

Access Control Lists (ACLs)

Contents

Introduction	10-4
Overview of Options for Applying ACLs on the Switch	10-5
Static ACLS	10-5
Dynamic Port ACLs	10-5
Terminology	10-10
Overview	10-15
Types of IP ACLs	10-15
ACL Applications	10-15
RACL Applications	10-16
VACL Applications	10-18
Static Port ACL and Dynamic Port ACL Applications	10-19
Multiple ACLs on an Interface	10-20
Features Common to All ACL Applications	10-22
General Steps for Planning and Configuring ACLs	10-24
ACL Operation	10-26
Introduction	10-26
The Packet-filtering Process	10-27
Planning an ACL Application	10-30
IP Traffic Management and Improved Network Performance	10-30
Security	10-32
Guidelines for Planning the Structure of an ACL	10-32
ACL Configuration and Operating Rules	10-33
How an ACE Uses a Mask To Screen Packets for Matches	10-36
What Is the Difference Between Network (or Subnet) Masks and the Masks Used with ACLs?	10-36
Rules for Defining a Match Between a Packet and an Access Control Entry (ACE)	10-37

Configuring and Assigning an ACL	10-41
Overview	10-41
General Steps for Implementing ACLs	10-41
Options for Permit/Deny Policies	10-42
ACL Configuration Structure	10-42
Standard ACL Structure	10-43
Extended ACL Configuration Structure	10-45
ACL Configuration Factors	10-46
The Sequence of Entries in an ACL Is Significant	10-46
Allowing for the Implied Deny Function	10-48
A Configured ACL Has No Effect Until You Apply It to an Interface	10-48
You Can Assign an ACL Name or Number to an Interface Even if the ACL Does Not Exist in the Switch's Configuration	10-48
Using the CLI To Create an ACL	10-49
General ACE Rules	10-49
Using CIDR Notation To Enter the ACL Mask	10-50
Configuring Standard ACLs	10-51
Configuring Named, Standard ACLs	10-53
Creating Numbered, Standard ACLs	10-56
Configuring Extended ACLs	10-60
Configuring Named, Extended ACLs	10-62
Configuring Numbered, Extended ACLs	10-74
Adding or Removing an ACL Assignment On an Interface	10-81
Filtering Routed IP Traffic	10-81
Filtering IP Traffic Inbound on a VLAN	10-82
Filtering Inbound IP Traffic Per Port	10-84
Classifier-Based Rate-Limiting with RL-PACLs	10-85
CLI Command for Rate Limiting	10-86
Viewing the RL-PACL Information	10-86
Show Access Lists by Port	10-87
Troubleshooting RL-PACLs	10-88
Deleting an ACL	10-89
Editing an Existing ACL	10-90
Using the CLI To Edit ACLs	10-90

General Editing Rules	10-90
Sequence Numbering in ACLs	10-91
Inserting an ACE in an Existing ACL	10-92
Deleting an ACE from an Existing ACL	10-94
Resequencing the ACEs in an ACL	10-95
Attaching a Remark to an ACE	10-96
Operating Notes for Remarks	10-99
Displaying ACL Configuration Data	10-100
Display an ACL Summary	10-101
Display the Content of All ACLs on the Switch	10-102
Display the RACL and VACL Assignments for a VLAN	10-103
Display Static Port ACL Assignments	10-104
Displaying the Content of a Specific ACL	10-105
Display All ACLs and Their Assignments in the Routing Switch Startup-Config File and Running-Config File	10-107
Creating or Editing ACLs Offline	10-108
Creating or Editing an ACL Offline	10-108
The Offline Process	10-108
Example of Using the Offline Process	10-109
Enable ACL “Deny” Logging	10-113
Requirements for Using ACL Logging	10-113
ACL Logging Operation	10-114
Enabling ACL Logging on the Switch	10-115
General ACL Operating Notes	10-117

Introduction

An Access Control List (ACL) is a list of one or more Access Control Entries (ACEs) specifying the criteria the switch uses to either permit (forward) or deny (drop) IP packets traversing the switch's interfaces. This chapter describes how to configure, apply, and edit ACLs in a network populated with the switches covered by this guide, and how to monitor ACL actions.

