use the **tunnel checksum** command to verify the checksum of incoming Generic Routing Encapsulation (GRE) packets and to include a checksum on outgoing packets. Use the **no** form of this command to disable checksum.

Syntax Description

No subcommands.

Default Values

By default, **tunnel checksum** is disabled.

Functional Notes

Both ends of the tunnel must have **tunnel checksum** enabled in order for a meaningful configuration. When both endpoints have **tunnel checksum** enabled, a packet with an incorrect checksum will be dropped. If the endpoints differ in their checksum configuration, all packets will still flow without any checksum verification.

Usage Examples

The following example enables checksum on the tunnel 1 interface:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#tunnel checksum
```

Technology Review

When enabled, the **tunnel checksum** will be calculated for each outgoing GRE packet with the result stored in the GRE header. The checksum present bit will also be set in the header.

tunnel destination <ip address>

Use the **tunnel destination** command to specify the IP address to use as the destination address for all packets transmitted on this interface. Use the **no** form of this command to clear the **tunnel destination** address.

Syntax Description

<ip address>	Specifies the IP address in dotted decimal notation to use as the destination address for all packets transmitted on this interface (for example: 192.22.73.101).

Default Values

By default, no tunnel destinations are defined.

Functional Notes

Until a tunnel interface has a destination IP address defined, it is not operational.

The tunnel destination IP address will be the value put into the destination field of the outer IP header after GRE encapsulation of the original packet. A route must be defined for the destination address. Be certain there are no recursive routes by ensuring that a tunnel's destination address will be routed out a physical interface. There is a possibility of creating a routing loop when tunnel interface traffic gets routed back to the same tunnel interface or to another tunnel interface, which in turn, does not have a route out a physical interface. In either case, the tunnel will go down for a period of one minute, after which it will come back up to determine if the recursive routes have been resolved. This allows time for routing protocols to converge on a valid route. If a static route has caused the recursive routing loop, the tunnel status may oscillate until the route is changed.

Usage Examples

The following example sets the tunnel destination IP address to **192.22.73.101**:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#tunnel destination 192.22.73.101
```

tunnel key <value>

Use the **tunnel key** command to specify a value shared by both endpoints of the tunnel that will provide minimal security and delineate between tunnels with the same source and destination addresses. Use the **no** form of this command to disable the key.

Syntax Description

<value>	Defines the key value for this tunnel (valid range: 1 to 4,294,967,294).

Default Values

By default, a key is not configured.

Functional Notes

A matching key value must be defined on both endpoints of the tunnel or packets will be discarded.

Usage Examples

The following example sets the key on a tunnel interface to a value of 1234:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#tunnel key 1234
```

Technology Review

When enabled, the key will be stored in the GRE header and the key present bit will be set.

tunnel mode gre

Use the **tunnel mode gre** command to encapsulate traffic destined for the tunnel interface in a Generic Routing Encapsulation (GRE) header. Use the **no** form of this command to set the tunnel to its default mode.

Syntax Description

No subcommands.

Default Values

By default, the tunnel interface will be configured for GRE mode.

Functional Notes

GRE is currently the only allowed mode for tunnel interface operation.

Usage Examples

The following example configures the tunnel interface for GRE mode:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#tunnel mode gre
```

tunnel sequence-datagrams

Use the **tunnel sequence-datagrams** command to enable sequence number checking on incoming Generic Routing Encapsulation (GRE) packets, to drop packets arriving out of order, and to include a sequence number in outgoing packets. Use the **no** form of this command to disable sequence number checking.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

Both ends of the tunnel must have sequence numbering enabled. When both endpoints have sequence numbering enabled, a packet arriving with a sequence number less than the current expected value will be dropped. If the endpoints differ in their sequence numbering configuration, all packets will still flow without any sequence number verification. Be careful enabling sequence number verification on a tunnel. The tunnel can easily become out of sequence due to network conditions outside of the tunnel endpoints. It may be difficult to establish a successful traffic flow after an out of sequence condition occurs.

Technology Review

When enabled, the next valid sequence number will be placed in the GRE header of each outgoing packet, and the sequence number present bit will be set.

Usage Examples

The following example enables sequence number processing on the tunnel interface:

```
ProCurve(config)#interface tunnel 1  
ProCurve(config-tunnel 1)#tunnel sequence-datagrams
```

tunnel source [*<ip address>* | *<interface>*]

Use the **tunnel source** command to specify the IP address or name of a physical interface to use as the source address for all packets transmitted on this interface. Use the **no** form of this command to clear the tunnel source address.

Syntax Description

<i><ip address></i>	Specifies the IP address in dotted decimal notation to use as the source address for all packets transmitted on this interface (for example: 192.22.73.101).
<i><interface></i>	Specifies the interface (in the format type <i><slot/port></i>) that contains the IP address to use as the source address for all packets transmitted on this interface.

Default Values

By default, a tunnel source is not defined.

Functional Notes

Until a tunnel interface has a source IP address defined and the physical interface used as the source is operational, the tunnel is not operational.

The tunnel source IP address will be the value put into the source field of the outer IP header after GRE encapsulation of the original packet.

Usage Examples

The following example sets the tunnel source IP address to **192.22.73.101**:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#tunnel source 192.22.73.101
```

The following example sets the tunnel source IP address to the address of the Ethernet interface labeled 0/1:

```
ProCurve(config)#interface tunnel 1
ProCurve(config-tunnel 1)#tunnel source eth 0/1
```

PPP INTERFACE CONFIGURATION COMMAND SET

To activate the PPP Interface Configuration mode, enter the **interface ppp** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)
description <text> [on page 1000](#)
do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)
shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycyname> [on page 853](#)
alias link <"text"> [on page 856](#)
backup commands [begin on page 857](#)
bandwidth <value> [on page 873](#)
bridge-group <group#> [on page 874](#)
crypto map <mapname> [on page 875](#)
dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password> [on page 877](#)
fair-queue <threshold> [on page 879](#)
hold-queue <queue size> out [on page 880](#)
ip commands [begin on page 883](#)
keepalive <seconds> [on page 899](#)
lldp receive [on page 900](#)
lldp send [management-address | port-description | system-capabilities | system-description | system-name | and-receive] [on page 901](#)
mtu <size> [on page 902](#)

peer default ip address <address> [on page 903](#)

ppp commands [begin on page 904](#)

pppoe ac-name <name> [on page 911](#)

pppoe service-name <name> [on page 912](#)

qos-policy out <mapname> [on page 913](#)

snmp trap link-status [on page 914](#)

username <username> password <password> [on page 915](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic to an interface. Use the **no** form of this command to remove an access policy association.

Note	<i>Configured access policies will only be active if the ip firewall command has been entered at the Global Configuration mode prompt to enable the SROS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.</i>

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor. (All access policies are case-sensitive.)

Note	<i>All access policy descriptors are case-sensitive.</i>

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**.

Usage Examples

The following example associates the access policy **UnTrusted** (to allow inbound traffic to the Web server) to the virtual PPP interface:

Enable the Secure Router OS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list **InWeb**:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access list with the PPP virtual interface (labeled 1):

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host <A.B.C.D>** to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the **<A.B.C.D> <wildcard>** format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Note

*The command **permit <A.B.C.D>** will also be assumed to mean **permit host <A.B.C.D>**.*

Step 3:

Create an access policy that uses a configured access list. Secure Router OS access policies are used to permit, deny, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> policy <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> policy <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list <access list names> self

All packets passed by the access list(s) entered and destined for any local interface on the unit will be

permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet or the Web GUI.

discard list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list *<access list names>* **address** *<IP address>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network. This function is also known as “many-to-one NAT.”

nat source list *<access list names>* **interface** *<interface>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network. This function is also known as “many-to-one NAT.”

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network. This function is also known as “port forwarding.”

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the Interface Configuration mode for the desired interface and enter **access policy** *<policy name>*. The following example assigns access policy **MatchAll** to the virtual PPP interface labeled 1:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#access-policy MatchAll
```

alias link<*text*>

Each configured PPP interface (when referenced using SNMP) contains a link (physical port) and a bundle (group of links). RFC 1471 (for Link Connection Protocol) provides an interface table to manage lists of bundles and associated links. The **alias link** command provides the management station an identifying description for each link (PPP physical).

Syntax Description

< <i>text</i> >	Describes the interface (for SNMP) by alphanumeric character string (must be encased in quotation marks).

Default Values

By default, the PPP identification string appears as empty quotes (" ").

Functional Notes

The **alias link** string should be used to uniquely identify a PPP link. Enter a string that clearly identifies the link.

Usage Examples

The following example defines a unique character string for the virtual PPP interface (1):

```
ProCurve(config)#ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#alias link "PPP_link_1"
```

Technology Review

Please refer to RFC 1990 for a more detailed discussion on PPP links and bundles.

backup auto-backup

Use the **backup auto-backup** command to configure the PPP interface to automatically attempt a backup upon failure. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically attempt backup upon a failure.

Usage Examples

The following example enables automatic backup on the endpoint:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#backup auto-backup
```

backup auto-restore

Use the **backup auto-restore** command to configure the interface to automatically discontinue backup when all network conditions are operational. Use the **no** form of this command to disable the auto-restore feature. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

No subcommands.

Default Values

By default, all backup endpoints will automatically restore the primary connection when the failure condition clears.

Usage Examples

The following example configures the Secure Router OS to automatically restore the primary connection when the failure condition clears:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#backup auto-restore
```

backup backup-delay <seconds>

Use the **backup backup-delay** command to configure the amount of time the router will wait after the failure condition is recognized before attempting to backup the link. Use the **no** form of this command to return to the default value. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<seconds>	Specifies the delay period (in seconds) a failure must be active before the Secure Router OS will enter backup operation on the interface. Range: 10 to 86400 seconds.

Default Values

By default, the **backup-delay** period is set to 10 seconds.

Usage Examples

The following example configures the Secure Router OS to wait 60 seconds (on an endpoint with an active alarm condition) before attempting backup operation:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#backup backup-delay 60
```

backup call-mode <role>

Use the **backup call-mode** command to specify whether the configured backup interfaces answers or originates (or a combination of both) backup calls. Use the **no** form of this command to return to the default value.

Syntax Description

<role>	Selects the role the router will take in backup of this interface.
answer	Answers and backs up primary link on failure.
answer-always	Answers and backs up regardless of primary link state.
originate	Originates backup call on primary link failure.
originate-answer	Originates or answers call on primary link failure.
originate-answer-always	Originates on failure; answers and backs up always.

Default Values

By default, **backup call-mode** is set to **originate-answer**.

Functional Notes

The majority of the configuration for PPP backup is configured in the PPP interface. However, the numbers dialed are configured in the primary interface. Full sample configurations follow:

Sample configuration for remote router (dialing out)

```
hostname "Remote7203dl"
enable password password
!
interface eth 0/1
  ip address 192.168.1.254 255.255.255.0
  no shutdown
!
interface modem 1/3
  no shutdown
!
interface t1 1/1
  coding b8zs
  framing esf
  clock source line
  tdm-group 1 timeslots 1-24
  no shutdown
!
```

```
interface ppp 1
    ip address 10.1.1.2 255.255.255.252
    backup call-mode originate
    backup number 5551111 analog ppp 2
    bind 1 t1 1/1 1 ppp 1
!
interface ppp 2
    description connected to corp for backup
    ip address 10.10.10.2 255.255.255.252
    ppp authentication pap
    ppp pap sent-username joe password pswrd
!
ip route 0.0.0.0 0.0.0.0 10.1.1.1
!
line telnet 0 4
    password password
```

Sample configuration for central router (dialing in)

```
hostname "Central7203dl"
enable password password
!
interface eth 0/1
    ip address 192.168.100.254 255.255.255.0
    no shutdown
!
interface modem 1/3
    no shutdown
!
interface t1 1/1
    coding b8zs
    framing esf
    clock source line
    tdm-group 1 timeslots 1-24
    no shutdown
!
interface ppp 1
    no shutdown
    bind 1 t1 1/1 1 ppp 1
    ip address 10.1.1.1 255.255.255.252
    backup call-mode answer
    backup number 555-8888 analog ppp 2
!
```

```
interface ppp 2
  description connection for remote 7203dl dialin for backup
  ip address 10.10.10.1 255.255.255.252
  ppp authentication pap
  username joe password pswrd
!
line telnet 0 4
  password password
```

Usage Examples

The following example configures the Secure Router OS to answer backup calls on this endpoint but never generate calls:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#backup call-mode answer-always
```

Technology Review

This technology review provides information regarding specific backup router behavior (i.e., when the router will perform backup, where in the configuration the Secure Router OS accesses specific routing information, etc.):

Dialing Out

1. The Secure Router OS determines to place an outbound call when either the Layer 1 or Layer 2 has a failure.
2. When placing outbound calls, the Secure Router OS matches the number dialed to a PPP interface.
3. When placing the call, the Secure Router OS uses the configuration of the related PPP interface for authentication and IP negotiation.
4. If the call fails to connect on the first number dialed, the Secure Router OS places a call to the second number if configured. The second number to be dialed references a separate PPP interface.

Dialing In

1. The Secure Router OS receives an inbound call on a physical interface.
2. Caller ID is used to match the **backup number** command to the configured PPP interface.
3. If a match is found, the call connects and the Secure Router OS pulls down the primary connection if it is not already in a down state.
4. If no match is found from Caller ID, the call is terminated.

backup connect-timeout <seconds>

Use the **backup connect-timeout** command to specify the number of seconds to wait for a connection after a call is attempted before trying to call again or dialing a different number. It is recommended this number be greater than 60. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<seconds>	Selects the amount of time (in seconds) that the router will wait for a connection before attempting another call (valid range: 10 to 300).

Default Values

By default, the **backup connect-timeout** period is set to 60 seconds.

Usage Examples

The following example configures the Secure Router OS to wait 120 seconds before retrying a failed backup call:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#backup connect-timeout 120
```

backup force <state>

Use the **backup force** command to manually override the automatic backup feature. This can be used to force a link into backup to allow maintenance to be performed on the primary link without disrupting data. Use the **no** form of this command to return to the normal backup operation state. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<state>	Selects the forced backup state of the link.
backup	Forces backup regardless of primary link state.
primary	Forces primary link regardless of its state.

Default Values

By default, this feature is disabled.

Usage Examples

The following example configures the Secure Router OS to force this interface into backup:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#backup force backup
```


backup maximum-retry <attempts>

Use the **backup maximum-retry** command to select the number of calls the router will make when attempting to backup a link. Use the **no** form of this command to return to the default state. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<attempts>	Selects the number of call retries that will be made after a link failure (valid range: 0 to 15).

Setting this value to 0 will allow unlimited retries during the time the network is failed.

Default Values

By default, **backup maximum-retry** is set to 0 attempts.

Usage Examples

The following example configures the Secure Router OS to retry a backup call four times before considering backup operation not available:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#backup maximum-retry 4
```

backup number <digits> [**analog** | **digital-56k** | **digital 64k**] <isdn min chan> <isdn max chan> <interface>

Use the **backup number** command to configure the phone number and the call type the router will dial upon network failure. Multiple entries can be made for an interface to allow alternate sites to be dialed. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode* <role> [on page 860](#).

Syntax Description

<digits>	Specifies the phone numbers to call when the backup is initiated.
analog	Indicates number connects to an analog modem.
digital-56k	Indicates number belongs to a digital 56 kbps per DS0 connection.
digital-64k	Indicates number belongs to a digital 64 kbps per DS0 connection.
<isdn min chan>	Specifies the minimum number of DS0s required for a digital 56 or 64 kbps connection (Range: 1 to 24).
<isdn mas chan>	Specifies the maximum number of DS0s desired for a digital 56 or 64 kbps connection (Range: 1 to 24).
<interface>	Specifies the Frame Relay sub-interface (e.g., ppp 1) to use when originating or answering using this number.

Default Values

By default, there are no configured **backup numbers**.

Usage Examples

The following example configures the Secure Router OS to dial 704-555-1212 (digital 64 kbps connection) to initiate backup operation on this endpoint using the configured PPP interface ppp 1:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#backup number 7045551212 digital-64k 1 1 ppp 1
```

backup priority <value>

Use the **backup priority** command to select the backup priority for this interface. This allows the user to establish the highest priority backup link and ensure that link will override backups attempted by lower priority links. Use the **no** form of this command to return to the default value. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<value>	Sets the relative priority of this link (valid range: 0 to 100). A value of 100 designates the highest priority.

Default Values

By default, **backup priority** is set to 50.

Usage Examples

The following example assigns the highest priority to this endpoint:

```
ProCurve(config)#interface ppp 16
ProCurve(config-ppp 16)#backup priority 100
```

backup randomize-timers

Use the **backup randomize-timers** command to randomize the call timers to minimize potential contention for resources. Use the **no** form of this command to return to the default value. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

No subcommands.

Default Values

By default, the Secure Router OS does not randomize the backup call timers.

Usage Examples

The following example configures the Secure Router OS to randomize the backup timers associated with this endpoint:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#backup randomize-timers
```

backup redial-delay <seconds>

Use the **backup redial-delay** command to configure the delay after an unsuccessful call until the call will be re-tried. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<seconds>	Specifies the delay in seconds between attempting to re-dial a failed backup attempt. Range: 10 to 3600.

Default Values

By default, **backup redial-delay** is set to 10 seconds.

Usage Examples

The following example configures a redial delay of 25 seconds on this endpoint:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#backup redial-delay 25
```

backup restore-delay <seconds>

Use the **backup restore-delay** command to configure the amount of time the router will wait after the network is restored before disconnecting the backup link and reverting to the primary. This setting is used to prevent disconnecting the backup link if the primary link is “bouncing” in and out of alarm. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

<seconds>	Specifies the number of seconds the Secure Router OS will wait (after a primary link is restored) before disconnecting backup operation. Range: 10 to 86,400.

Default Values

By default, **backup restore-delay** is set to 10 seconds.