Feature	Default	CLI
Standard ACLs	None	10-51
Extended ACLs	None	10-60
Enable or Disable an ACL	n/a	10-81
Display ACL Data	n/a	10-100
Delete an ACL	n/a	10-89
Configure an ACL from a TFTP Server	n/a	10-108
Enable ACL Logging	n/a	10-115

IP filtering with ACLs can help improve network performance and restrict network use by creating policies for:

- **Switch Management Access:** Permits or denies in-band management access. This includes limiting and/or preventing the use of designated protocols that ride on top of IP, such as TCP, UDP, IGMP, ICMP, and others. Also included are the use of precedence and ToS criteria, and control for application transactions based on source and destination IP addresses and transport layer port numbers.
- **Application Access Security:** Eliminates unwanted IP traffic in a path by filtering IP packets where they enter or leave the switch on specific VLAN interfaces.

ACLs can filter IP traffic to or from a host, a group of hosts, or entire subnets.

Notes

ACLs can enhance network security by blocking selected IP traffic, and can serve as part of your network security program. *However, because ACLs do not provide user or device authentication, or protection from malicious manipulation of data carried in IP packet transmissions, they should not be relied upon for a complete security solution.*

ACLs on the switches covered by this manual do not screen non-IP traffic such as AppleTalk and IPX.

Overview of Options for Applying ACLs on the Switch

To apply ACL filtering, assign a configured ACL to the interface on which you want the IP traffic filtering to occur. VLAN and routed IP traffic ACLs can be applied statically using the switch configuration. Port traffic ACLs can be applied either statically or dynamically (using a RADIUS server).

Static ACLS

Static ACLs are configured on the switch. To apply a static ACL, you must assign it to an interface (VLAN or port). The switch supports three static ACL applications:

Routed IP Traffic ACL (RACL). An RACL is an ACL configured on a VLAN to filter routed IP traffic entering or leaving the switch on that interface, as well as IP traffic having a destination on the switch itself. (Except for filtering IP traffic to an IP address on the switch itself, RACLs can operate only while IP routing is enabled. Refer to “Notes on IP Routing” on page 10-25.)

VLAN ACL (VACL). A VACL is an ACL configured on a VLAN to filter IP traffic entering the switch on that VLAN interface and having a destination on the same VLAN.

Static Port ACL. A static port ACL is an ACL configured on a port to filter IP traffic entering the switch on that port, regardless of whether the IP traffic is routed, switched, or addressed to a destination on the switch itself.

Dynamic Port ACLs

A dynamic port ACL is configured on a RADIUS server for assignment to a given port when the server authenticates a specific client on that port. When the client is authenticated, the ACL configured for that client on the server is assigned to the port and applied to the IP traffic received inbound on that port from the authenticated client. When the client session ends, the ACL is removed from the port. The switch allows as many dynamic port ACLs on a port as it allows authenticated clients.

Access Control Lists (ACLs)

Overview of Options for Applying ACLs on the Switch

Note

This chapter describes the ACL applications you can statically configure on the switch. For information on dynamic port ACLs assigned by a RADIUS server, refer to the chapter 7, “Configuring RADIUS Server Support for Switch Services”.

Table 10-1. Command Summary for Standard ACLs

Action	Command(s)	Page
Create a Standard, Named ACL <i>or</i> Add an ACE to the End of an Existing Standard, Named ACL	ProCurve(config)# ip access-list standard < name-str > ProCurve(config-std-nacl)# < deny permit > < any host <SA > SA/< mask-length > SA < mask >> ¹ [log] ²	10-53
Create a Standard, Numbered ACL <i>or</i> Add an ACE to the End of an Existing Standard, Numbered ACL	ProCurve(config)# access-list < 1-99 > < deny permit > < any host <SA > SA/< mask-length > SA < mask >> [log] ²	10-56
Use a Sequence Number To Insert an ACE in a Standard ACL	ProCurve(config)# ip access-list standard < name-str 1-99 > ProCurve(config-std-nacl)# 1-2147483647 < deny permit > < any host <SA > SA/< mask-length > SA < mask >> ¹ [log] ²	10-91
Use an ACE's Sequence Number To Delete the ACE from a Standard ACL	ProCurve(config)# ip access-list standard < name-str 1-99 > ProCurve(config-std-nacl)# no < 1-2147483647 >	10-94
Resequence the ACEs in a Standard ACL	ProCurve(config)# ip access-list resequence < name-str 1-99 > < 1-2147483647 > < 1-2147483646 >	10-95

Enter or Remove a Remark from a Standard ACL	ProCurve(config)# ip access-list standard < name-str 1-99 > ProCurve(config-ext-nacl)# [remark < remark-str > no < 1-2147483647 > remark]	10-96 10-98
<p style="text-align: center;"><i>For numbered, standard ACLs only, the following remark commands can be substituted for the above:</i></p> <p style="text-align: center;">ProCurve(config)# access-list < 1 - 99 > remark < remark-str > ProCurve(config)# [no] access-list < 1 - 99 > remark</p>		
Delete a Standard ACL	ProCurve(config)# no ip access-list standard < name-str 1-99 >	10-89
<p style="text-align: center;"><i>For numbered, standard ACLs, the following command can be substituted for the above:</i></p> <p style="text-align: center;">ProCurve(config)# access-list < 1 - 99 > remark < remark-str ></p>		

¹The mask can be in either dotted-decimal notation (such as 0.0.15.255) or CIDR notation (such as /20).

²The [log] function applies only to “deny” ACLs, and generates a message only when there is a “deny” match.

Access Control Lists (ACLs)

Overview of Options for Applying ACLs on the Switch

Table 10-2. Command Summary for Extended ACLs

Action	Command(s)	Page
Create an Extended, Named ACL <i>or</i> Add an ACE to the End of an Existing, Extended ACL	ProCurve(config)# ip access-list extended < name-str 100-199 > ProCurve(config-std-nacl)# < deny permit > < ip ip-protocol ip-protocol-nbr > < any host <SA > SSA/< mask-length > SA < mask >> ¹ < any host < DA > DA/< mask-length > DA < mask >> ¹ < tcp udp > < any host <SA > SA/< mask-length > SA < mask >> ¹ [comparison-operator < value >] < any host <DA > DA/< mask-length > DA < mask >> ¹ [comparison-operator < value >] [established] < igmp > < any host <SA > SA/< mask-length > SA < mask >> ¹ < any host < DA > DA/< mask-length > DA < mask >> ¹ [igmp-packet-type] < icmp > < any host <SA > SA/< mask-length > SA < mask >> ¹ < any host < DA > DA/< mask-length > DA < mask >> ¹ [[< 0 - 255 > [0 - 255]] icmp-message] [precedence < priority >] [tos < tos- setting >] [log] ²	10-62
Create an Extended, Numbered ACL <i>or</i> Add an ACE to the End of an Existing, Numbered ACL	ProCurve(config)# access-list < 100-199 > < deny permit > < ip-options tcp/udp-options igmp-options icmp-options > [precedence < priority >] [tos < tos- setting >] [log] ² Note: Uses the same IP, TCP/UDP, IGMP, and ICMP options as shown above for "Create an Extended, Named ACL".	10-74
Insert an ACE by Assigning a Sequence Number	ProCurve(config)# ip access-list extended < name-str 100-199 > ProCurve(config-ext-nacl)# 1-2147483647 < deny permit > Uses the options shown above for "Create an Extended, Named ACL".	10-92
Delete an ACE by Specifying Its Sequence Number	ProCurve(config)# ip access-list extended < name-str 100-199 > ProCurve(config-std-nacl)# no < 1-2147483647 >	10-94
Resequence the ACEs in an ACL	ProCurve(config)# ip access-list resequence < name-str 100-199 > < 1-2147483647 > < 1-2147483646 >	10-95

¹The mask can be in either dotted-decimal notation (such as 0.0.15.255) or CIDR notation (such as /20).