Usage Examples

The following example configures the Secure Router OS to wait 30 seconds before disconnecting backup operation and restoring the primary connection for this endpoint:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#backup restore-delay 30
```

backup schedule [day | enable-time | disable-time]

Use the **backup schedule** command to set the time of day that backup will be enabled. Use this command if backup is desired only during normal business hours and on specific days of the week. Use the **no** form of this command to disable backup (as specified). For more detailed information on Frame Relay backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

day	Sets the days to allow backup (valid range: Monday through Sunday).
enable-time	Sets the time of day to enable backup. Time is entered in 24-hour format (00:00).
disable-time	Sets the time of day to disable backup.

Default Values

By default, backup is enabled for all days and times if the **backup auto-backup** command has been issued and the backup schedule has not been entered.

Usage Examples

The following example enables backup Monday through Friday 8:00 am to 7:00 pm:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#backup schedule enable-time 08:00
ProCurve(config-ppp 1)#backup schedule disable-time 19:00
ProCurve(config-ppp 1)#no backup schedule day Saturday
ProCurve(config-ppp 1)#no backup schedule day Sunday
```

backup shutdown

Use the **backup shutdown** command to deactivate all backup functionality in the unit. Backup configuration parameters are kept intact, but the unit will not initiate (or respond) to backup sequences in the event of a network outage. Use the **no** form of this command to reactivate the backup interface. For more detailed information on PPP backup functionality, refer to the *Functional Notes* and *Technology Review* sections of the command *backup call-mode <role>* [on page 860](#).

Syntax Description

No subcommands.

Default Values

By default, all Secure Router OS interfaces are disabled.

Usage Examples

The following example deactivates the configured backup interface:

```
ProCurve(config)#interface ppp 16  
ProCurve(config-ppp 16)#backup shutdown
```


bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Specifies the bandwidth value in kbps.

Default Values

To view default values, use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the PPP interface to 10 Mbps:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#bandwidth 10000
```

bridge-group <group#>

Use the **bridge-group** command to assign an interface to the specified bridge group. This command is supported on all Ethernet interfaces, PPP virtual interfaces, and Frame Relay virtual sub-interfaces.

Syntax Description

<group#>	Assigns a bridge group number (range: 1 to 255).

Default Values

By default, there are no configured bridge groups.

Functional Notes

A bridged network can provide excellent traffic management to reduce collisions and limit the amount of bandwidth wasted with unnecessary transmissions when routing is not necessary. Any two interfaces can be bridged (Ethernet to T1 bridge, Ethernet to Frame Relay sub-interface, etc.).

Usage Examples

The following example assigns the PPP interface to bridge-group 1:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#bridge-group 1
```

crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<mapname> Assigns a crypto map name to the interface.

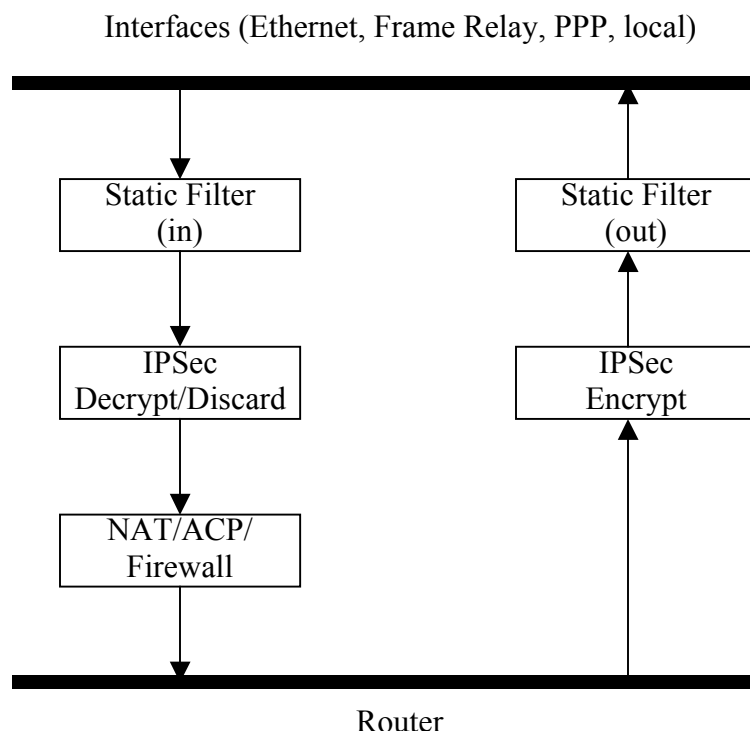
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following notes in mind.

When defining the policy class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the unencrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical Secure Router OS data-flow logic.



As shown in the previous diagram, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only unencrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy class on an interface. When specifying the ACLs for a crypto map, the source information is the private local-side, unencrypted source of the data. The destination information will be the far end, unencrypted destination of the data. However, ACLs for a policy class work in reverse. The source information for the ACL in a policy class is the far end. The destination information is the local side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the PPP interface:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#crypto map MyMap
```

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password>

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

Refer to *Functional Notes*, below, for argument descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A Web-based interface provides two levels of control over your domain, catering to average or power users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs, and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service, in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

When to Use Dynamic DNS

If your IP address does not change often (or at all) but you still want an easy name to remember it by (without having to purchase your own domain name), Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com), you need Custom DNS service, which also provides full dynamic and static IP address support.

Dynamic DNS service can be extremely helpful for site-to-site VPN connections using dynamic IP addressing. Instead of creating a tunnel to the IP-address, a tunnel to the DNS-name is established.

Usage Examples

The following example sets the **dynamic-dns** to **dyndns-custom** with hostname **host**, username **user**, and password **pass**:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#dynamic-dns dyndns-custom host user pass
```

fair-queue <threshold>

Use the **fair-queue** command to enable weighted fair queuing (WFQ) on an interface. Use the **no** form of this command to disable WFQ and enable FIFO queueing for an interface. WFQ is enabled by default for WAN interfaces.

Syntax Description

<threshold>	Optional. Specifies the maximum number of packets that can be present in each conversation sub-queue. Packets received for a conversation after this limit is reached are discarded. Range: 16 to 512 packets.

Default Values

By default, **fair-queue** is enabled with a threshold of 64 packets.

Usage Examples

The following example enables WFQ on the interface with a threshold set at 100 packets:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#fair-queue 100
```

hold-queue <queue size> out

Use the **hold-queue out** command to change the overall size of an interface's WAN output queue.

Syntax Description

<queue size>	Specifies the total number of packets the output queue can contain before packets are dropped. Range 16 to 1000.

Default Values

The default queue size for WFQ is 400. The default queue size for PPP FIFO and Frame Relay round-robin is 200.

Usage Examples

The following example sets the overall output queue size to 700:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#hold-queue 700 out
```


ip address dhcp

Use the **ip address dhcp** command to use Dynamic Host Configuration Protocol (DHCP) to obtain an address on the interface. Use the **no** form of this command to remove a configured IP address (using DHCP) and disable DHCP operation on the interface.

Syntax Description

No subcommands.

Default Values

No default is necessary for this command.

Usage Examples

The following example enables DHCP operation on the PPP sub-interface 1.1:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip address dhcp
```

ip address negotiated [no-default]

Use the **ip address negotiated** command to allow the interface to negotiate (i.e., be assigned) an IP address from the far end PPP connection. Use the **no** form of this command to disable the negotiation for an IP address.

Syntax Description

no-default	Optional. Prevents the insertion of a default route. Some systems already have a default route configured and need a static route to the PPP interface to function correctly.

Default Values

By default, the interface is assigned an address with the **ip address** *<address><mask>* command.

Usage Examples

The following example enables the PPP interface to negotiate an IP address from the far end connection:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip address negotiated
```

The following example enables the PPP interface to negotiate an IP address from the far end connection without inserting a default route:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip address negotiated no-default
```

ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<listname>	Indicates assigned IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the router to only allow Telnet traffic into the PPP interface:

```
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#interface ppp 1
ProCurve(config-ppp 1)#ip access-group TelnetOnly in
```

ip address negotiated

Use the **ip address negotiated** command to allow the interface to negotiate (i.e., be assigned) an IP address from the far end PPP connection. Use the **no** form of this command to disable the negotiation for an IP address

Syntax Description

No subcommands.

Default Values

By default, the interface is assigned an address with the **ip address** *<address><mask>* command.

Usage Examples

The following example enables the PPP interface to negotiate an IP address from the far end connection:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip address negotiated
```

ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface. Use the optional **secondary** keyword to define a secondary IP address. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional. Configures a secondary IP address for the specified interface.

Default Values

By default, there are no assigned IP addresses.

Functional Notes

Use secondary IP addresses to allow dual subnets on a single interface (when you need more IP addresses than the primary subnet can provide). When using secondary IP addresses, avoid routing loops by verifying that all devices on the network segment are configured with secondary IP addresses on the secondary subnet.

Usage Examples

The following example configures a secondary IP address of **192.22.72.101/30**:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#ip address 192.22.72.101 255.255.255.252 secondary
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note *The **ip helper** command must be used in conjunction with the **ip forward-protocol** command to configure the Secure Router OS forward UDP broadcast packets. Refer to [ip forward-protocol udp <port number>](#) on page 318 for more information.*

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign helper address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is User Datagram Protocol (UDP).
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip helper-address 192.33.5.99
```

ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

immediate-leave	If only one host (or IGMP snooping switch) is connected to the interface, when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	Controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer, if no receiver responds, the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65535 ms. Default: 1000 ms.
querier-timeout <seconds>	Specifies the number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60 to 300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Specifies the interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Specifies the maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically-connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip igmp last-member-query-interval 200
```

ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface and place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding. Refer to *ip mcast-stub helper-address <ip address>* [on page 327](#) and *ip mcast-stub upstream* [on page 890](#) for more information.

Usage Examples

The following example enables multicast forwarding and IGMP on the interface:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip mcast-stub downstream
```


ip mcast-stub helper-enable

Use the **ip mcast-stub helper-enable** command to assign the **ip mcast-stub helper-address** as the IGMP proxy. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address**, **ip mcast-stub upstream**, and **ip mcast-stub downstream** commands. When enabled, the interface becomes a helper forwarding interface. The IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the unit to perform as an IGMP proxy.

Usage Examples

The following example sets the helper address as the IGMP proxy:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip mcast-stub helper-enable
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface.

Usage Examples

The following example enables multicast forwarding on the interface:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip mcast-stub upstream
```

ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key <password>	Assigns a simple-text authentication password to be used by other routers using the OSPF simple password authentication.
cost <value>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65,535.
dead-interval <seconds>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, neighboring devices will determine that the device is down. Range: 0 to 32,767.
hello-interval <seconds>	Specifies the interval between hello packets sent on the interface. Range: 0 to 32,767.
message-digest-key <keyid> md5 <key>	Configure OSPF Message Digest 5 (MD5) authentication (16-byte max) keys.
priority <value>	Sets the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <seconds>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32,767.
transmit-delay <seconds>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32,767.

Default Values

retransmit-interval <seconds>	5 seconds
transmit-delay <seconds>	1 second
hello-interval <seconds>	10 seconds: Ethernet, point-to-point, Frame Relay, and PPP
dead-interval <seconds>	40 seconds

Usage Example

The following example sets the maximum number of seconds allowed between hello packets to 25,000:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip ospf dead-interval 25000
```

ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Optional. Selects message-digest authentication type.
null	Optional. Specifies that no authentication be used.

Default Values

By default, **ip ospf authentication** is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the PPP interface:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip ospf authentication null
```

ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Sets the network type for broadcast.
point-to-point	Sets the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP and Frame Relay default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip ospf network broadcast
```

ip proxy-arp <address> <subnet mask>

Use the **ip proxy-arp** to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example, 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, proxy ARP is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the Secure Router OS will respond to all proxy ARP requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the virtual PPP interface:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip proxy-arp
```

ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Accepts only received RIP version 1 packets on the interface.
2	Accepts only received RIP version 2 packets on the interface.

Default Values

By default, all interfaces implement RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip receive version** to specify a RIP version that overrides the **version** (in the Router RIP) configuration.

The Secure Router OS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the virtual PPP interface to accept only RIP version 2 packets:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip rip receive version 2
```

ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface. Use the **no** form of this command to restore the default value.

Syntax Description

1	Transmits only RIP version 1 packets on the interface.
2	Transmits only RIP version 2 packets on the interface.

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** to specify a RIP version that overrides the **version** (in the Router RIP) configuration.

The Secure Router OS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the virtual PPP interface to transmit only RIP version 2 packets:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ip rip send version 2
```


ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note *Using Network Address Translation (NAT) or the Secure Router OS firewall capabilities on an interface requires process switching mode (using the **no ip route-cache** command).*

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on the virtual PPP interface:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip route-cache
```

ip unnumbered <interface>

Use the **ip unnumbered** command to use the IP address assigned to the specified interface for all IP processing on the active interface. Use the **no** form of this command to remove the unnumbered configuration.

Syntax Description

<interface>	Specifies the interface (in the format type slot/port) that contains the IP address to use as the source address for all packets transmitted on this interface.

Default Values

By default, all interfaces are configured to use a specified IP address (using the **ip address** command).

Functional Notes

If **ip unnumbered** is enabled on an interface, all IP traffic from the interface will use a source IP address taken from the specified interface. For example, specifying **ip unnumbered eth 0/1** while in the PPP Interface Configuration mode configures the PPP interface to use the IP address assigned to the Ethernet interface for all IP processing. In addition, the Secure Router OS uses the specified interface information when sending route updates over the unnumbered interface. Static routes may either use the interface name (ppp 1) or the far-end address (if it will be discovered).

Usage Examples

The following example configures the PPP interface (labeled **ppp 1**) to use the IP address assigned to the Ethernet interface (**eth 0/1**):

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#ip unnumbered eth 0/1
```

keepalive <seconds>

Use the **keepalive** command to enable the transmission of keepalive packets on the interface and specify the time interval in seconds between transmitted packets.

Syntax Description

<seconds>	Defines the time interval (in seconds) between transmitted keepalive packets (valid range: 0 to 32,767 seconds).

Default Values

By default, the time interval between transmitted keepalive packets is 10 seconds.

Functional Notes

If three keepalive packets are sent to an interface with no response, the interface is considered down. To detect interface failures quickly, specify a smaller keepalive time.

Usage Examples

The following example specifies a keepalive time of 5 seconds on the virtual PPP interface:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#keepalive 5
```

lldp receive

Use the **lldp receive** command to allow LLDP packets to be received on this interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are configured to send and receive LLDP packets.

Usage Examples

The following example configures the PPP interface to receive LLDP packets:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#lldp receive
```

lldp send [management-address | port-description | system-capabilities | system-description | system-name | and-receive]

Use the **lldp send** command to configure this interface to transmit LLDP packets or to control the types of information contained in the LLDP packets transmitted by this interface.

Syntax Description

management-address	Enables transmission of management address information on this interface.
port-description	Enables transmission of port description information on this interface.
system-capabilities	Enables transmission of this device's system capabilities on this interface.
system-description	Enables transmission of this device's system description on this interface.
system-name	Enables transmission of this device's system name on this interface.
and-receive	Configures this interface to both transmit and receive LLDP packets.

Default Values

By default, all interfaces are configured to transmit and receive LLDP packets of all types.

Functional Notes

Individual LLDP information can be enabled or disabled using the various forms of the **lldp send** command. For example, use the **lldp send and-receive** command to enable transmit and receive of all LLDP information. Then use the **no lldp send port-description** command to prevent LLDP from transmitting port description information.

Usage Examples

The following example configures the PPP interface to transmit LLDP packets containing all enabled information types:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#lldp send
```

The following example configures the PPP to transmit and receive LLDP packets containing all information types:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#lldp send and-receive
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit (MTU) size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size> Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:

ATM interfaces	64 to 1520
Ethernet interfaces	64 to 1500
FDL interfaces	64 to 256
HDLC interfaces	64 to 1520
Loopback interfaces	64 to 1500
Tunnel interfaces	64 to 18,190
Virtual Frame Relay sub-interfaces	64 to 1520
Virtual PPP interfaces	64 to 1500

Default Values

<size> The default values for the various interfaces are listed below:

ATM interfaces	1500
Ethernet interfaces	1500
FDL interfaces	256
HDLC interfaces	1500
Loopback interfaces	1500
Tunnel interfaces	1500
Virtual Frame Relay sub-interfaces	1500
Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the virtual PPP interface:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#mtu 1200
```

peer default ip address <address>

Use the **peer default ip address** command to specify the default IP address of the remote end of this interface.

Syntax Description

<address>	Specifies the default IP address for the remote end (A.B.C.D).

Default Values

By default, there is no assigned peer default IP address.

Functional Notes

This command is useful if the peer does not send the IP address option during PPP negotiations.