²The [log] function applies only to "deny" ACLs, and generates a message only when there is a "deny" match.

Action	Command(s)	Page
Enter or Remove a Remark	ProCurve(config)# ip access-list extended < name-str 100-199 >	10-96
	ProCurve(config-ext-nacl)# [remark < remark-str > no remark]	10-98
	<i>For numbered, extended ACLs only, the following remark commands can be substituted for the above:</i>	
	ProCurve(config)# access-list < 100 - 199 > remark < remark-str >	
	ProCurve(config)# [no] access-list < 100 - 199 > remark	
Delete an Extended ACL	ProCurve(config)# no ip access-list extended < name-str 100-199 >	10-89
	<i>For numbered, extended ACLs only, the following command can also be used:</i>	
	ProCurve(config)# no access-list < 100 - 199 >	

Table 10-3. Command Summary for Enabling, Disabling, and Displaying ACLs

Enable or Disable an RACL	ProCurve(config)# [no] vlan < vid > ip access-group < identifier > < in out >	10-81
Enable or Disable a VACL	ProCurve(config)# [no] vlan < vid > ip access-group < identifier > < vlan >	
Enable or Disable a Static Port ACL	ProCurve(config)# [no] interface < port-list Trkx > access-group < identifier > in ProCurve(eth-< port-list > Trkx >)# [no] ip access-group < identifier > in	
Displaying ACL Data	ProCurve(config)# show access-list ProCurve(config)# show access-list < acl-identifier > ProCurve(config)# show access-list config ProCurve(config)# show access-list vlan < vid > ProCurve(config)# show access-list radius	10-100

Terminology

Access Control Entry (ACE): A policy consisting of criteria and an action (permit or deny) to execute on a packet if it meets the criteria. The elements composing the criteria include:

- source IP address and mask (standard and extended ACLs)
- destination IP address and mask (extended ACLs only)
- either of the following:
 - all IP traffic
 - IP traffic of a specific IP protocol (extended ACLs only)
(In the cases of TCP, UDP, ICMP, and IGMP, the criteria can include either all IP traffic of the protocol type or only the IP traffic of a specific sub-type within the protocol.)
- option to log packet matches with **deny** ACEs
- optional use of IP precedence and ToS settings (extended ACLs only)

Access Control List (ACL): A list (or set) consisting of one or more explicitly configured Access Control Entries (ACEs) and terminating with an implicit “deny” ACE. ACLs can be used to filter IP traffic and to select IP traffic to be monitored (mirrored). ACL types include “standard” and “extended”. See “Standard ACL” and “Extended ACL”. For filtering IP traffic, both can be applied in any of the following ways:

- **RACL:** an ACL assigned to filter routed IP traffic entering or leaving the switch on a VLAN. (Separate assignments are required for inbound and outbound IP traffic.)
- **VACL:** an ACL assigned to filter inbound IP traffic on a specific VLAN configured on the switch
- **Static Port ACL:** an ACL assigned to filter inbound IP traffic on a specific switch port
- **Dynamic Port ACL:** dynamic ACL assigned to a port by a RADIUS server to filter inbound IP traffic from an authenticated client on that port

An ACL can be configured on a VLAN as an RACL or VACL (or both), and on a port (or static trunk) as a static port ACL. (Dynamic port ACLs are configured on a RADIUS server.)

See also “ACL Mirroring”.

ACE: See “Access Control Entry”.

ACL: See “Access Control List”.

ACL ID: A number or alphanumeric string used to identify an ACL. A *standard* ACL ID can have either an alphanumeric string or a number in the range of 1 to 99. An *extended* ACL ID can have either an alphanumeric string or a number in the range of 100 to 199. See also “Identifier”.