Usage Examples

The following example sets the default peer IP address to 192.22.71.50:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#peer default ip address 192.22.71.50
```

ppp authentication [chap | eap | pap]

Use the **ppp authentication** command to specify the authentication protocol on the PPP virtual interface that the peer should use to authenticate itself.

Syntax Description

chap	Configures CHAP authentication on the interface.
eap	Configures EAP authentication on the interface.
pap	Configures PAP authentication on the interface.

Default Values

By default, PPP endpoints have no authentication configured.

Technology Review

CHAP and PAP are two authentication methods that enjoy widespread support. Both methods are included in the Secure Router OS and are easily configured.

Note	<i>The authentication method set up on the local router can be different from that on the peer. Also, just because one router requires authentication from its peer does not mean it also has to authenticate itself to the peer.</i>

Defining PAP

The Password Authentication Protocol (PAP) is used to verify that the PPP peer is a permitted device by checking a username and password configured on the peer. The username and password are both sent unencrypted across the connecting private circuit.

PAP requires two-way message passing. First, the router that is required to be authenticated (say the peer) sends an authentication request with its username and password to the router requiring authentication (say the local router). The local router then looks up the username and password in the username database within the PPP interface, and if they match sends an authentication acknowledge back to the peer.

Note	<i>The PPP username and password database is separate and distinct from the global username password database. For PAP and CHAP, use the database under the PPP interface configuration.</i>

Several example scenarios are given below for clarity.

Configuring PAP Example 1: Only the local router requires the peer to authenticate itself.

On the local router (hostname Local):

```
Local(config-ppp 1)#ppp authentication pap
```

```
Local(config-ppp 1)#username farend password same
```

On the peer (hostname Peer):


```
Peer(config-ppp 1)#ppp pap sent-username farend password same
```

The first line of the configuration sets the authentication mode as PAP. This means the peer is required to authenticate itself to the local router via PAP. The second line is the username and password expected to be sent from the peer. On the peer, the **ppp pap sent-username** command is used to specify the appropriate matching username and password.

Configuring PAP Example 2: Both routers require the peer to authenticate itself.

On the local router (hostname Local):

```
Local(config-ppp 1)#ppp authentication pap
```

```
Local(config-ppp 1)#username farend password far
```

```
Local(config-ppp 1)#ppp pap sent-username nearend password near
```

On the peer (hostname Peer):

```
Peer(config-ppp 1)#ppp authentication pap
```

```
Peer(config-ppp 1)#username nearend password near
```

```
Peer(config-ppp 1)#ppp pap sent-username farend password far
```

Now both routers send the authentication request, verify that the username and password sent match what is expected in the database, and send an authentication acknowledge.

Defining CHAP

The Challenge-Handshake Authentication Protocol (CHAP) is a three-way authentication protocol composed of a challenge response and success or failure. The MD5 protocol is used to protect usernames and passwords in the response.

First, the local router (requiring its peer to be authenticated) sends a "challenge" containing only its own unencrypted username to the peer. The peer then looks up the username in the username database within the PPP interface, and if found takes the corresponding password and its own hostname and sends a "response" back to the local router. This data is encrypted. The local router verifies that the username and password are in its own username database within the PPP interface, and if so sends a "success" back to the peer.

Note	<i>The PPP username and password database is separate and distinct from the global username password database. For PAP and CHAP, use the database under the PPP interface configuration.</i>

Several example scenarios are given below for clarity.

Configuring CHAP Example 1: Only the local router requires the peer to authenticate itself.

On the local router (hostname Local):

```
Local(config-ppp 1)#ppp authentication chap
```

```
Local(config-ppp 1)#username Peer password same
```

On the peer (hostname Peer):

```
Peer(config-ppp 1)#username Local password same
```

The first line of this configuration sets the authentication mode to CHAP. This means the peer is required

to authenticate itself to the local router via CHAP. The second line is the username and password expected to be sent from the peer. The peer must also have the **username** up both to verify the incoming username from the local router and to use the password (along with its hostname) in the response to the local router.

Note *Both ends must have identical passwords.*

Configuring CHAP Example 2: Both routers require the peer to authenticate itself.

On the local router (hostname Local):

```
Local(config-ppp 1)#ppp authentication chap
```

```
Local(config-ppp 1)#username Peer password same
```

On the peer (hostname Peer):

```
Peer(config-ppp 1)#ppp authentication chap
```

```
Peer(config-ppp 1)#username Local password same
```

This is basically identical to Example 1 except that both routers will now challenge each other and respond.

Configuring CHAP Example 3: Using the ppp chap hostname command as an alternate solution.

On the local router (hostname Local):

```
Local(config-ppp 1)#ppp authentication chap
```

```
Local(config-ppp 1)#username Peer password same
```

```
Local(config-ppp 1)#ppp chap hostname nearend
```

On the peer (hostname Peer):

```
Peer(config-ppp 1)#username nearend password same
```

Notice the peer is expecting username "nearend" even though the local router's hostname is "Local." Therefore the local router can use the **ppp chap hostname** command to send the correct name on the challenge.

Configuring CHAP Example 4: Using the ppp chap password command as an alternate solution.

On the local router (hostname Local):

```
Local(config-ppp 1)#ppp authentication chap
```

```
Local(config-ppp 1)#username Peer password different
```

On the peer (hostname Peer):

```
Peer(config-ppp 1)#username Local password same
```

```
Peer(config-ppp 1)#ppp chap password different
```

Here the local router challenges with hostname "Local." The peer verifies the name in the username database, but instead of sending the password "same" in the response, it uses the one in the **ppp chap password** command. The local router then verifies that user "Peer" with password "different" is valid and sends a "success."

ppp chap hostname <hostname>

Use the **ppp chap hostname** command to configure an alternate hostname for CHAP PPP authentication. Use the **no** form of this command to remove a configured hostname. For more information on PAP and CHAP functionality, refer to the *Technology Review* section for the command *ppp authentication [chap | eap | pap]* [on page 904](#).

Syntax Description

<hostname>	Specifies a hostname by alphanumeric string up to 80 characters in length.

Default Values

By default, there are no configured PPP CHAP hostnames.

Usage Examples

The following example specifies a PPP CHAP hostname of **my_host**:

```
ProCurve(config)#ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ppp chap hostname my_host
```

ppp chap password <password>

Use the **ppp chap password** command to configure an alternate password when the peer requires CHAP PPP authentication. Use the **no** form of this command to remove a configured password. For more information on PAP and CHAP functionality, refer to the *Technology Review* section for the command *ppp authentication [chap | eap | pap]* [on page 904](#).

Syntax Description

<password>	Specifies a password by alphanumeric string up to 80 characters in length.

Default Values

By default, there is no defined PPP CHAP password.

Usage Examples

The following example specifies a PPP CHAP password of **my_password**:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#ppp chap password my_password
```

ppp multilink

Use the **ppp multilink** command to enable multilink PPP (MPPP) operation. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, MPPP is disabled.

Functional Notes

When enabled, this interface is capable of the following:

- Combining multiple physical links into one logical link.
- Receiving upper layer protocol data units (PDU), fragmenting and transmitting over the physical links based upon the physical link MTU.
- Receiving fragments over the physical links and reassembling them into PDUs.

Usage Examples

The following example enables MPPP:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#ppp multilink
```

ppp pap sent-username <username> password <password>

Use the **ppp pap sent-username/password** command to configure a username and password when the peer requires PAP PPP authentication. Use the **no** form of this command to remove a configured password. For more information on PAP and CHAP functionality, refer to the *Technology Review* section for the command *ppp authentication [chap | eap | pap]* [on page 904](#).

Syntax Description

<username>	Specifies a username by alphanumeric string up to 80 characters in length (the username is case-sensitive).
<password>	Specifies a password by alphanumeric string up to 80 characters in length (the password is case-sensitive).

Default Values

By default, there is no defined **ppp pap sent-username** and **password**.

Usage Examples

The following example specifies a PPP PAP sent-username of **local** and a password of **my_password**:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#ppp pap sent-username local password my_password
```

pppoe ac-name <name>

Use the **pppoe ac-name** command to identify the Access Concentrator (AC) with which the Secure Router OS expects to establish a PPPoE session. Use the **no** form of this command to return to the default setting.

Syntax Description

<name>	Specifies an AC by text string (up to 255 characters) corresponding to the AC-Name Tag under RFC 2516. If this field is not specified, any access concentrator is acceptable. The AC value may be a combination of trademark, model, and serial ID information (or simply the MAC address of the unit).

Default Values

By default, no AC is specified.

Usage Examples

The following example identifies the AC with which the Secure Router OS expects to establish a PPPoE session:

```
ProCurve(config)#interface ppp 1
```

```
ProCurve(config-ppp 1)#pppoe acc-name Access_Concentrator_Name
```

pppoe service-name <name>

Use the **pppoe service-name** command to use this tag value to filter PPPoE session offers from PPPoE servers. Use the **no** form of this command to return to the default setting.

Syntax Description

<name>	Specifies a service name by text string (up to 255 characters) corresponding to the Service-Name Tags under RFC 2516. This string indicates an ISP name (or a class or quality of service). If this field is not specified, any service is acceptable.

Default Values

By default, no names are specified.

Usage Examples

The following example defines a service type that is not to be accepted by the Secure Router OS:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#pppoe service-name Service_Name
```


qos-policy out <mapname>

Use the **qos-policy out** command to apply a previously-configured QoS map to an interface. Use the **no** form of this command to remove the map from the interface. The keyword **out** specifies that this policy will be applied to outgoing packets.

Syntax Description

<mapname>	Specifies the name of a previously-created QoS map (refer to <i>qos map</i> <mapname> <sequence number> on page 363 for more information).

Default Values

No default value is necessary for this command.

Usage Examples

The following example applies the QoS map **VOICEMAP** to the PPP 1 interface:

```
ProCurve(config)#interface ppp 1
ProCurve(config-ppp 1)#qos-policy out VOICEMAP
```

snmp trap link-status

Use the **snmp trap link-status** command to control the Simple Network Management Protocol (SNMP) variable ifLinkUpDownTrapEnable (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the ifLinkUpDownTrapEnable OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 ifLinkUpDownTrapEnable OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the virtual PPP interface:

```
ProCurve(config)#interface ppp 1  
ProCurve(config-ppp 1)#no snmp trap link-status
```

username <username> password <password>

Configures the username and password of the peer to use for PPP authentication.

Syntax Description

<username>	Specifies a username by alphanumerical string up to 30 characters in length (the username is case-sensitive).
<password>	Specifies a password by alphanumerical string up to 30 characters in length (the password is case-sensitive).

Default Values

By default, there is no established username and password.

Functional Notes

PAP uses this entry to check received information from the peer. CHAP uses this entry to check the received peer hostname and a common password.

Usage Examples

The following example creates a username of **PROCURVE** with password **ROUTER** for the PPP link labeled 5:

```
ProCurve(config)#interface ppp 5  
ProCurve(config-ppp 5)#username PROCURVE password ROUTER
```

LOOPBACK INTERFACE CONFIGURATION COMMAND SET

To activate the Loopback Interface Configuration mode, enter the **interface loopback** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

shutdown [on page 1008](#)

All other commands for this command set are described in this section in alphabetical order.

access-policy <polycyname> [on page 917](#)

bandwidth <value> [on page 920](#)

crypto map <mapname> [on page 921](#)

dynamic-dns [*dyndns* | *dyndns-custom* | *dyndns-static*] <hostname> <username> <password> [on page 923](#)

ip commands [begin on page 925](#)

mtu <size> [on page 940](#)

snmp trap [on page 941](#)

snmp trap link-status [on page 942](#)

access-policy <polycyname>

Use the **access-policy** command to assign a specified access policy for the inbound traffic to an interface. Use the **no** form of this command to remove an access policy association.

Note *Configured access policies will only be active if the **ip firewall** command has been entered at the Global Configuration mode prompt to enable the SROS security features. All configuration parameters are valid, but no security data processing will be attempted unless the security features are enabled.*

Syntax Description

<polycyname>	Identifies the configured access policy by alphanumeric descriptor (all access policy descriptors are case-sensitive).

Default Values

By default, there are no configured access policies associated with an interface.

Functional Notes

To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy <policy name>**.

Usage Examples

The following example associates the access policy **UnTrusted** (to allow inbound traffic to the Web server) to the loopback interface:

Enable the Secure Router OS security features:

```
ProCurve(config)#ip firewall
```

Create the access list (this is the packet selector):

```
ProCurve(config)#ip access-list extended InWeb
```

```
ProCurve(config-ext-nacl)#permit tcp any host 63.12.5.253 eq 80
```

Create the access policy that contains the access list **InWeb**:

```
ProCurve(config)#ip policy-class UnTrusted
```

```
ProCurve(config-policy-class)#allow list InWeb
```

Associate the access policy with the loopback interface:

```
ProCurve(config)#interface loopback 1
```

```
ProCurve(config-loop 1)#access-policy UnTrusted
```

Technology Review

Creating access policies and lists to regulate traffic through the routed network is a four-step process:

Step 1:

Enable the security features of the Secure Router OS using the **ip firewall** command.

Step 2:

Create an access list to permit or deny specified traffic. Standard access lists provide pattern matching for source IP addresses only. (Use extended access lists for more flexible pattern matching.) IP addresses can be expressed in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** <A.B.C.D> to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the <A.B.C.D> <wildcard> format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Step 3:

Create an IP policy class that uses a configured access list. Secure Router OS access policies are used to permit, deny, or manipulate (using NAT) data for each physical interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list <access list names>

All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list <access list names>

All packets passed by the access list(s) entered will be dropped from the router system.

allow list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list <access list names> **policy** <access policy name>

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

nat source list <access list names> **address** <IP address> **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network.

nat source list *<access list names>* **interface** *<interface>* **overload**

All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. This hides private IP addresses from outside the local network.

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Step 4:

Apply the created access policy to an interface. To assign an access policy to an interface, enter the interface configuration mode for the desired interface and enter **access policy** *<policy name>*. The following example assigns access policy **MatchAll** to the loopback interface labeled 1:

```
ProCurve(config)#interface loopback 1
```

```
ProCurve(config-loop 1)#access-policy MatchAll
```

bandwidth <value>

Use the **bandwidth** command to provide the bandwidth value of an interface to the higher-level protocols. This value is used in cost calculations. Use the **no** form of this command to restore the default values.

Syntax Description

<value>	Specifies bandwidth in kbps.

Default Values

To view default values, use the **show interfaces** command.

Functional Notes

The **bandwidth** command is an informational value that is communicated to the higher-level protocols to be used in cost calculations. This is a routing parameter only and does not affect the physical interface.

Usage Examples

The following example sets bandwidth of the loopback interface to 10 Mbps:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#bandwidth 10000
```


crypto map <mapname>

Use the **crypto map** command to associate crypto maps with the interface.

Note *When you apply a map to an interface, you are applying all crypto maps with the given map name. This allows you to apply multiple crypto maps if you have created maps which share the same name but have different map index numbers.*

*For VPN configuration example scripts, refer to the **VPN Configuration Guide** located on the ProCurve SROS Documentation CD provided with your unit.*

Syntax Description

<mapname> Specifies the crypto map name that you wish to assign to the interface.

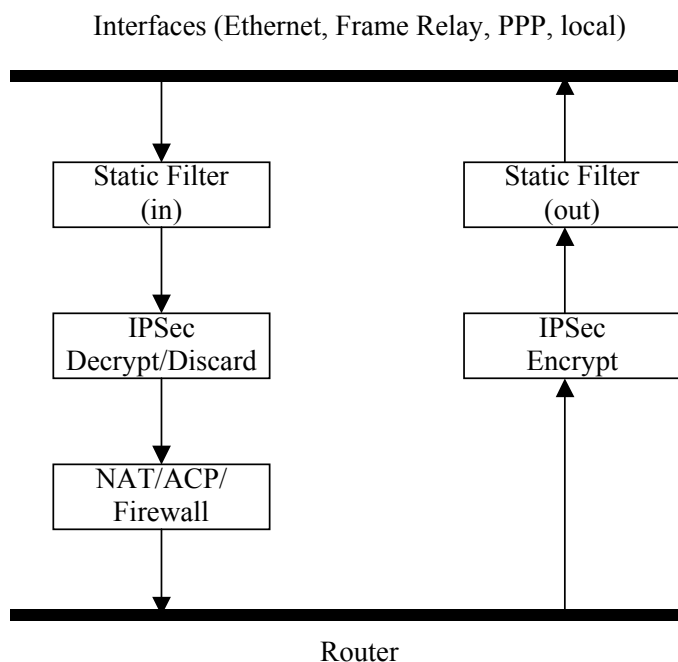
Default Values

By default, no crypto maps are assigned to an interface.

Functional Notes

When configuring a system to use both the stateful inspection firewall and IKE negotiation for VPN, keep the following information in mind:

When defining the policy class and associated access-control lists (ACLs) that describe the behavior of the firewall, do not forget to include the traffic coming into the system over a VPN tunnel terminated by the system. The firewall should be set up with respect to the unencrypted traffic that is destined to be sent or received over the VPN tunnel. The following diagram represents typical Secure Router OS data-flow logic.



As shown in the previous diagram, data coming into the product is first processed by the static filter associated with the interface on which the data is received. This access group is a true static filter and is available for use regardless of whether the firewall is enabled or disabled. Next (if the data is encrypted) it is sent to the IPSec engine for decryption. The decrypted data is then processed by the stateful inspection firewall. Therefore, given a terminating VPN tunnel, only unencrypted data is processed by the firewall.

The ACLs for a crypto map on an interface work in reverse logic to the ACLs for a policy-class on an interface. When specifying the ACLs for a crypto map, the source information is the private local-side, unencrypted source of the data. The destination information will be the far end, unencrypted destination of the data. However, ACLs for a policy class work in reverse. The source information for the ACL in a policy class is the far end. The destination information is the local side.