Note: RADIUS-assigned ACLs are identified by client authentication data and do not use the ACL ID strings described here.

ACL Mask: Follows any IP address (source or destination) listed in an ACE. Defines which bits in a packet’s corresponding IP addressing must exactly match the IP addressing in the ACE, and which bits need not match (wildcards). See also “How an ACE Uses a Mask To Screen Packets for Matches” on page 10-36.)

CIDR: This is the acronym for Classless Inter-Domain Routing.

Connection-Rate ACL: An optional feature used with Connection-Rate filtering based on virus-throttling technology. For more information, refer to the chapter 3, “Virus Throttling”.

DA: The acronym used in text to represent *Destination IP Address*. In an IP packet, this is the destination IP address carried in the header, and identifies the destination intended by the packet’s originator. In an extended ACE, this is the second of two IP addresses required by the ACE to determine whether there is a match between a packet and the ACE. See also “SA”.

Deny: An ACE configured with this action causes the switch to drop an IP packet for which there is a match within an applicable ACL.

Dynamic Port ACL: An ACL assigned by a RADIUS server to a port to filter inbound IP traffic from a client authenticated by the server for that port. A dynamic port ACL filters all inbound IP traffic, regardless of whether it is switched or routed. When the client session ends, the dynamic port ACL for that client is removed from the port.

Extended ACL: This type of Access Control List uses layer-3 IP criteria composed of source and destination IP addresses and (optionally) TCP/UDP port, ICMP, IGMP, precedence, or ToS criteria to determine whether there is a match with an IP packet. Except for RADIUS-assigned ACLs, which use client credentials for identifiers, extended ACLs require an alphanumeric name or an identification number (ID) in the range of 100 - 199.

identifier: The term used in ACL syntax statements to represent either the name or number by which the ACL can be accessed. See also **name-str**.
Note: RADIUS-assigned ACLs are identified by client authentication data and do not use the identifiers described in this chapter.

Implicit Deny: If the switch finds no matches between an IP packet and the configured criteria in an applicable ACL, then the switch denies (drops) the packet with an implicit **deny any** function (for standard ACLs) or an implicit **deny ip any any** function (for extended ACLs). You can preempt the Implicit Deny in a given ACL by configuring a **permit any** (standard) or **permit IP any any** (extended) as the last explicit ACE in the ACL. Doing so permits any IP packet that is not explicitly permitted or denied by other ACEs configured sequentially earlier in the ACL. Unless otherwise noted, Implicit Deny refers to the “deny” function enforced by both standard and extended ACLs.

Inbound Traffic: For the purpose of defining where the switch applies ACLs to filter IP traffic, inbound traffic is any IP packet that meets one of the following criteria:

- Routed ACL (RACL): Inbound traffic is any IP packet entering the switch on a VLAN interface (or a subnet in a multinetted VLAN) with a destination IP address (DA) that is for any of the following:
 - an external device on a different VLAN or subnet than the interface on which it arrived
 - an IP address configured on the switch itself
 - a broadcast

Note that, except for IP traffic addressed to the switch itself, and outbound IP traffic generated by the switch, routing must be configured on the switch to enable support for RACL applications.

- VLAN ACL (VACL): Inbound traffic is any IP packet entering the switch on a VLAN interface (or a subnet in a multinetted VLAN).
- Static Port ACL: Inbound traffic is any IP packet entering the switch on the port.
- Dynamic Port ACL: Where a RADIUS server has authenticated a client and assigned an ACL to the port to filter the client’s IP traffic, inbound traffic is any IP packet entering the switch from that client.

name-str: The term used in extended ACL syntax statements to represent the “name string”; the alphanumeric string used to identify the ACL. See also **identifier** and **ACL-ID**.

Named ACL: An ACL created with the **ip access-list < extended | standard > < name-str >** command and then populated using the **< deny | permit >** command in the Named ACL (**nacl**) CLI context. (Refer to “Entering the “Named ACL” (nacl) Context” on page 10-53.)

Numbered ACL: An ACL created and initially populated by using the **access-list < 1-99 | 100 - 199 >** command. (Refer to “Creating or Adding to a Standard, Numbered ACL” on page 10-57.) After a numbered ACL has been created, the switch manages it in the same way as a named ACL, meaning that it can be applied and edited in the same way as a named ACL.

Outbound Traffic: For defining the points where the switch applies an RACL to filter IP traffic, outbound traffic is routed IP traffic *leaving the switch* through a VLAN interface (or a subnet in a multinetted VLAN). “Outbound traffic” can also apply to switched IP traffic leaving the switch on a VLAN interface, but VACLs do not filter outbound switched IP traffic. (Refer also to “ACL Applications” on page 10-15.)

Permit: An ACE configured with this action allows the switch to forward a routed IP packet for which there is a match within an applicable ACL.

Permit Any Forwarding: An ACE configured with this action causes the switch to forward all routed IP packets that have not been permitted or denied by earlier ACEs in the list. In a standard ACL, this is **permit any**. In an extended ACL, it is **permit ip any any**.

RACL: See “Routed ACL”.

RADIUS-Assigned ACL: See “Dynamic Port ACL”.

remark-str: The term used in ACL syntax statements to represent the variable “remark string”; a set of alphanumeric characters you can include in a remark in an ACL. A remark string can include up to 100 characters and must be delimited by single or double quotes if any spaces are included in the string.

Rate-Limit Port ACLs (RL-PACLs): allows you to create an ACL and apply it on a per-port basis to rate-limit network traffic.

Routed ACL (RACL): An ACL applied to routed IP traffic that is entering or leaving the switch on a given VLAN. See also “Access Control List”.

SA: The acronym for *Source IP Address*. In an IP packet, this is the source IP address carried in the IP header, and identifies the packet’s sender. In a standard ACE, this is the IP address used by the ACE to determine whether

there is a match between a packet and the ACE. In an extended ACE, this is the first of two IP addresses used by the ACE to determine whether there is a match between a packet and the ACE. See also “DA”.

seq-#: The term used in ACL syntax statements to represent the sequence number variable used to insert an ACE within an existing list. The range allowed for sequence numbers is 1 - 2147483647.

Standard ACL: This type of access control list uses the layer-3 IP criteria of source IP address to determine whether there is a match with an IP packet. Except for RADIUS-assigned ACLs, standard ACLs require an alphanumeric name or an identification number (ID) in the range of 1-99. See also **identifier** on page 10-12.

Static Port ACL: An ACL statically configured on a specific port, group of ports, or trunk. A static port ACL filters all incoming IP traffic on the port, regardless of whether it is switched or routed.

VACL: See “VLAN ACL”.

VLAN ACL (VACL): An ACL applied to all IP traffic entering the switch on a given VLAN interface. See also “Access Control List”.

Wildcard: The part of a mask that indicates the bits in a packet’s IP addressing that do not need to match the corresponding bits specified in an ACL. See also **ACL Mask** on page 10-11.

Overview

Types of IP ACLs

A permit or deny policy for IP traffic you want to filter can be based on source IP address alone, or on source IP address plus other IP factors.

Standard ACL: Use a standard ACL when you need to permit or deny IP traffic based on source IP address only. Standard ACLs are also useful when you need to quickly control a performance problem by limiting IP traffic from a subnet, group of devices, or a single device. (This can block all IP traffic from the configured source, but does not hamper IP traffic from other sources within the network.) A standard ACL uses an alphanumeric ID string or a numeric ID of 1 through 99. You can specify a single host, a finite group of hosts, or any host.

Extended ACL: Use an extended ACL when simple IP source address restrictions do not provide the sufficient IP traffic selection criteria needed on an interface. Extended ACLs allow use of the following criteria:

- source and destination IP address combinations
- IP protocol options

Extended, named ACLs also offer an option to permit or deny IP connections using TCP for applications such as Telnet, http, ftp, and others.