Usage Examples

The following example applies all crypto maps with the name **MyMap** to the loopback interface:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#crypto map MyMap
```

dynamic-dns [dyndns | dyndns-custom | dyndns-static] <hostname> <username> <password>

Use the **dynamic-dns** command to configure Dynamic DNS service provided by Dynamic Network Services, Inc. (www.dyndns.org).

Syntax Description

See *Functional Notes*, below, for argument descriptions.

Default Values

No default is necessary for this command.

Functional Notes

dyndns - The Dynamic DNSSM service allows you to alias a dynamic IP address to a static hostname in various domains. This allows your unit to be more easily accessed from various locations on the Internet. This service is provided for up to five hostnames.

dyndns-custom - DynDNS.org's Custom DNSSM service provides a full DNS solution, giving you complete control over an entire domain name. A Web-based interface provides two levels of control over your domain, catering to average or power users. Five globally redundant DNS servers ensure that your domain will always resolve.

A choice of two interfaces is available. The basic interface is designed for most users. It comes preconfigured for the most common configuration and allows for easy creation of most common record types. The advanced interface is designed for system administrators with a solid DNS background, and provides layout and functionality similar to a BIND zone file, allowing for the creation of nearly any record type.

Custom DNSSM can be used with both static and dynamic IPs and has the same automatic update capability through Custom DNS-aware clients as Dynamic DNS.

dyndns-static - The Static DNS service is similar to Dynamic DNS service in that it allows a hostname such as yourname.dyndns.org to point to your IP address. Unlike a Dynamic DNS host, a Static DNS host does not expire after 35 days without updates, but updates take longer to propagate through the DNS system. This service is provided for up to five hostnames.

When to Use Dynamic DNS

If your IP address does not change often (or at all) but you still want an easy name to remember it by (without having to purchase your own domain name), Static DNS service is ideal for you.

If you would like to use your own domain name (such as yourname.com), you need Custom DNS service, which also provides full dynamic and static IP address support.

Dynamic DNS service can be extremely helpful for site-to-site VPN connections using dynamic IP addressing. Instead of creating a tunnel to the IP-address, a tunnel to the DNS-name is established.

Usage Examples

The following example sets the **dynamic-dns** to **dyndns-custom** with hostname **host**, username **user**, and password **pass**.

```
ProCurve(config)#interface loopback 1
```

```
ProCurve(config-loop 1)#dynamic-dns dyndns-custom host user pass
```

ip access-group <listname> [in | out]

Use the **ip access-group** command to create an access list to be used for packets transmitted on or received from the specified interface. Use the **no** form of this command to disable this type of control.

Syntax Description

<listname>	Specifies IP access list name.
in	Enables access control on packets received on the specified interface.
out	Enables access control on packets transmitted on the specified interface.

Default Values

By default, these commands are disabled.

Functional Notes

When this command is enabled, the IP destination address of each packet must be processed by the assigned access-list parameters before being passed through to the router system. If the packet is not acceptable per these settings, it is dropped.

Usage Examples

The following example sets up the router to allow only Telnet traffic into the loopback interface:

```
ProCurve(config)#ip access-list extended TelnetOnly
ProCurve(config-ext-nacl)#permit tcp any any eq telnet
ProCurve(config-ext-nacl)#interface loopback 1
ProCurve(config-loop 1)#ip access-group TelnetOnly in
```

ip address <address> <mask> [secondary]

Use the **ip address** command to define an IP address on the specified interface. Use the optional keyword **secondary** to define a secondary IP address. Use the **no** form of this command to remove a configured IP address.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<mask>	Specifies the subnet mask that corresponds to the listed IP address.
secondary	Optional. Configures a secondary IP address for the specified interface.

Default Values

By default, there are no assigned IP addresses.

Functional Notes

Use secondary IP addresses to allow dual subnets on a single interface (when you need more IP addresses than the primary subnet can provide). When using secondary IP addresses, avoid routing loops by verifying that all devices on the network segment are configured with secondary IP addresses on the secondary subnet.

Usage Examples

The following example configures a secondary IP address of **192.22.72.101/30**:

```
ProCurve(config)#interface loopback 1
```

```
ProCurve(config-loop 1)#ip address 192.22.72.101 255.255.255.252 secondary
```

To view configured secondary IP address(es), use the **show running-config** command. Secondary IP address information is not available through other **show** commands.

ip helper-address <address>

Use the **ip helper-address** command to configure the Secure Router OS to forward User Datagram Protocol (UDP) broadcast packets received on the interface. Use the **no** form of this command to disable forwarding packets.

Note	<i>The ip helper command must be used in conjunction with the ip forward-protocol command to configure the Secure Router OS to forward UDP broadcast packets.</i>

Syntax Description

<address>	Specifies the destination IP address (in dotted decimal notation) for the forwarded UDP packets.

Default Values

By default, broadcast UDP packets are not forwarded.

Functional Notes

When used in conjunction with the **ip forward-protocol** command, the **ip helper-address** feature allows you to customize which broadcast packets are forwarded.

To implement the helper address feature, assign helper address(es) (specifying the device that needs to receive the broadcast traffic) to the interface closest to the host that transmits the broadcast packets. When broadcast packets (of the specified type forwarded using the **ip forward-protocol** command) are received on the interface, they will be forwarded to the device that needs the information.

Only packets meeting the following criteria are considered eligible by the **ip helper-address** feature:

1. The packet IP protocol is User Datagram Protocol (UDP).
2. Any UDP port specified using the **ip forward-protocol** command.
3. The media access control (MAC) address of the frame is an all-ones broadcast address (ffff.ffff.ffff).
4. The destination IP address is broadcast defined by all ones (255.255.255.255) or a subnet broadcast (for example, 192.33.4.251 for the 192.33.4.248/30 subnet).

Usage Examples

The following example forwards all DNS broadcast traffic to the DNS server with IP address 192.33.5.99:

```
ProCurve(config)#ip forward-protocol udp domain
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip helper-address 192.33.5.99
```

ip igmp

Use the **ip igmp** command to configure multicasting-related functions for the interface.

Syntax Description

immediate-leave	If only one host (or IGMP snooping switch) is connected to the interface, when a leave is received, multicast of that group is immediately terminated as opposed to sending a group query and timing out the group if no device responds. Works in conjunction with ip igmp last-member-query-interval . Applies to all groups when configured.
last-member-query-interval <milliseconds>	Controls the timeout used to detect whether any group receivers remain on an interface after a receiver leaves a group. If a receiver sends a leave group message (IGMP Version 2), the router sends a group-specific query on that interface. After twice the time specified by this command plus as much as one second longer, if no receiver responds, the router removes that interface from the group and stops sending that group's multicast packets to the interface. Range: 100 to 65535 ms. Default: 1000 ms.
querier-timeout <seconds>	Specifies the number of seconds that the router waits after the current querier's last query before it takes over as querier (IGMP V2). Range: 60 to 300 seconds. Default: 2x the query-interval value.
query-interval <seconds >	Specifies the interval at which IGMP queries are sent on an interface. Host query messages are addressed to the all-hosts multicast group with an IP TTL of 1. The router uses queries to detect whether multicast group members are on the interface and to select an IGMP designated router for the attached segment (if more than one multicast router exists). Only the designated router for the segment sends queries. For IGMP V2, the designated router is the router with the lowest IP address on the segment. Range: 0 to 65535 seconds. Default: 60 seconds.
query-max-response-time <seconds>	Specifies the maximum response time advertised by this interface in queries when using IGMP V2. Hosts are allowed a random time within this period to respond, reducing response bursts. Default: 10 seconds.
static-group <group-address>	Configures the router's interface to be a statically connected member of the specified group. Packets received on the correct RPF interface are forwarded to this interface regardless of whether any receivers have joined the specified group using IGMP.
version [1 2]	Sets the interface's IGMP version. The default setting is version 2.

Usage Examples

The following example sets the query message interval on the interface to 200 milliseconds:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip igmp last-member-query-interval 200
```


ip mcast-stub downstream

Use the **ip mcast-stub downstream** command to enable multicast forwarding and IGMP (router mode) on an interface and place it in multicast stub downstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub upstream** commands. Downstream interfaces connect to segments with multicast hosts. Multiple interfaces may be configured in downstream mode; however, interfaces connecting to the multicast network (upstream) should not be configured in downstream mode. Interfaces configured as downstream should have the lowest IP address of all IGMP-capable routers on the connected segment in order to be selected as the designated router and ensure proper forwarding. Refer to *ip mcast-stub helper-address <ip address>* on page 327 and *ip mcast-stub upstream* on page 931 for more information.

Usage Examples

The following example enables multicast forwarding and IGMP on the interface:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip mcast-stub downstream
```

ip mcast-stub helper-enable

Use the **ip mcast-stub helper-enable** command to assign the **ip mcast-stub helper-address** as the IGMP proxy. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address**, **ip mcast-stub upstream**, and **ip mcast-stub downstream** commands. When enabled, the interface becomes a helper forwarding interface. The IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the unit to perform as an IGMP proxy. Refer to *ip mcast-stub helper-address <ip address>* on page 389, *ip mcast-stub downstream* on page 929, and *ip mcast-stub upstream* on page 931 for more information.

Usage Examples

The following example sets the helper address as the IGMP proxy:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#ip mcast-stub helper-enable
```

ip mcast-stub upstream

Use the **ip mcast-stub upstream** command to enable multicast forwarding on an interface and place it in multicast stub upstream mode. Use the **no** form of this command to disable.

Syntax Description

No subcommands.

Default Values

By default, this command is disabled.

Functional Notes

This command is used in IP multicast stub applications in conjunction with the **ip mcast-stub helper-address** and **ip mcast-stub downstream** commands. When enabled, the interface becomes a candidate to be a helper forwarding interface. If chosen as the best path toward the helper address by the router's unicast route table, the IGMP host function is dynamically enabled and the interface becomes the active upstream interface, enabling the router to perform as an IGMP proxy. Though multiple interfaces may be candidates, no more than one interface will actively serve as the helper forwarding interface. Refer to *ip mcast-stub helper-address <ip address>* on page 327 and *ip mcast-stub downstream* on page 929 for more information.

Usage Examples

The following example enables multicast forwarding on the interface:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip mcast-stub upstream
```

ip ospf

Use the **ip ospf** command to customize OSPF settings (if needed).

Syntax Description

authentication-key	Specifies a simple-text authentication password to be used by other routers using <i><password></i> the OSPF simple password authentication.
cost <i><value></i>	Specifies the OSPF cost of sending a packet on the interface. This value overrides any computed cost value. Range: 1 to 65535.
dead-interval <i><seconds></i>	Sets the maximum interval allowed between hello packets. If the maximum is exceeded, neighboring devices will determine that the device is down. Range: 0 to 32767.
hello-interval <i><seconds></i>	Specifies the interval between hello packets sent on the interface. Range: 0 to 32767.
message-digest-key <i><keyid></i> md5 <i><key></i>	Configure OSPF Message Digest 5 (MD5) authentication (16-byte max) keys.
priority <i><value></i>	Sets the OSPF priority. The value set in this field helps determine the designated router for this network. Range: 0 to 255.
retransmit-interval <i><seconds></i>	Specifies the time between link-state advertisements (LSAs). Range: 0 to 32767.
transmit-delay <i><seconds></i>	Sets the estimated time required to send an LSA on the interface. Range: 0 to 32767.

Default Values

retransmit-interval <i><seconds></i>	5 seconds
transmit-delay <i><seconds></i>	1 second
hello-interval <i><seconds></i>	10 seconds: Ethernet, point-to-point, Frame Relay, and PPP
dead-interval <i><seconds></i>	40 seconds

Usage Examples

The following example sets the maximum number of seconds allowed between hello packets to 25K:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip ospf dead-interval 25000
```

ip ospf authentication [message-digest | null]

Use the **ip ospf authentication** command to authenticate an interface that is performing OSPF authentication.

Syntax Description

message-digest	Optional. Specifies message-digest authentication type.
null	Optional. Specifies that no authentication be used.

Default Values

By default, this is set to null (meaning no authentication is used).

Usage Examples

The following example specifies that no authentication will be used on the loopback interface:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#ip ospf authentication null
```

ip ospf network [broadcast | point-to-point]

Use the **ip ospf network** command to specify the type of network on this interface.

Syntax Description

broadcast	Sets the network type for broadcast.
point-to-point	Sets the network type for point-to-point.

Default Values

By default, Ethernet defaults to broadcast. PPP and Frame Relay default to point-to-point.

Functional Notes

A point-to-point network will not elect designated routers.

Usage Examples

The following example designates a broadcast network type:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#ip ospf network broadcast
```

ip proxy-arp <address> <subnet mask>

Use the **ip proxy-arp** command to enable proxy Address Resolution Protocol (ARP) on the interface. Use the **no** form of this command to disable this feature.

Syntax Description

<address>	Defines the IP address for the interface in dotted decimal notation (for example: 192.22.73.101).
<subnet mask>	Specifies the subnet mask that corresponds to the listed IP address.

Default Values

By default, proxy ARP is enabled.

Functional Notes

In general, the principle of proxy ARP allows a router to insert its IP address in the source IP address field of a packet (if the packet is from a host on one of its subnetworks). This allows hosts to reach devices on other subnetworks without implementing routing or specifying a default gateway.

If proxy ARP is enabled, the Secure Router OS will respond to all proxy ARP requests with its specified MAC address and forward packets accordingly.

Enabling proxy ARP on an interface may introduce unnecessary ARP traffic on the network.

Usage Examples

The following enables proxy ARP on the loopback interface:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip proxy-arp
```

ip rip receive version [1 | 2]

Use the **ip rip receive version** command to configure the RIP version the unit accepts in all RIP packets received on the interface.

Syntax Description

1	Accepts only received RIP version 1 packets on the interface.
2	Accepts only received RIP version 2 packets on the interface.

Default Values

By default, all interfaces implement RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip receive version** to specify a RIP version that overrides the **version** (in the Router RIP) configuration.

The Secure Router OS only accepts one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the loopback interface to accept only RIP version 2 packets:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip rip receive version 2
```


ip rip send version [1 | 2]

Use the **ip rip send version** command to configure the RIP version the unit sends in all RIP packets transmitted on the interface.

Syntax Description

1	Transmits only RIP version 1 packets on the interface.
2	Transmits only RIP version 2 packets on the interface.

Default Values

By default, all interfaces transmit RIP version 1 (the default value for the **version** command).

Functional Notes

Use the **ip rip send version** to specify a RIP version that overrides the **version** (in the Router RIP) configuration.

The secure Router OS only transmits one version (either 1 or 2) on a given interface.

Usage Examples

The following example configures the loopback interface to transmit only RIP version 2 packets:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip rip send version 2
```

ip route-cache

Use the **ip route-cache** command to enable route caching on the interface. Use the **no** form of this command to disable route caching and return to process switching mode.

Note	<i>Using Network Address Translation (NAT) or the Secure Router OS firewall capabilities on an interface requires process switching mode (using the no ip route-cache command).</i>

Syntax Description

No subcommands.

Default Values

By default, route caching is enabled on all interfaces.

Functional Notes

Route caching allows an IP interface to provide optimum performance when processing IP traffic.

Usage Examples

The following example enables route caching on the loopback interface:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip route-cache
```

ip unnumbered <interface>

Use the **ip unnumbered** command to use the IP address assigned to the specified interface for all IP processing on the active interface. Use the **no** form of this command to remove the unnumbered configuration.

Syntax Description

<interface>	Specifies the interface in the format type slot/port (e.g., ppp 1) that contains the IP address to be used as the source address for all packets transmitted on this interface. Enter ip unnumbered ? for a complete list of valid interfaces.

Default Values

By default, all interfaces are configured to use a specified IP address (using the **ip address** command).

Functional Notes

If **ip unnumbered** is enabled on an interface, all IP traffic from the interface will use a source IP address taken from the specified interface. For example, specifying **ip unnumbered ppp 1** while in the Ethernet Interface Configuration mode configures the Ethernet interface to use the IP address assigned to the PPP interface for all IP processing. In addition, the Secure Router OS uses the specified interface information when sending route updates over the unnumbered interface.

Usage Examples

The following example configures the loopback interface (labeled **loop 1**) to use the IP address assigned to the PPP interface (**ppp 1**):

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#ip unnumbered ppp 1
```

mtu <size>

Use the **mtu** command to configure the maximum transmit unit size for the active interface. Use the **no** form of this command to return to the default value.

Syntax Description

<size> Configures the window size for transmitted packets. The valid ranges for the various interfaces are listed below:

ATM interfaces	64 to 1520
Ethernet interfaces	64 to 1500
FDL interfaces	64 to 256
HDLC interfaces	64 to 1520
Loopback interfaces	64 to 1500
Tunnel interfaces	64 to 18,190
Virtual Frame Relay sub-interfaces	64 to 1520
Virtual PPP interfaces	64 to 1500

Default Values

<size> The default values for the various interfaces are listed below:

ATM interfaces	1500
Ethernet interfaces	1500
FDL interfaces	256
HDLC interfaces	1500
Loopback interfaces	1500
Tunnel interfaces	1500
Virtual Frame Relay sub-interfaces	1500
Virtual PPP interfaces	1500

Functional Notes

OSPF will not become adjacent on links where the MTU sizes do not match. If router A and router B are exchanging hello packets but their MTU sizes do not match, they will never reach adjacency. This is by design and required by the RFC.

Usage Examples

The following example specifies an MTU of 1200 on the loopback interface:

```
ProCurve(config)#interface loopback 1
ProCurve(config-loop 1)#mtu 1200
```

snmp trap

Use the **snmp trap** command to enable all supported Simple Network Management Protocol (SNMP) traps on the interface.

Syntax Description

No subcommands.

Default Values

By default, all interfaces (except virtual Frame Relay interfaces and sub-interfaces) have SNMP traps enabled.

Usage Examples

The following example enables SNMP capability on the Ethernet interface:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#snmp trap
```

snmp trap link-status

Use the **snmp trap link-status** command to control the Simple Network Management Protocol (SNMP) variable *ifLinkUpDownTrapEnable* (RFC2863) to enable (or disable) the interface to send SNMP traps when there is an interface status change. Use the **no** form of this command to disable this trap.

Syntax Description

No subcommands.

Default Values

By default, the *ifLinkUpDownTrapEnable* OID is enabled for all interfaces except virtual Frame Relay interfaces.

Functional Notes

The **snmp trap link-status** command is used to control the RFC2863 *ifLinkUpDownTrapEnable* OID (OID number 1.3.6.1.2.1.31.1.1.1.14.0).