Connection-Rate ACL. An optional feature used with Connection-Rate filtering based on virus-throttling technology. Refer to the chapter 3, “Virus Throttling”.

ACL Applications

ACL filtering is applied to IP traffic as follows:

- Routed ACL (RACL)— on a VLAN configured with an RACL:
 - routed IP traffic entering or leaving the switch. (Routing can be between different VLANs or between different subnets in the same VLAN. IP routing *must* be enabled.)
 - routed IP traffic having a destination address (DA) on the switch itself. In figure 10-1 on page 10-17, this is any of the IP addresses shown in VLANs “A”, “B”, and “C”. (IP routing need not be enabled.)

- outbound traffic generated by the switch itself.
- VLAN ACL (VACL): on a VLAN configured with a VACL, any inbound IP traffic, regardless of whether it is switched or routed. On a multi-netted VLAN, this includes all inbound IP traffic from any subnet.
- Static port ACL: any inbound IP traffic on that port.
- Dynamic port ACL: on a port having an ACL assigned by a RADIUS server to filter an authenticated client's IP traffic, any inbound IP traffic from that client

(For information on RADIUS-assigned ACLs, refer to chapter 7, “Configuring RADIUS Server Support for Switch Services”.)

- ACL Mirroring: applies an ACL to a port or VLAN to mirror selected IP traffic to a mirror destination. In this context, a **permit** ACE means to mirror the specified IP traffic; a **deny** ACE means to avoid mirroring. (A **log** keyword in a **deny** ACE is ignored when the associated ACL is used for mirroring.) Refer to “Local and Remote Traffic Mirroring” in the appendix titled “Monitoring and Analyzing Switch Operation” in the *Management and Configuration Guide* for your switch.
- Connection-Rate ACL: An optional feature used with Connection-Rate filtering based on virus-throttling technology. Refer to the chapter 3, “Virus Throttling”.

RACL Applications

RACLs filter routed IP traffic entering or leaving the switch on VLANs configured with the “in” and/or “out” ACL option

```
vlan < vid > ip access-group < identifier > < in | out >
```

For example, in figure 10-1:

- You would assign either an inbound ACL on VLAN 1 or an outbound ACL on VLAN 2 to filter a packet routed between subnets on different VLANs; that is, from the workstation 10.28.10.5 on VLAN 1 to the server at 10.28.20.99 on VLAN 2. (An outbound ACL on VLAN 1 or an inbound ACL on VLAN 2 would not filter the packet.)
- Where multiple subnets are configured on the same VLAN, then you can use either inbound or outbound ACLs to filter routed IP traffic between the subnets on the VLAN if the traffic source and destination IP addresses are on devices external to the switch.