Usage Examples

The following example disables the link-status trap on the loopback interface:

```
ProCurve(config)#interface loopback 1  
ProCurve(config-loop 1)#no snmp trap link-status
```

LINE (CONSOLE) INTERFACE CONFIG COMMAND SET

To activate the Line (Console) Interface Configuration mode, enter the **line console 0** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#line console 0  
ProCurve(config-con 0)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

databits [7 | 8] [on page 944](#)

flowcontrol [none | software in] [on page 945](#)

line-timeout <minutes> [on page 946](#)

login [on page 947](#)

login authentication <aaa login list> [on page 948](#)

login local-userlist [on page 949](#)

parity [even | mark | none | odd | space] [on page 950](#)

password [md5] <password> [on page 951](#)

speed <rate> [on page 952](#)

stopbits [1 | 2] [on page 953](#)

databits [7 | 8]

he **databits** command to set the number of databits per character for a terminal session. This value must match the configuration of your VT100 terminal or terminal emulator software. The default is 8 databits per character. Use the **no** form of this command to return to the default value.

Syntax Description

7	Specifies 7 data bits per character.
8	Specifies 8 data bits per character.

Default Values

<i><option></i>	8

Usage Examples

The following example configures 7 databits per character for the console terminal session:

```
ProCurve(config)#line console 0  
ProCurve(config-con 0)#databits 7
```


flowcontrol [none | software in]

Use the **flowcontrol** command to set flow control for the line console.

Syntax Description

none	Specifies no flow control.
software in	Configures the Secure Router OS to derive flow control from the attached device.

Default Values

By default, flow control is set to none.

Usage Examples

The following example configures no flow control for the line console:

```
ProCurve(config)#line console 0  
ProCurve(config-con 0)#flowcontrol none
```

line-timeout <minutes>

Use the **line-timeout** command to specify the number of minutes a line session may remain inactive before the Secure Router OS terminates the session. Use the **no** form of this command to return to the default value.

Syntax Description

<minutes>	Specifies the number of minutes a line session may remain inactive before the Secure Router OS terminates the session.

Entering a **line-timeout** value of 0 disables the feature.

Default Values

<minutes>	15 minutes (Console and Telnet)

Usage Examples

The following example specifies a timeout of 2 minutes:

```
ProCurve(config)#line console 0
```

```
ProCurve(config-con 0)#line-timeout 2
```

login

Use the **login** command to enable security login on the line session requiring the password configured using the **password** command. Use the **no** form of this command to disable the login feature.

Syntax Description

No subcommands.

Default Values

By default, there is no login password set for access to the unit.

Usage Examples

The following example enables the security login feature and specifies a password on the available console session:

```
ProCurve(config)#line console 0  
ProCurve(config-console 0)#login  
ProCurve(config-console 0)#password mypassword
```

login authentication <aaa login list>

Use the **login authentication** command to specify the named AAA login list to use for authenticating users connecting on this line.

Syntax Description

<aaa login list>	Specifies the AAA login list to use for authentication.

Default Values

The default value is the default AAA list.

Functional Notes

If the AAA subsystem is activated but no login authentication list is given, the default list is used. If the default list is used but the default list is not configured, the behavior for consoles is to be granted access. This prevents a lockout configuration.

Usage Examples

The following example specifies that **myList** will be used for authenticating users connecting on this line:

```
ProCurve(config)#line console 0
```

```
ProCurve(config-con 0)#login authentication myList
```

login local-userlist

Use the **login local-userlist** command to enable security login for the terminal session requiring the usernames and passwords configured using the **username/password** Global Configuration command. Use the **no** form of this command to disable the login local-userlist feature.

Note *All user properties assigned using the **username/password** command are valid when using the **login local-userlist** command.*

Syntax Description

No subcommands.

Default Values

By default, there is no login password set for access to the unit.

Usage Examples

The following example displays creating a local userlist and enabling the security login feature on the **CONSOLE** port:

```
ProCurve(config)#username my_user password my_password
ProCurve(config)#line console 0
ProCurve(config-con 0)#login local-userlist
```

When connecting to the unit, the following prompts are displayed:

```
User Access Login
Username: Procurve
Password:
Router#
```

parity [even | mark | none | odd | space]

Use the **parity** command to specify the type of parity used as error correction. This value must match the configuration of your VT100 terminal or terminal emulator software. Use the **no** form of this command to return to the default value.

Syntax Description

even	The parity bit is set to 0 if the number of 1 bits in the data sequence is odd, or set to 1 if the number of 1 bits is even.
mark	The parity bit is always set to 1.
none	No parity bit used.
odd	The parity bit is set to 1 if the number of 1 bits in the data sequence is even, or set to 0 if the number is odd.
space	The parity bit is always set to 0.

Default Values

<i><option></i>	none

Functional Notes

Parity is the process used to detect whether characters have been altered during the data transmission process. Parity bits are appended to data frames to ensure that parity (whether it be odd or even) is maintained.

Usage Examples

The following example specifies mark parity for the console terminal session:

```
ProCurve(config)#line console 0
ProCurve(config-con 0)#parity mark
```

password [md5] <password>

Use the **password** command to configure the password (with optional encryption) required on the line session when security login is enabled (using the **login** command). Use the **no** form of this command to remove a configured password.

Syntax Description

md5	Optional. Specifies Message Digest 5 (md5) as the encryption protocol to use when displaying the enable password during show commands. If the md5 keyword is not used, encryption is not used when displaying the enable password during show commands.
<password>	Alphanumeric character string (up to 16 characters) used to specify the password for the line session.

Default Values

By default, there is no login password set for access to the unit.

Usage Examples

The following example enables the security login feature and specifies a password on the **CONSOLE** port:

```
ProCurve(config)#line console 0
ProCurve(config-con 0)#login
ProCurve(config-con 0)#password mypassword
```

To provide extra security, the Secure Router OS can encrypt the enable password when displaying the current configuration. For example, the following is a **show configuration** printout (password portion) with an unencrypted enable password (procurve):

```
!
enable password procurve
!
```

Alternately, the following is a **show configuration** printout (password portion) with an enable password of procurve using md5 encryption:

```
!
enable password md5 encrypted 5aa5fbae7d01a90e79fb57705ce74676
!
```

speed *<rate>*

Use the **speed** command to specify the data rate for the **Console** port. This setting must match your VT100 terminal emulator or emulator software. Use the **no** form of this command to restore the default value.

Syntax Description

<i><rate></i>	Specifies rate of data transfer on the interface (2400; 4800; 9600; 19,200; 38,400; 57,600; or 115,200 bps).

Default Values

<i><rate></i>	9600 bps

Usage Examples

The following example configures the **Console** port for 19200 bps:

```
ProCurve(config)#line console 0  
ProCurve(config-con 0)#speed 19200
```


stopbits [1 | 2]

Use the **stopbits** command to set the number of stopbits per character for a terminal session. This value must match the configuration of your VT100 terminal or terminal emulator software. The default is 1 stopbit per character. Use the **no** form of this command to return to the default value.

Syntax Description

1	Specifies 1 stopbit per character.
2	Specifies 2 stopbits per character.

Default Values

<i><option></i>	1

Usage Examples

The following example configures 2 stopbits per character for the console terminal session:

```
ProCurve(config)#line console 0  
ProCurve(config-con 0)#stopbits 2
```

LINE (TELNET) INTERFACE CONFIG COMMAND SET

To activate the Line (Telnet) Interface Configuration mode, enter the **line telnet** command specifying a Telnet session(s) at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#line telnet 0 4
ProCurve(config-telnet0-4)#
```

You can select a single line by entering the **line telnet** command followed by the line number (0-4). For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#line telnet 2
ProCurve(config-telnet2)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

access-class *<listname>* [in on page 955](#)
line-timeout *<minutes>* [on page 956](#)
login [on page 957](#)
login authentication *<aaa login list>* [on page 958](#)
login local-userlist [on page 959](#)
password [*md5*] *<password>* [on page 960](#)

access-class <listname> in

Use the **access-class in** command to restrict Telnet access using a configured access list. Received packets passed by the access list will be allowed. Use the access list configuration to deny hosts or entire networks or to permit specified IP addresses.

Syntax Description

<listname>	Alphanumeric descriptor for identifying the configured access list (all access list descriptors are case-sensitive).

Default Values

By default, there are no configured access lists associated with Telnet sessions.

Functional Notes

When using the **access-class in** command to associate an access list with a Telnet session, remember to duplicate the **access-class in** command for all configured Telnet sessions 0 through 4. Telnet access to the unit using a particular Telnet session is not possible. Users will be assigned the first available Telnet session.

Usage Examples

The following example associates the access list **Trusted** (to allow Telnet sessions from the 192.22.56.0/24 network) with all Telnet sessions (0 through 4):

Create the access list:

```
ProCurve(config)#ip access-list standard Trusted
ProCurve(config)#permit 192.22.56.0 0.0.0.255
```

Enter the line (telnet):

```
ProCurve(config)#line telnet 0 4
```

Associate the access list with the Telnet session:

```
ProCurve(config-telnet0-4)#access-class Trusted in
```

line-timeout <minutes>

Use the **line-timeout** command to specify the number of minutes a line session may remain inactive before the Secure Router OS terminates the session. Use the **no** form of this command to return to the default value.

Syntax Description

<minutes>	Specifies the number of minutes a line session may remain inactive before the Secure Router OS terminates the session.

Entering a **line-timeout** value of 0 disables the feature.

Default Values

<minutes>	15 minutes (Console and Telnet)

Usage Examples

The following example specifies a timeout of 2 minutes:

```
ProCurve(config)#line telnet 0
```

```
ProCurve(config-telnet0)#line-timeout 2
```

login

Use the **login** command to enable security login on the line session requiring the password configured using the **password** command. Use the **no** form of this command to disable the login feature.

Syntax Description

No subcommands.

Default Values

By default, there is no login password set for access to the unit.

Usage Examples

The following example enables the security login feature and specifies a password on all the available Telnet sessions (0 through 4):

```
ProCurve(config)#line telnet 0 4  
ProCurve(config-telnet0-4)#login  
ProCurve(config-telnet0-4)#password mypassword
```

login authentication <aaa login list>

Use the **login authentication** command to specify the named AAA login list to use for authenticating users connecting on this line.

Syntax Description

<code><aaa login list></code>	Specifies the AAA login list to use for authentication.

Default Values

The default value is the default AAA list.

Functional Notes

If the AAA subsystem is activated but no login authentication list is given, the default list is used. If the default list is used but the default list is not configured, the behavior for telnets is to use the local user database.

Usage Examples

The following example specifies that **myList** will be used for authenticating users connecting on this line:

```
ProCurve(config)#line telnet 2
```

```
ProCurve(config-telnet2)#login authentication myList
```

login local-userlist

Use the **login local-userlist** command to enable security login for the terminal session requiring the usernames and passwords configured using the **username/password** Global Configuration command. Use the **no** form of this command to disable the login local-userlist feature.

Note *All user properties assigned using the **username/password** command are valid when using the **login local-userlist** command.*

Syntax Description

No subcommands.

Default Values

By default, there is no login password set for access to the unit.

Usage Examples

The following example displays creating a local userlist and enabling the security login feature:

```
ProCurve(config)#username my_user password my_password  
ProCurve(config)#line telnet 0  
ProCurve(config-telnet0)#login local-userlist
```

When connecting to the unit, the following prompts are displayed:

```
User Access Login  
Username: my_user  
Password:  
Router#
```

password [md5] <password>

Use the **password** command to configure the password (with optional encryption) required on the line session when security login is enabled (using the **login** command). Use the **no** form of this command to remove a configured password.

Syntax Description

md5	Optional. Specifies Message Digest 5 (md5) as the encryption protocol to use when displaying the enable password during show commands. If the md5 keyword is not used, encryption is not used when displaying the enable password during show commands.
<password>	Alphanumeric character string (up to 16 characters) used to specify the password for the line session.

Default Values

By default, there is no login password set for access to the unit.

Usage Examples

The following example enables the security login feature and specifies a password for the Telnet session 0:

```
ProCurve(config)#line telnet 0
```

```
ProCurve(config-telnet0)#login
```

```
ProCurve(config-telnet0)#password mypassword
```

To provide extra security, the Secure Router OS can encrypt the enable password when displaying the current configuration. For example, the following is a **show configuration** printout (password portion) with an unencrypted enable password (procurve):

```
!
```

```
enable password procurve
```

```
!
```

Alternately, the following is a **show configuration** printout (password portion) with an enable password of procurve using md5 encryption:

```
!
```

```
enable password md5 encrypted 5aa5fbae7d01a90e79fb57705ce74676
```

```
!
```

ROUTER (RIP) CONFIGURATION COMMAND SET

To activate the Router (RIP) Configuration mode, enter the **router rip** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#router rip  
ProCurve(config-rip)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

auto-summary [on page 962](#)

default-metric <value> [on page 963](#)

network <address> <subnet mask> [on page 964](#)

passive-interface <interface> [on page 965](#)

redistribute connected [metric <value>] [on page 966](#)

redistribute ospf [metric <value>] [on page 967](#)

redistribute static [metric <value>] [on page 968](#)

version [1 | 2] [on page 969](#)

auto-summary

Use the **auto-summary** command to have RIP version 2 summarize subnets to the classful boundaries. Use the **no** form of this command to disable this summarization.

Syntax Description

No subcommands.

Default Values

By default, auto-summary is disabled.

Functional Notes

Use this command if you are subdividing a classful network into many subnets and these subnets are to be advertised over a slow link (64K or less) to a router that can only reach the classful network via the router you are configuring.

Usage Examples

The following example configures the router to not automatically summarize network numbers:

```
ProCurve(config)#router rip
```

```
ProCurve(config-rip)#no auto-summary
```

default-metric <value>

Use the **default-metric** command to set the default metric value for the RIP routing protocol. Use the **no** form of this command to return to the default settings.

Syntax Description

<value>	Sets the default metric value (range: 1 to 4,294,967,295 Mbps).

Default Values

By default, this value is set at 0.

Functional Notes

The metric value defined using the **redistribute** command overrides the **default-metric** command's metric setting. See *redistribute ospf [metric <value>]* [on page 967](#) for related information.

Usage Examples

The following example shows a router using both RIP and OSPF routing protocols. The example advertises OSPF-derived routes using the RIP protocol and assigns the OSPF-derived routes a RIP metric of 10.

```
ProCurve(config)#router rip
ProCurve(config-rip)#default-metric 10
ProCurve(config-rip)#redistribute ospf
```

network <address> <subnet mask>

Use the **network** command to enable RIP on the specified network. The Secure Router OS will only allow processing (sending and receiving) RIP messages on interfaces with IP addresses that are contained in the networks listed using this command. All RIP messages received on interfaces not listed using this command will be discarded. To allow for receiving and participating in RIP but not for transmitting, use the **passive-interface** command (see *passive-interface* <interface> [on page 965](#)). Use the **no** form of this command to remove a network from the list.

Syntax Description

<address>	Specifies IP address of the network on which RIP will be enabled.
<subnet mask>	Specifies subnet mask that corresponds to the entered IP address.

Default Values

By default, RIP is not enabled.

Usage Examples

The following example enables RIP on the 102.22.72.252/30, 192.45.2.0/24, and 10.200.0.0/16 networks:

```
ProCurve(config)#router rip
ProCurve(config-rip)#network 102.22.72.252 255.255.255.252
ProCurve(config-rip)#network 192.45.2.0 255.255.255.0
ProCurve(config-rip)#network 10.200.0.0 255.255.0.0
```

passive-interface <interface>

Use the **passive-interface** command to disable the transmission of routing updates on the specified interface. All routing updates received on that interface will still be processed (and advertised to other interfaces), but no updates will be transmitted to the network connected to the specified interface. Multiple **passive-interface** commands may be used to create a customized list of interfaces. Use the **no** form of this command to enable the transmission of routing updates on an interface.

Syntax Description

<interface>	Specifies the interface that will not transmit routing updates.

Default Values

By default, RIP is not enabled.

Usage Examples

The following example disables routing updates on the Frame Relay link (labeled 1.17) and the PPP link (labeled 1):

```
ProCurve(config)#router rip  
ProCurve(config-rip)#passive-interface frame-relay 1.17  
ProCurve(config-rip)#passive-interface ppp 1
```

redistribute connected [metric <value>]

Use the **redistribute connected** command to pass routes from one network to another, regardless of the routing protocol implemented on the routing domain. Using the **connected** keyword allows the propagation of routes connected to other interfaces using the RIP routing protocol. Use the **no** form of this command to disable the propagation of the specified route type.

Syntax Description

metric <value>	Optional. Specifies the hop count to use for advertising redistributed OSPF routes in RIP.

Default Values

By default, RIP is not enabled.

Functional Notes

Redistributing connected routes imports those routes into RIP without the interfaces in question actually participating in RIP. The connected routes imported this way are not covered by a network command and therefore do not send/receive RIP traffic.

Usage Examples

The following example passes the connected routes found in the route table to other networks running the RIP routing protocol:

```
ProCurve(config)#router rip  
ProCurve(config-rip)#redistribute connected
```

redistribute ospf [metric <value>]

Use the **redistribute ospf** command to advertise routes from one protocol to another, regardless of the routing protocol implemented on the routing domain. Using the **ospf** keyword allows the propagation of OSPF routes into RIP. Use the **no** form of this command to disable the propagation of the specified route type.

Syntax Description

metric <value>	Optional. Specifies the hop count to use for advertising redistributed OSPF routes in RIP.

Default Values

By default, this command is disabled.

Functional Notes

Redistributing OSPF routes imports those routes into RIP without the interfaces in question actually participating in RIP. The OSPF routes imported this way are not covered by a network command and therefore do not send/receive RIP traffic.

If **redistribute ospf** is enabled and no metric value is specified, the value defaults to **0**. The metric value defined using the **redistribute ospf metric** command overrides the **default-metric** command's metric setting. See the section *default-metric <value>* [on page 963](#) for more information.

Usage Examples

The following example imports OSPF routes into RIP:

```
ProCurve(config)#router rip
```

```
ProCurve(config-rip)#redistribute ospf
```

redistribute static [metric <value>]

Use the **redistribute static** command to pass routes from one network to another, regardless of the routing protocol implemented on the routing domain. Using the **static** keyword allows the propagation of static routes to other interfaces using the RIP routing protocol. Use the **no** form of this command to disable the propagation of the specified route type.

Note *The gateway network for the static route must participate in RIP by using the network command for the gateway network.*

Syntax Description

metric <value>	Optional. Specifies the hop count to use for advertising redistributed OSPF routes in RIP

.

Default Values

By default, RIP is not enabled.

Functional Notes

Redistributing static routes allows other network devices to learn about paths (not compatible with their system) without requiring manual input to each device on the network.

version [1 | 2]

Use the **version** command to specify (globally) the Routing Information Protocol (RIP) version used on all IP interfaces. This global configuration is overridden using the configuration commands **ip rip send version** and **ip rip receive version**. Use the **no** form of this command to return to the default value.

Syntax Description

1	RIP version 1
2	RIP version 2

Default Values

By default, RIP is not enabled.

Usage Examples

The following example specifies RIP version 2 as the global RIP version:

```
ProCurve(config)#router rip  
ProCurve(config-rip)#version 2
```

ROUTER (OSPF) CONFIGURATION COMMAND SET

To activate the Router (OSPF) Configuration mode, enter the **router ospf** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable
ProCurve#configure terminal
ProCurve(config)#router ospf
ProCurve(config-ospf)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

area <area id> *default-cost* <value> [on page 971](#)

area <area id> *range* <ip address> <network mask> [*advertise* | *not-advertise*] [on page 972](#)

area <area id> *stub* [*no-summary*] [on page 973](#)

auto-cost *reference-bandwidth* <rate> [on page 974](#)

default-information-originate [*always*] [*metric* <value>] [*metric-type* type] [on page 975](#)

default-metric <value> [on page 976](#)

network <ip address> <wildcard> *area* <area id> [on page 977](#)

redistribute *connected* [on page 978](#)

redistribute *rip* [on page 979](#)

redistribute *static* [on page 980](#)

summary-address <address> <mask | prefix mask> *not-advertise* [on page 981](#)

timers *lsa-group-pacing* <seconds> [on page 982](#)

timers *spf* <delay> <hold> [on page 983](#)

area <area id> default-cost <value>

Use the **area default-cost** command to assign a cost of the default summary route sent into a stub area or not-so-stubby-area (NSSA). Use the **no** form of this command to delete the assigned cost.

Syntax Description

<area id>	Specifies identifier for this area. Enter as an integer (range: 0 to 4,294,967,295) or an IP address <A.B.C.D>.
<value>	Specifies default summary route cost. Range: 0 to 166,777,214.

Default Values

By default, the summary route cost is set to 0. There is no default for the area ID.

Usage Examples

The following example defines a default cost of 85 to a specific area:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#area 192.22.72.0 default-cost 85
```

area <area id> range <ip address> <network mask> [advertise | not-advertise]

Use the **area range** command to configure area route summarizations and to determine whether an address range is advertised to the networks.

Syntax Description

<area id>	Specifies identifier for this area. Enter as an integer (range: 0 to 4,294,967,295) or an IP address <A.B.C.D>.
<ip address>	The IP address of the advertised summary route.
<network mask>	The mask of the advertised summary route.
advertise	The specified address range will be advertised to other networks.
not-advertise	The specified address range will not be advertised to other networks.

Default Values

By default, OSPF is not enabled.

Usage Examples

The following example defines an address range for a specific area and allows the unit to advertise this range to other networks.

```
ProCurve(config)#router ospf
```

```
ProCurve(config-ospf)#area 11.0.0.0 range 11.0.0.0 255.0.0.0 advertise
```

area <area id> stub [no-summary]

Use the **area stub** command to configure an area as a stub area. Use the **no** form of this command to disable stub-designation for areas defined as stubs using this command.

Syntax Description

<area id>	Specifies identifier for this stub area. Enter as an integer (range: 0 to 4,294,967,295) or an IP address <A.B.C.D>.
no-summary	Optional. Use this optional keyword to designate the area as a total stub area. No summary link advertisements will be sent by the ABR into the stub area.

Default Values

By default, OSPF is not enabled.

Technology Review

It is important to coordinate configuration of all routers and access servers in the stub area. The **area stub** command must be configured for each of those pieces of equipment. Use the **area router configuration** command with the **area default-cost** command to specify the cost of a default internal router sent into a stub area by an ABR. See *area <area id> default-cost <value>* on page 971 for related information.

Usage Examples

The following example configures area 2 as a stub area.

```
ProCurve(config)#router ospf
ProCurve(config-ospf)#area 2 stub
```

auto-cost reference-bandwidth <rate>

Use the **auto-cost reference-bandwidth** command to assign a different interface cost to an interface. It may be necessary to assign a higher number to high-bandwidth links. This value is used in OSPF metric calculations.

Syntax Description

<rate>	Sets the default reference-bandwidth rate (range: 1 to 4,294,967 Mbps).

Default Values

By default, the rate is set to 100.

Usage Examples

The following example sets the auto cost reference-bandwidth to 1000 Mbps:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#auto-cost reference-bandwidth 1000
```

default-information-originate [always] [metric <value>] [metric-type <type>]

Use the **default-information-originate** command to cause an ASBR to generate a default route. It must have its own default route before it generates one unless the **always** keyword is used.

Syntax Description

always	Specifies to always advertise default route.
metric <value>	Configures metric value (range is 0 to 16,777,214).
metric type <type>	Configures metric type (1 or 2).

Default Values

metric <value>	10
metric type <type>	2

Usage Examples

The following example configures a router to always advertise default routes, and assigns the default routes a metric value of 10000 and a metric type of 2.

```
ProCurve(config)#router ospf
```

```
ProCurve(config-ospf)#default-information-originate always metric 10000 metric-type 2
```

default-metric <value>

Use the **default-metric** command to set a metric value for redistributed routes.

Syntax Description

<value>	Sets the default metric value (range: 0 to 4,294,967,295).

Default Values

By default, this value is set at 20.

Functional Notes

The metric value defined using the **redistribute** command overrides the **default-metric** command's metric setting. See *redistribute ospf [metric <value>]* on page 967 for related information.

Usage Examples

The following example shows a router using both RIP and OSPF routing protocols. The example advertises RIP-derived routes using the OSPF protocol and assigns the RIP-derived routes an OSPF metric of 10.

ProCurve(config)#**router ospf**

ProCurve(config-ospf)#**default-metric 10**

ProCurve(config-ospf)#**redistribute rip**

network *<ip address>* *<wildcard>* **area** *<area id>*

Use the **network area** command to enable routing on an IP stack and to define area IDs for the interfaces on which OSPF will run. Use the **no** form of this command to disable OSPF routing for interfaces defined using this command.

Syntax Description

<i><ip address></i>	Network address <A.B.C.D>.
<i><wildcard></i>	The wildcard mask is in an IP-address-type format and includes “don’t care” bits.
<i><area id></i>	Specifies identifier for this area. Enter as an integer (range: 0 to 4,294,967,295) or an IP address <A.B.C.D>.

Default Values

No default values required for this command.

Technology Review

In order for OSPF to operate on an interface, the *primary* address for the interface must be included in the **network area** command. Assigning an interface to an OSPF area is done using the **network area** command. There is no limit to the number of **network area** commands used on a router. If the address ranges defined for different areas overlap, the first area in the **network area** command list is used and all other overlapping portions are disregarded. Try to avoid overlapping to avoid complications.

Usage Examples

In the following example, the OSPF routing process is enabled and two OSPF areas are defined:

```
ProCurve(config)#router ospf
ProCurve(config-ospf)#network 192.22.72.101 0.0.0.255 area 0
ProCurve(config-ospf)#network 10.0.0.0 0.255.255.255 area 10.0.0.0
```

redistribute connected

Use the **redistribute connected** command to advertise routes from one protocol to another. Using the **connected** keyword allows the advertisement of connected routes into the OSPF routing protocol. This will advertise all connected routes on OSPF-enabled interfaces. It does not enable OSPF on all interfaces. Use the **no** form of this command to disable the propagation of the specified route type.

Syntax Description

connected	Optional. Keyword that specifies the Secure Router OS to advertise connected routes to OSPF areas.

Default Values

By default, this command is disabled.

Functional Notes

Redistributing connected routes imports those routes into OSPF without the interfaces in question actually participating in OSPF. The connected routes imported this way are not covered by a network command and therefore do not send/receive OSPF traffic.

Usage Examples

The following example imports connected routes into OSPF:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#redistribute connected
```

redistribute rip

Use the **redistribute rip** command to advertise routes from one protocol to another, regardless of the routing protocol implemented on the routing domain. Using the **rip** keyword allows the propagation of RIP routes into OSPF. Use the **no** form of this command to disable the propagation of the specified route type.

Syntax Description

rip	Optional. Keyword that specifies the Secure Router OS to import RIP routes into OSPF.

Default Values

By default, this command is disabled.

Functional Notes

Redistributing RIP routes imports those routes into OSPF without the interfaces in question actually participating in OSPF. The RIP routes imported this way are not covered by a network command and therefore do not send/receive OSPF traffic.

Usage Examples

The following example imports RIP routes into OSPF:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#redistribute rip
```

redistribute static

Use the **redistribute static** command to advertise routes from one protocol to another. Using the **static** keyword allows the advertisement of static routes into the OSPF routing protocol. This will advertise all static routes on OSPF-enabled interfaces. It does not enable OSPF on all interfaces. Use the **no** form of this command to disable the propagation of the specified route type.

Syntax Description

static	Optional. Keyword that specifies the Secure Router OS to import static routes into OSPF.

Default Values

By default, this command is disabled.

Functional Notes

Redistributing static routes imports those routes into OSPF without the interfaces in question actually participating in OSPF. The static routes imported this way are not covered by a network command and therefore do not send/receive OSPF traffic.

Usage Examples

The following example imports static routes into OSPF:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#redistribute static
```

summary-address <address> <mask | prefix mask> not-advertise

Use this command to control address summarization of routes that are redistributed into OSPF from other sources (e.g., RIP-to-OSPF, static-to-OSPF, etc.). The **not-advertise** option causes suppression of routes that match the specified mask/prefix mask pair.

Syntax Description

<address>	Specifies IP address or Prefix A.B.C.D.
<mask prefix mask>	Routes matching this mask/prefix mask pair will be suppressed if the not-advertise command is enabled.
not advertise	Optional. Causes suppression of routes that match the specified mask/prefix mask pair.

Default Values

By default, this command is disabled.

Usage Examples

The following example suppresses advertisement of the routes which match the specified address/mask:

```
ProCurve(config)#router ospf
```

```
ProCurve(config-ospf)#summary-address 11.0.0.0 255.0.0.0 not-advertise
```

timers lsa-group-pacing <seconds>

Use the **timers lsa-group-pacing** command to change the link state advertisement (LSA) refresh interval.

Syntax Description

<seconds>	Sets the LSA refresh interval in seconds (range: 10-1,800).

Default Values

By default, this value is set at 240 seconds.

Usage Examples

The following example sets the refresh interval for six minutes:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#timers lsa-group-pacing 360
```

timers spf *<delay>* *<hold>*

Use the **timers spf** command to configure the shortest path first (SPF) calculation and hold intervals.

Syntax Description

<i><delay></i>	Specifies time in seconds between OSPF's receipt of topology changes and the beginning of SPF calculations.
<i><hold></i>	Specifies time in seconds between consecutive SPF calculations. Range: 10-1800 seconds.

Default Values

<i><delay></i>	5 seconds
<i><hold></i>	10 seconds

Usage Examples

The following example defines a delay of 10 seconds and a hold-time of 30 seconds:

```
ProCurve(config)#router ospf  
ProCurve(config-ospf)#timers spf 10 30
```

QUALITY OF SERVICE (QoS) MAP COMMANDS

A QoS policy is defined using a QoS map in the CLI. The QoS map is a named list with sequenced entries. An entry contains a single match reference and one or more actions (priority, set, or both). To activate the QoS Command Set (which allows you to create and/or edit a map), enter a valid version of the QoS command at the Global Configuration mode prompt. Multiple map entries for the same QoS map are differentiated by a sequence number. The sequence number is used to assign match order.

Once created, a QoS map must be applied to an interface (using the **qos-policy out** *<map-name>* command) in order to actively process traffic. Any traffic for the interface that is not sent to the priority queue is sent using the default queuing method for the interface (such as weighted fair queuing).

For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#qos map VOICEMAP 10  
ProCurve(config-qos-map)#match precedence 5  
ProCurve(config-qos-map)#priority 512  
ProCurve(config-qos-map)#exit  
ProCurve(config)#interface fr 1  
ProCurve(config-fr 1)#qos-policy out VOICEMAP
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

bandwidth [*percent* *<value>* | *remaining percent* *<value>* | *<value>*] [on page 985](#)

match [on page 988](#)

priority [*<bandwidth>* | *<bandwidth><burst>* | *unlimited*] [on page 989](#)

set dscp *<0-63>* [on page 990](#)

set precedence *<0-7>* [on page 991](#)

bandwidth [percent <value> | remaining percent <value> | <value>]

Use the **bandwidth** command to specify bandwidth allocation for individual traffic classes (for CBWFQ configurations). Use the **no** form of this command to remove a configured bandwidth allocation.

Syntax Description

percent <value>	Allocates a minimum bandwidth for a traffic class, specifying the minimum as a percentage of the total interface bandwidth. See <i>Functional Notes</i> for more details.
remaining percent <value>	Allocates a minimum bandwidth for a traffic class, specifying the minimum as a percentage of the total interface bandwidth not allocated to priority classes in the QoS map. See <i>Functional Notes</i> for more details.
<value>	Allocates the minimum bandwidth for a traffic class, specifying the minimum as an absolute bandwidth in kilobits per second (Range: 8 to 2,000,000 kbps).

Default Values

By default, there is no bandwidth allocation configured for a QoS map entry.

Functional Notes

When configuring **bandwidth** allocations for CBWFQ, there are a few rules that must be obeyed.

1. The units of the bandwidth (kbps, percent, or remaining percent) must be consistent for all class-based entries (using the **bandwidth** command) in a QoS map set.
2. The total bandwidth between all **priority** entries and class-based entries (**bandwidth**) in a QoS map set should not be configured beyond the specified **max-reserved-bandwidth** (default 75 percent) on the interface that the QoS policy is applied to (using the **qos-policy** command), or the map will be disabled. Even with the configuration limit, class-based queues can still use more than the **max-reserved-bandwidth** limitation, up to 100% of the bandwidth, if enough traffic is present.

When the configured QoS map is applied to a physical interface, the Secure Router OS displays bandwidth information for the map and the physical interface. For example, the Frame Relay interface (**fr 1**) has been connected to the E1 interface (**e1 1/1**) using the **bind** command. Applying the QoS map (**MyMapA**) to the Frame Relay interface (**fr 1**) produces the following status message:

```
2005.08.09 07:28:22 QOS.INTERFACE QOS policy "MyMapA" requires 1288 kbps of bandwidth
and 1488 kbps is now available for interface fr 1 -> the QOS policy for this port has been forced
ACTIVE.
```

This status message displays the sum total of the bandwidths specified in the QoS map (in red above) and the available interface bandwidth using the total line rate configured on the interface (in blue above).

For all configurations using the ProCurve Secure Router dl 8xT1/E1 Wide Module the status message reflects the sum total of the bandwidths specified in the QoS map and the available interface bandwidth using the total available line rate for the interface (regardless of the channels specified by the TDM group).

For example, if the following QoS map is applied to a frame relay interface (**fr 1**) bound to an E1 interface (**e1 1/1**) on the ProCurve Secure Router dl 8xT1/E1 Wide Module:

```
qos map MyMap1 10
  match precedence 1
  priority 256
qos map MyMap1 20
  match precedence 2
  bandwidth percent 25
qos map MyMap1 30
  match precedence 3
  bandwidth percent 25
```

Then, the total bandwidth required (for the QoS map) is calculated as such: $256 + 0.25(1984) + 0.25(1984)$ for a total of 1248 kbps. The total available interface bandwidth is displayed as 1984 kbps for all E1 configurations on the wide module (regardless of the channels specified in the TDM group).

- Up to 4 class-based entries (**bandwidth** commands) can be configured in a particular QoS map set. Up to 16 class-based entries can be configured (four entries on four QoS maps).
- Within a QoS map entry, CBWFQ bandwidth and low latency priority actions are mutually exclusive. However, bandwidth and priority actions may be applied to different entries in the same QoS map.

Determining Bandwidth Entries

Note *When possible, use the **bandwidth** <value> command to specify an absolute amount of bandwidth (in kbps) for the traffic class.*

When determining the **percent** <value> entry, use the following formula:

$$\frac{\text{Bandwidth}}{\text{Line Rate}}$$

where

<i>Bandwidth</i>	minimum amount of bandwidth needed for the traffic (in kbps)
<i>Line Rate</i>	total data rate configured on the interface (for example, 8 DS0s (64 kbps per DS0) on a T1 equals a line rate of 512 kbps)

For example, to specify 80 kbps of data on an interface with a total of 512 kbps of available bandwidth, enter the following command:

ProCurve(config-qos-map 1)#**bandwidth percent 16**

When determine the **remaining percent** <value> entry, use the following formula:

$$\frac{\text{Bandwidth}}{\text{Line Rate} - \text{Priority Traffic}}$$

where

<i>Bandwidth</i>	minimum amount of bandwidth needed for the traffic (in kbps)
<i>Line Rate</i>	total data rate configured on the interface (for example, 8 DS0s (64 kbps per DS0) on a T1 equals a line rate of 512 kbps)
<i>Priority Traffic</i>	amount of bandwidth reserved using the priority command

For example, to specify 80 kbps of data on an interface with a total of 512 kbps of available bandwidth and 256 kbps reserved (using the **priority** command), enter the following command:

```
ProCurve(config-qos-map 1)#bandwidth remaining percent 31
```

Usage Examples

The following example creates a QoS map with 4 traffic classes (based on IP packet precedence values) and allocates bandwidth to each class:

```
ProCurve(config)#qos map MyMap 1  
ProCurve(config-qos-map)#match precedence 5  
ProCurve(config-qos-map)#bandwidth percent 25
```

```
ProCurve(config)#qos map MyMap 2  
ProCurve(config-qos-map)#match precedence 3  
ProCurve(config-qos-map)#bandwidth percent 10
```

```
ProCurve(config)#qos map MyMap 3  
ProCurve(config-qos-map)#match precedence 2  
ProCurve(config-qos-map)#bandwidth percent 10
```

```
ProCurve(config)#qos map MyMap 4  
ProCurve(config-qos-map)#match precedence 1  
ProCurve(config-qos-map)#bandwidth percent 15
```

match

Use the **match** command to specify which traffic should be processed by this QoS map. Possible variations of this command include:

```
match dscp <0-63>
match ip rtp <port #>
match ip rtp <first port # in range> <last port # in range>
match ip rtp <first port # in range> <last port # in range> all
match list <listname>
match precedence <0-7>
match protocol bridge
match protocol bridge netbeui
```

Syntax Description

all	Optional. Specifies matching all UDP port numbers in the specified range (even and odd). Valid only for ip rtp matches.
dscp <0-63>	Matches IP packets with the specified DSCP value.
ip rtp <start><end>	Matches RTP packets with even UDP destination port numbers in the specified range (between start and end).
list <listname>	Specifies the name of the access list (ACL) you wish to use to match packets for this QoS map. See <i>ip access-list extended</i> <listname> on page 288 for more information on creating access lists.
precedence <0-7>	Matches IP packets with the specified IP precedence value.
protocol bridge	Matches frames being bridged by the router.
protocol bridge netbeui	Matches only NetBEUI frames being bridged by the router.

Default Values

No default value is necessary for this command.

Usage Examples

The following example assigns a traffic match pattern to the existing QoS map **VOICEMAP**:

```
ProCurve(config)#qos map VOICEMAP 10
ProCurve(config-qos-map)#match ip rtp 16384 20000
```

priority [*<bandwidth>* | *<bandwidth><burst>* | unlimited]

The **priority** command provides a high-priority queue, prioritizing this traffic above all others. If no traffic is present in any other queue, priority traffic is allowed to burst up to the interface rate; otherwise, priority traffic above the specified bandwidth is dropped. Use the **no** form of this command to disable this feature.

Syntax Description

<i><bandwidth></i>	Specifies a low latency queue, prioritizing this traffic above all other user traffic in kilobits per second (Range: 8 to 1,000,000). If no traffic is present in any other queue, priority traffic is allowed to burst up to the interface rate. Otherwise, priority traffic above the specified bandwidth will be dropped.
<i><burst></i>	Optional. Specifies the maximum burst size (in bytes) for traffic in this priority queue. This parameter should be left unconfigured for optimal performance (Range: 3 to 1,000,000).
unlimited	Optional. Specifies no limits on the priority queue bandwidth. Excessive traffic matching the QoS map can potentially use all of the available bandwidth on the interface, so use this feature with caution.

Default Values

No default value is necessary for this command.

Functional Notes

The priority queue is intended for constant bit rate traffic such as voice due to the rate limiting. The sum of the bandwidths reserved by priority commands for all entries of a QoS map cannot exceed the **max-reserved-bandwidth** rate specified for the interfaces that the map is applied to.

Usage Examples

The following example assigns the matched traffic to a high priority output queue of 512 kbps for any interface using the **VOICEMAP 10** QoS map:

```
ProCurve(config)#qos map VOICEMAP 10
ProCurve(config-qos-map)#match ip rtp 16384 20000
ProCurve(config-qos-map)#priority 512
```

set dscp <0-63>

The **set dscp** command is an optional command for a QoS map that can be used to modify the DSCP field (on matching packets) to the specified value.

Syntax Description

<0-63>	Specifies the decimal DSCP value.

Default Values

No default value is necessary for this command.

Usage Examples

This command sets the DSCP value (for all matching traffic) to **46**:

```
ProCurve(config)#qos map VOICEMAP 10
```

```
ProCurve(config-qos-map)#set dscp 46
```

set precedence <0-7>

The **set precedence** command is an optional command for a QoS map that can be used to modify the IP precedence value (on matching packets) to the specified value.

Syntax Description

<0-7>	Specifies the decimal IP precedence value.

Default Values

No default value is necessary for this command.

Usage Examples

This command sets the IP precedence value (for all matching traffic) to **5**:

```
ProCurve(config)#qos map VOICEMAP 10
```

```
ProCurve(config-qos-map)#set precedence 5
```

AS PATH LIST COMMAND SET

To activate the Autonomous System (AS) Path List Configuration mode, enter the **ip as-path-list** command at the Global Configuration mode prompt. For example:

```
ProCurve>enable  
ProCurve#configure terminal  
ProCurve(config)#ip as-path-list listname  
ProCurve(config-as-path-list)#
```

The following commands are common to multiple command sets and are covered in a centralized section of this guide. For more information, refer to the sections listed below:

do [on page 1001](#)
end [on page 1002](#)
exit [on page 1003](#)

All other commands for this command set are described in this section in alphabetical order.

deny [on page 993](#)
permit [on page 994](#)

deny

Use the **deny** command to deny any BGP route that matches the as-path attributes.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example denies BGP routes that match as-path attributes:

```
ProCurve(config)#ip as-path-list listname  
ProCurve(config-as-path-list)#deny
```

permit

Use the **permit** command to allow all BGP routes that matches the as-path attributes.

Syntax Description

No subcommands.

Default Values

No default value necessary for this command.

Usage Examples

The following example permits BGP routes that match the as-path attributes:

```
ProCurve(config)#ip as-path-list listname
ProCurve(config-as-path-list)#permit
```

COMMON COMMANDS

The following section contains descriptions of commands which are common across multiple command sets. These commands are listed in alphabetical order.

alias <"text"> [on page 996](#)

bind <#> <from interface> <slot/port> <tdm-group#> <to interface> <slot/port> [on page 997](#)

description <text> [on page 1000](#)

do [on page 1001](#)

end [on page 1002](#)

exit [on page 1003](#)

ping <address> [on page 1004](#)

show running-config [on page 1006](#)

shutdown [on page 1008](#)

alias <*“text”*>

Use the **alias** command to populate the ifAlias OID (Interface Table MIB of RFC2863) for all physical and virtual interfaces when using Simple Network Management Protocol (SNMP) management stations.

Syntax Description

<i><input></i>	Alphanumeric character string (no more than 64 characters) describing the interface (for SNMP) — must be encased in quotation marks.

Default Values

No defaults required for this command.

Functional Notes

The ifAlias OID is a member of the ifXEntry object-type (defined in RFC2863) used to provide a non-volatile, unique name for various interfaces. This name is preserved through power cycles. Enter a string (using the **alias** command) which clearly identifies the interface.

Usage Examples

The following example defines a unique character string for the T1 interface:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#alias "CIRCUIT_ID_23-908-8887-401"
```

Technology Review

Please refer to RFC2863 for more detailed information on the ifAlias display string.

bind <#> <from interface> <slot/port> <tdm-group#> <to interface>
<slot/port>

Use the **bind** command to create a bind map from a created tdm-group on an interface to a virtual interface.

Caution *Changing **bind** settings could potentially result in service interruption.*

Syntax Description

<#>	Number descriptor or label for identifying the bind (useful in systems that allow multiple binds)
<from interface>	Specifies the interface (physical or virtual) on one end of the bind. Enter bind 1 ? for a list of valid interfaces.
<slot/port>	Used when a physical interface is specified in the <from interface> subcommand (For example: specifying the T1 port of a T1 module would be t1 1/1).
<tdm-group#>	Specifies which configured tdm-group to use for this bind. This subcommand only applies to T1 physical interfaces.
<to interface>	Specifies the virtual interface on the other end of the bind. Use the ? to display a list of valid interfaces.
<slot/port>	Used when a physical interface is specified in the <to interface> subcommand. (For example, specifying the primary T1 port of a T1 module would be t1 1/1).
<rbs >	This optional field is used in order to maintain robbed bit signaling through the bind when voice is being delivered.

Default Values

By default, there are no configured binds.

Functional Notes

Binds provide the mechanism for binding a configured virtual (layer 2) endpoint with a physical (layer 1) interface. Supported layer 2 protocols include Frame Relay and point-to-point protocol (PPP).

Usage Examples

The following example creates a Frame Relay endpoint and binds it to the t1 1/1 physical interface:

1. Create the Frame Relay virtual endpoint and set the signaling method:

```
ProCurve(config)#interface frame-relay 1  
ProCurve(config-fr 1)#frame-relay lmi-type cisco
```

2. Create the sub-interface and configure the PVC parameters (including DLCI and IP address):

```
ProCurve(config-fr 1)#interface fr 1.1  
ProCurve(config-fr 1.1)#frame-relay interface-dlci 17  
ProCurve(config-fr 1.1)#ip address 168.125.33.252 255.255.255.252
```

3. Create the tdm-group of 12 DS0s (64K) on the t1 physical interface:

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#tdm-group 1 timeslots 1-12 speed 64  
ProCurve(config-t1 1/1)#exit
```

4. Connect the Frame Relay sub-interface with port t1 1/1:

```
ProCurve(config)#bind 1 t1 1/1 1 fr 1
```

Technology Review

Creating an endpoint that uses a layer 2 protocol (such as Frame Relay) is generally a four-step process:

Step 1:

Create the Frame Relay virtual endpoint (using the **interface frame-relay** command) and set the signaling method (using the **frame-relay lmi-type** command). Also included in the Frame Relay virtual endpoint are all the applicable Frame Relay timers logging thresholds, encapsulation types, etc. Generally, most Frame Relay virtual interface parameters should be left at their default state. For example, the following creates a Frame Relay interface labeled **7** and sets the signaling method to **ansi**.

```
ProCurve(config)#interface frame-relay 7  
ProCurve(config-fr 7)#frame-relay lmi-type ansi
```

Step 2:

Create the sub-interface and configure the PVC parameters. Using the sub-interface, apply access policies to the interface, create bridging interfaces, configure backup, assign an IP address, and set the PVC data-link control identifier (DLCI). For example, the following creates a Frame Relay sub-interface labeled **22**, sets the DLCI to **30**, and assigns an IP address of **193.44.69.253** to the interface.

```
ProCurve(config-fr 7)#interface fr 7.22  
ProCurve(config-fr 7.22)#frame-relay interface-dlci 30  
ProCurve(config-fr 7.22)#ip address 193.44.69.253 255.255.255.252
```

Step 3:

Specify the group of DS0s used for signaling on the T1 interface by creating a **tdm-group**. Group any number of contiguous DS0s together to create a data pipe for layer 2 signaling. Also use the **tdm-group** command to specify the per-DS0 signaling rate on the interface. For example, the following creates a tdm-group labeled **9** containing 20 DS0s (each DS0 having a data rate of 56 kbps).

```
ProCurve(config)#interface t1 1/1  
ProCurve(config-t1 1/1)#tdm-group 9 timeslots 1-20 speed 56  
ProCurve(config-t1 1/1)#exit
```

Step 4:

Make the association between the layer 2 endpoint and the physical interface using the **bind** command. Supported layer 2 protocols include Frame Relay and point-to-point protocol (PPP). For example, the following creates a bind (labeled **5**) to make an association between the Frame Relay virtual interface (**fr 7**) and the tdm-group configured on interface t1 1/1 (**tdm-group 9**).

```
ProCurve(config)#bind 5 t1 1/1 9 fr 7
```

description *<text>*

Use the **description** command to identify the specified interface, both physical and virtual (for example, circuit ID, contact information, etc.).

Syntax Description

<code><text></code>	Identifies the specified interface using up to 80 alphanumeric characters.

Default Values

No defaults required for this command.

Usage Examples

The following example enters comment information using the **description** command:

```
ProCurve(config)#interface t1 1/1
```

```
ProCurve(config-t1 1/1)#description This is the Dallas office T1
```


do

Use the **do** command to execute any command, regardless of the active configuration mode.

Syntax Description

No subcommands.

Default Values

No defaults required for this command.

Functional Notes

Use the **do** command to view configurations or interface states after configuration changes are made without exiting to the Enable mode.

Usage Examples

The **do** command provides a way to execute commands in other configuration modes without taking the time to exit the current configuration mode and enter the desired one. The following example shows the **do** command used to remove all dynamic entries from the ARP cache:

```
ProCurve(config)#do clear arp-cache
```

end

Use the **end** command to exit the current configuration mode and enter the Enable Security mode.

Note	<i>When exiting the Global Configuration mode, remember to perform a copy running-config startup-config to save all configuration changes.</i>

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following example shows the **end** command being executed in the T1 Configuration mode:

```
ProCurve(config-t1 1/1)#end
```

```
#
```

```
#- Enable Security mode command prompt
```

exit

Use the **exit** command to exit the current Configuration mode and enter the previous one. For example, using the **exit** command in an interface configuration mode will activate the Global Configuration mode. When using the **exit** command in the Basic mode, the current session will be terminated.

Note	<i>When exiting the Global Configuration mode, remember to perform a copy running-config startup-config to save all configuration changes.</i>

Syntax Description

No subcommands.

Default Values

No defaults necessary for this command.

Usage Examples

The following example shows the **exit** command being executed in the Global Configuration mode:

```
ProCurve(config)#exit  
#
```

- Enable Security mode command prompt

ping <address>

Use the **ping** command (at the Enable Command mode prompt) to verify IP network connectivity.

Syntax Description

<address>	Optional. Specifies the IP address of the system to ping. Entering the ping command with no specified address prompts the user with parameters for a more detailed ping configuration. See Functional Notes (below) for more information.

Default Values

No default value necessary for this command.

Functional Notes

The **ping** command helps diagnose basic IP network connectivity using the Packet InterNet Groper program to repeatedly bounce Internet Control Message Protocol (ICMP) Echo_Request packets off a system (using a specified IP address). The Secure Router OS allows executing a standard **ping** request to a specified IP address or provides a set of prompts to configure a more specific **ping** configuration.

The following is a list of output messages from the **ping** command:

!	Success
-	Destination Host Unreachable
\$	Invalid Host Address
X	TTL Expired in Transit
?	Unknown Host
*	Request Timed Out

The following is a list of available extended **ping** fields with descriptions:

Target IP address:	Specifies the IP address of the system to ping.
Repeat Count:	Number of ping packets to send to the system (valid range: 1 to 1000000).
Datagram Size:	Size (in bytes) of the ping packet (valid range: 1 to 1448).
Timeout in Seconds:	If a ping response is not received within the timeout period, the ping is considered unsuccessful (valid range: 1 to 5 seconds).
Extended Commands:	Specifies whether additional commands are desired for more ping configuration parameters.
Source Address	Specifies the IP address to use as the source address in the ECHO_REQ (or interface) packets.
Data Pattern:	Specifies an alphanumerical string to use (the ASCII equivalent) as the data pattern in the ECHO_REQ packets.

Sweep Range of Sizes:	Varies the sizes of the ECHO_REQ packets transmitted.
Sweep Min Size:	Specifies the minimum size of the ECHO_REQ packet (valid range: 0 to 1488).
Sweep Max Size:	Specifies the maximum size of the ECHO_REQ packet (valid range: Sweep Min Size to 1448).
Sweep Interval:	Specifies the interval used to determine packet size when performing the sweep (valid range: 1 to 1448).
Verbose Output:	Specifies an extended results output.

Usage Examples

The following is an example of a successful **ping** command:

#ping

Target IP address:**192.168.0.30**

Repeat count[1-1000000]:**5**

Datagram Size [1-1000000]:**100**

Timeout in seconds [1-5]:**2**

Extended Commands? [y or n]:**n**

Type CTRL+C to abort.

Legend: '!' = Success '?' = Unknown host '\$' = Invalid host address

'*' = Request timed out '-' = Destination host unreachable

'x' = TTL expired in transit

Pinging 192.168.0.30 with 100 bytes of data:

!!!!

Success rate is 100 percent (5/5) round-trip min/avg/max = 19/20.8/25 ms

show running-config

Use the **show running-config** command to display a text print of all the non-default parameters contained in the current running configuration file. Use the **verbose** keyword to display a text print of the entire configuration (including parameters in their default state). Specific portions of the running-config may be displayed, based on the command entered.

Variations of this command include the following:

```
show running-config
show running-config access-lists
show running-config access-lists verbose
show running-config checksum
show running-config interface <interface>
show running-config interface <interface> verbose
show running-config interface vlan <vlan id>
show running-config interface vlan <vlan id> verbose
show running-config policy-class
show running-config policy-class verbose
show running-config qos-map
show running-config qos-map verbose
show running-config verbose
```

Syntax Description

access-lists	Displays the current running configuration for all configured IP access lists.
interface <interface>	Displays the current running configuration for a particular interface. Type the show running-config interface ? command to display a list of valid interfaces.
policy-class	Displays the current running configuration for all configured policy classes.
qos-map	Displays the current running configuration for all configured QoS maps.
verbose	Displays the entire running configuration to the terminal screen (versus only the <i>*Optional</i> non-default values).
checksum	Displays the encrypted Message Digest 5 (md5) version of the running <i>*Optional</i> configuration.

Default Values

No default value necessary for this command.

Usage Examples

The following is a sample output from the **show running-config** command:

```
>enable
#show running-config
Building configuration...
!
no enable password
!
ip subnet-zero
ip classless
ip routing
!
event-history on
no logging forwarding
logging forwarding priority-level info
no logging email
!
ip policy-timeout tcp all-ports 600
ip policy-timeout udp all-ports 60
ip policy-timeout icmp 60
!
interface eth 0/1.....
```

shutdown

Use the **shutdown** command to disable the interface (both physical and virtual) so that no data will be passed through. Use the **no** form of this command to turn on the interface and allow it to pass data. By default, all interfaces are disabled.

Syntax Description

No subcommands.

Default Values

By default, all interfaces are disabled.

Usage Examples

The following example administratively disables the modem interface:

```
ProCurve(config)#interface modem 1/2  
ProCurve(config-modem 1/2)#shutdown
```


Index

A

- aaa authentication 238
- aaa authentication enable default 240
- aaa authentication login 241
- aaa group server 503
- aaa group server radius 242
- aaa on 243
- aaa processes 245
- able 11
- access-class in 955
- access-policy 413, 523, 664, 726, 801, 853, 917
- ADSL Interface Configuration command set 776
- advertisement-interval 787
- alias 996
- alias link 856
- anning-tree mode 390
- antireplay 484, 493
- area default-cost 971
- area range 972
- area stub 973
- arp arpa 246, 526
- as-path-list 788
- ATM Interface Configuration command set 722
- atm routed-bridged ip 729
- ATM Sub-Interface Config command set 725
- attribute 463, 473
- authentication 474
- auto cost reference-bandwidth 974
- autosynch 22, 247

B

- backup auto-backup 667, 730, 857
- backup auto-restore 417, 668, 731, 805, 858
- backup backup-delay 418, 669, 732, 806, 859
- backup call-mode 419, 670, 733, 807, 860
- backup connect-timeout 422, 673, 736, 810, 863
- backup force 423, 674, 737, 811, 864
- backup maximum retry 424, 675, 812, 865
- backup number 425, 676, 813, 866
- backup priority 426, 677, 814, 867
- backup randomize-timers 427, 678, 815, 868
- backup redial-delay 428, 679, 816, 869
- backup restore-delay 429, 680, 817, 870
- backup schedule 430, 681, 818, 871
- backup shutdown 431, 682, 819, 872
- bandwidth 432, 527, 644, 683, 738, 820, 873, 920, 985
- banner 249
- basic 4
- Basic Mode command set 10
- bgp 780

- BGP Configuration command set 780
- bgp fast-external-fallover 781
- bgp log-neighbor-changes 782
- BGP Neighbor Configuration command set 786
- bgp router-id 783
- bgp-neighbor 786
- bind 997
- bonding txadd-timer 631
- bonding txcid-timer 632
- bonding txdeq-timer 633
- bonding txfa-timer 634
- bonding txinit-timer 635
- bonding txnull-timer 636
- boot config 250, 251
- BRI Interface Configuration command set 630
- bridge protocol ieee 252
- bridge-group 433, 528, 684, 739, 874

C

- CA Profile command set 505
- caller-id override 627, 637
- certificate 517
- certificate ca 518
- Certificate Configuration command set 516
- clear access-list 24
- clear arp-cache 25
- clear arp-entry 26
- clear bridge 27
- clear buffers 28
- clear counters 29
- clear crypto ike sa 30
- clear crypto ipsec sa 31
- clear dump-core 32
- clear event-history 33
- clear host 34
- clear ip bgp 35
- clear ip dhcp-server binding 36
- clear ip igmp group 37
- clear ip policy-sessions 38
- clear ip policy-stats 40
- clear ip prefix-list 41
- clear ip route 42
- clear lldp counters 43
- clear lldp counters interface 44
- clear lldp neighbors 45
- clear pppoe 46
- clear process cpu max 47
- clear qos map 48
- clear spanning-tree counters 49
- clear spanning-tree detected-protocols 50

CLI

- accessing with PC 4
- error messages 8
- introduction 4
- shortcuts 6
- client authentication host 464
- client authentication host xauth-type 465
- client authentication server list 466
- client configuration pool 467
- client-identifier 395
- client-name 397
- clock auto-correct-dst 51
- clock no-auto-correct-dst 52
- clock set 53
- clock source 580, 605
- clock timezone 54
- coding 581, 595, 606, 620
- command descriptions 9
- command level path 6
- Command Line Interface
 - accessing with PC 4
 - error messages 8
 - shortcuts 6
- command security levels
 - basic 4
 - enable 4
- common CLI functions 7
- common commands 995
- configuration 236
- configuration modes
 - global 5
 - interface 5
 - line 5
 - router 5
- configure 56
- connected 966, 978
- console port
 - configuring 4
 - receiving files 60, 64, 68, 71, 73
- copy boot 57
- copy cflash 58
- copy cflash tftp 59
- copy cflash xmodem 60
- copy console 61
- copy flash 62
- copy flash tftp 63
- copy flash xmodem 64
- copy interface 65
- copy running-config 66, 69
- copy running-config tftp 67
- copy running-config xmodem 68
- copy startup-config tftp 70
- copy startup-config xmodem 71

- copy tftp 72
- copy xmodem 73
- crl 519
- crl optional 506
- crypto ca authenticate 253
- crypto ca certificate chain 255
- crypto ca enroll 256
- crypto ca import certificate 257
- crypto ca import crl 258
- crypto ca profile 259
- crypto ike 260
- crypto ike policy 462
- crypto ike remote-id 264
- crypto ipsec transform-set 266
- crypto map 267, 434, 529, 685, 740, 821, 875, 921
- Crypto Map IKE command set 483
- crypto map ipsec-ike 483
- crypto map ipsec-manual 492
- Crypto Map Manual command set 492

D

- databits 944
- debug 8
- debug aaa 74
- debug access-list 75
- debug atm events 76
- debug atm oam 77
- debug atm packet 78
- debug backup 79
- debug crypto 81
- debug dialup-interfaces 82
- debug dynamic-dns 83
- debug firewall 84
- debug frame-relay 85
- debug frame-relay multilink 86
- debug hdlc 87
- debug interface 88
- debug interface adsl events 89
- debug ip bgp 90
- debug ip dhcp-client 91
- debug ip dhcp-server 92
- debug ip dns-client 93
- debug ip dns-proxy 94
- debug ip icmp 95
- debug ip igmp 96
- debug ip mrouting 97
- debug ip ospf 98
- debug ip rip 99
- debug ip routing 100
- debug ip tcp events 101
- debug ip tcp md5 102
- debug ip udp 103

debug isdn events 104
debug lldp 105
debug port-auth 106
debug ppp 107
debug pppoe client 108
debug radius 109
debug snmp 110
debug spanning-tree 111
debug spanning-tree bpdu 112
debug system 113
debug-bridge 80
default-information-originate 975
default-metric 963, 976
default-router 398
description 1000
DHCP Pool command set 394
dialin 628
dir 114, 115
disable, basic mode 11
disable, enable mode 116
distance bgp 784
dns-server 399, 480
do 1001
domain-name 400
DSX-1 Interface Configuration command set 594
dynamic-dns 436, 531, 687, 742, 823, 877, 923
E
E1 Interface Configuration command set 604
ebgp-multihop 789
email address 507
enable 11
enable password 269
enable, basic mode 11
enable, enable mode 20
enable, understanding 4
encapsulation 744
encapsulation 802.1q 533
encapsulation frame-relay ietf 645
encryption 475
end 1002
enrollment retry 508
enrollment terminal 509
enrollment url 510
erase 117
erase cflash 118
erase file-system cflash 119
erase flash 118
et-clock-source 571
Ethernet Interface Configuration command set 520
Ethernet Sub-Interface Configuration command set 520
event-history on 270

event-history priority 271
events 120
exception report 273
exception report generate 121
exit 1003
F
fair-queue 438, 646, 745, 879
fdl 582
flowcontrol 945
fqdn 511
Frame Relay Interface Configuration command set 643
Frame Relay Sub-Interface Config command set 663
frame-relay bc 689
frame-relay be 690
frame-relay fragment 691
frame-relay interface-dlci 692
frame-relay intf-type 647
frame-relay lmi-n391dce 648
frame-relay lmi-n391dte 649
frame-relay lmi-n392dce 650
frame-relay lmi-n392dte 651
frame-relay lmi-n393dce 652
frame-relay lmi-n393dte 653
frame-relay lmi-t391dte 654
frame-relay lmi-t392dce 655
frame-relay lmi-type 656
frame-relay multilink 657
framing 583, 596, 607, 621
ftp authentication 274
full-duplex 534
G
G.703 Interface Configuration command set 619
Global Configuration Mode command set 236
group 476
H
half-duplex 535
hardware-address 401
hash 477
HDLC Configuration command set 411
hold-queue 439, 659, 746, 880
hold-timer 785, 790
host 402
hostname 275
I
ignore dcd 572
IKE Client command set 479
IKE Policy Attributes command set 473
IKE Policy command set 462
ike-policy 485, 494
initiate 468
init-string 629

interface 276
interface adsl 776
interface atm 722, 725
interface bri 630
interface e1 604
interface ethernet 520
interface ethernet sub 520
interface frame-relay 277, 643, 663
interface G.703 619
interface hdlc 411
interface hdlc 279
interface loopback 281, 916
interface modem 626
interface ppp 282, 851
interface range 520
interface serial 570
interface t1 579, 594
interface tunnel 284, 800
invert etclock 573
invert rxclock 574
invert txclock 575
ip access-group 440, 536, 693, 747, 825, 883, 925
ip access-list extended 285
ip access-list standard 291
ip address 826
ip address dhcp 537, 694, 748, 881
ip address negotiated 882, 884
ip address secondary 441, 539, 697, 750, 885, 926
ip classless 296
ip crypto 297
ip default-gateway 298
ip dhcp 698, 751
ip dhcp release 540
ip dhcp renew 541
ip dhcp-server database local 299
ip dhcp-server excluded-address 300
ip dhcp-server ping packets 301
ip dhcp-server ping timeout 302
ip dhcp-server pool 303, 394
ip domain-lookup 304
ip domain-name 305
ip domain-proxy 306
ip firewall 307
ip firewall alg 312
ip firewall attack-log threshold 314
ip firewall check reflexive-traffic 313
ip firewall check syn-flood 315
ip firewall check winnuke 316
ip firewall policy-log threshold 317
ip forward-protocol udp 318
ip ftp access-class 320

ip ftp agent 321
ip ftp source-interface 322
ip helper-address 442, 542, 699, 752, 827, 886, 927
ip host 323
ip http 324
ip igmp 443, 543, 700, 753, 829, 887, 928
ip igmp join 325
ip load-sharing per-destination 326
ip mcast-stub downstream 444, 544, 701, 754, 830, 888, 929
ip mcast-stub helper-address 327, 545
ip mcast-stub helper-enable 445, 702, 755, 831, 889, 930
ip mcast-stub upstream 446, 546, 703, 756, 832, 890, 931
ip multicast routing 328
ip name-server 329
ip ospf 447, 547, 704, 757, 833, 891, 932
ip ospf authentication 448, 548, 705, 758, 834, 892, 933
ip ospf network 449, 549, 706, 759, 835, 893, 934
ip policy-class 330
ip policy-timeout 333
ip prefix-list description 335
ip prefix-list seq 336
ip proxy-arp 450, 550, 707, 760, 836, 894, 935
ip radius source-interface 337
ip rip receive version 451, 551, 708, 761, 837, 895, 936, 969
ip rip send version 452, 552, 709, 762, 838, 896, 937, 969
ip route 338
ip route-cache 453, 553, 710, 763, 839, 897, 938
ip routing 339
ip snmp agent 341
ip snmp source-interface 342
ip subnet-zero 343
ip tftp source-interface 344
ip unnumbered 454, 554, 711, 764, 898, 939
ip-address 512
ip-range 481
isdn ldn1 638
isdn ldn2 639
isdn spid1 640
isdn spid2 641
isdn switch-type 642

K

keepalive 455, 840, 899

L

lbo 584

lease 403

lifetime 478

line 345

Line (Console) Interface Configuration command set 943

Line (Telnet) Interface Configuration command set 954

line console 943
line telnet 954
line-length 597
line-timeout 946, 956
lldp 346
lldp receive 456, 555, 712, 841, 900
lldp send 457, 556, 713, 842, 901
lldp send-and-receive 458, 557, 714, 843
local-as 791
local-id 469
logging console 347
logging email 348
logging email address-list 349
logging email on 349
logging email priority-level 350
logging email receiver-ip 351
logging email sender 352
logging email source-interface 353
logging facility 354
logging forwarding on 355
logging forwarding priority-level 355, 356
logging forwarding receiver-ip 357
logging forwarding source-interface 358
login 947, 957
login authentication 948, 958
login local-userlist 949, 959
logout 12, 123
loop-alarm-detect 608
Loopback Interface Configuration command set 916
loopback network 585, 598, 609, 622
loopback remote line 586
loopback remote line inband 599
loopback remote payload 587
loopback remote V54 610

M

mac address-table aging-time 359
mac address-table static 360
mac-address 558
match 988
match address 486, 495
modem countrycode 361
mtu 459, 559, 715, 765, 844, 902, 940

N

nat-traversal 470
netbios-name-server 404, 482
netbios-node-type 405
network 406, 964
network area 977
next-hop-self 792
no enable password 269
ntp-server 407

O

oam retry 767
oam-pvc managed 766
option 408

P

parity 950
passive-interface 965
password 513, 793, 951, 960
peer 471
peer default ip address 903
ping 13, 1004
point-to-point 277
port-auth supplicant enable 560
ppoe ac-name 911
ppp authentication 904
ppp chap hostname 907
ppp chap password 908
ppp chap sent-username/password 910
PPP Interface Configuration command set 851
ppp multilink 909
pppoe service-name 912
prefix-list 794
preventing unauthorized users 5
priority 989
pvc 768

Q

QoS command set 984
qos map 363
qos-policy out 660, 769, 913

R

Radius Group command set 503
radius-server 364
radius-server host 366
redistribute connected 966, 978
redistribute ospf 967
redistribute rip 979
redistribute static 968, 980
reload 124
remote-alarm 588, 611
remote-as 795
remote-loopback 589, 600, 612
respond 472
retrain 777
rip 979
route-map 367, 796
Router (OSPF) Configuration command set 970
Router (RIP) Configuration command set 961
router bgp 368
router ospf 369, 970
router rip 370

S

sa4tx-bit 613
send-community standard 797
Serial Interface Configuration command set 570
serial-mode 576
serial-number 514
server 504
set dscp 990
set peer 488, 497
set pfs 489
set precedence 991
set security-association lifetime 490
set session-key 498
set transform-set 491, 502
shortcuts 6
show access-lists 125
show arp 126
show atm 127
show backup interfaces 129
show bridge 130
show buffers 131
show buffers users 132
show cflash 133
show clock 15, 134
show configuration 135
show connections 137
show crypto ca 138
show crypto ike 139
show crypto ipsec 141
show crypto map 142
show debugging 143
show dialin interfaces 144
show dynamic-dns 145
show event-history 146, 270
show file 147
show flash 148
show frame-relay 149
show frame-relay fragment 151
show frame-relay multilink 152
show hosts 153
show interfaces 154
show interfaces adsl 157
show ip access-lists 158
show ip arp 159
show ip as-path-list 160
show ip bgp community 161
show ip bgp community-list 163
show ip bgp neighbors 167
show ip bgp regexp 164
show ip bgp summary 166
show ip cache 170
show ip community-list 171
show ip dhcp-client lease 172
show ip dhcp-server binding 173
show ip igmp groups 174
show ip igmp interface 175
show ip interfaces 176
show ip mroute 177
show ip ospf 178
show ip ospf database 179
show ip ospf interface 181
show ip ospf neighbor 182
show ip ospf summary-address 183
show ip policy-class 184
show ip policy-sessions 185
show ip policy-stats 186
show ip prefix-list 187
show ip protocols 188
show ip route 189
show ip traffic 191
show lldp 193
show lldp device 194
show lldp interface 195
show lldp neighbors interface 196
show lldp neighbors realtime 197
show lldp neighbors statistics 198
show memory 199
show modules 201
show output-startup 202
show port-auth 203
show pppoe 204
show processes 205
show qos map 206
show queue 209
show queuing 210
show radius statistics 211
show route map 212
show running-config 214, 1006
show snmp 16, 216
show snmp 217
show spanning-tree, status 218
show startup-config 219
show startup-config checksum 221
show tcp info 222
show tech 226
show test-pattern 590, 614
show udp info 226
show users 227
show version 17, 228
show version cflash 229
show version flash 229
shutdown 1008

signaling-mode 601

snmp trap 562, 577, 661, 723, 941

snmp trap link-status 461, 563, 578, 591, 602, 615, 623, 662, 724, 914, 942

snmp-server chassis-id 372

snmp-server community 373

snmp-server contact 374

snmp-server enable traps 375

snmp-server host traps 376

snmp-server host traps version 377

snmp-server location 378

snmp-server management-url 379

snmp-server management-url-label 380

snmp-server source-interface 381

snmp-server view 382

snr-margin 778

sntp server 383

soft-reconfiguration inbound 798

spanning-tree bpdudfilter 564, 716, 770

spanning-tree bpduguard 565, 717, 771

spanning-tree bpduguard filter 385

spanning-tree edgeport 566, 718, 772

spanning-tree forward-time 387

spanning-tree hello-time 388

spanning-tree link-type 567, 719, 773

spanning-tree max-age 389

spanning-tree path-cost 720, 774

spanning-tree pathcost 391

spanning-tree port-priority 568, 721

spanning-tree priority 392, 775

speed 569, 952

static 968

stopbits 953

subject-name 515

summary-address not-advertise 981

T

T1 Interface Configuration command set 579

tdm-group 592, 616

telnet 18, 230

terminal length 231

test-pattern 593, 603, 617, 624

tftp-server 409

timers lsa-group-pacing 982

timers spf 983

timezone-offset 410

traceroute 19, 232

training-mode 779

ts16 618, 625

tunnel checksum 845

tunnel destination 846

tunnel key 847

tunnel mode gre 848

tunnel sequence-datagrams 849

tunnel source 850

U

unauthorized users 5

undebg all 233

update-source 799

username password 393

V

version 969

VT100 configuration 4

W

wall 234

warranty 2

write 235