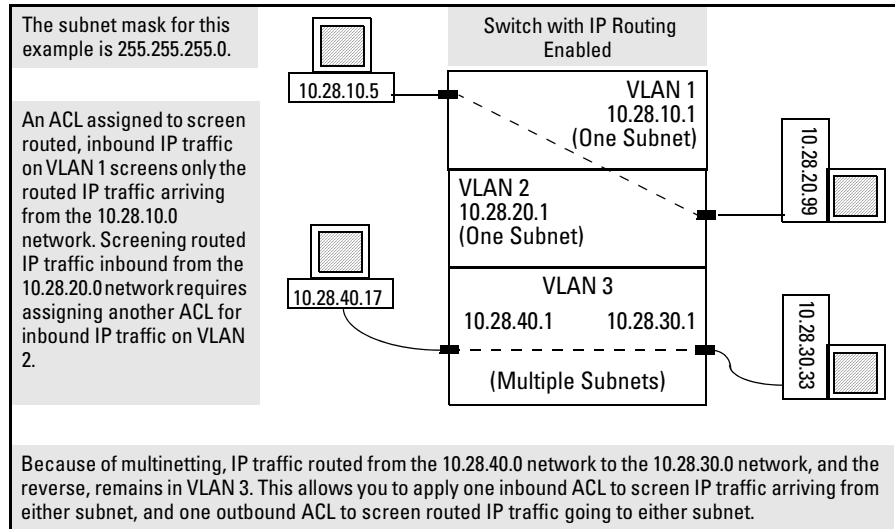


Figure 10-1. Example of RACL Filter Applications on Routed IP Traffic

Notes

The switch allows one inbound RACL assignment and one outbound RACL assignment configured per VLAN. This is in addition to any other ACL assigned to the VLAN or to any ports on the VLAN. You can use the same RACL or different RACLs to filter inbound and outbound routed IP traffic on a VLAN.

RACLs do not filter IP traffic that remains in the same subnet from source to destination (switched IP traffic) unless the destination IP address (DA) or source IP address (SA) is on the switch itself.

VACL Applications

VACLs filter any IP traffic entering the switch on a VLAN configured with the “VLAN” ACL option.

```
vlan < vid > ip access-group < identifier > vlan
```

For example, in figure 10-2, you would assign a VACL to VLAN 2 to filter all inbound switched or routed IP traffic received from clients on the 10.28.20.0 network. In this instance, routed IP traffic received on VLAN 2 from VLANs 1 or 3 would not be filtered by the VACL on VLAN 2.

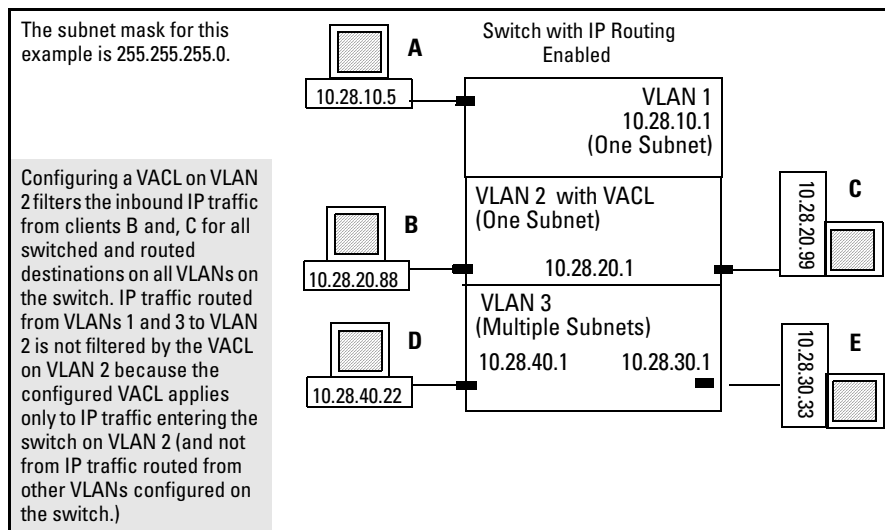


Figure 10-2. Example of VACL Filter Applications on IP Traffic Entering the Switch

Note

The switch allows one VACL assignment configured per VLAN. This is in addition to any other ACL applications assigned to the VLAN or to ports in the VLAN.

Static Port ACL and Dynamic Port ACL Applications

- **Static Port ACL:** filters any IP traffic inbound on the designated port, regardless of whether it is switched or routed.
- **Dynamic (RADIUS-assigned) Port ACL:** filters IP traffic inbound from the client whose authentication resulted in the ACL assignment to the designated port. For example, client “A” connects to a given port and is authenticated by a RADIUS server. Because the server is configured to assign an ACL to the port used by the authenticated client, all IP traffic inbound on the port from client “A” is filtered.

Effect of Dynamic Port ACLs When Multiple Clients Are Using the Same Port. Some network configurations may allow multiple clients to authenticate through a single port where a RADIUS server assigns a separate, dynamic port ACL in response to each client’s authentication on that port. In such cases, a given client’s inbound traffic will be allowed only if the RADIUS authentication response for that client includes a dynamic port ACL. For example, in figure 10-3 (below), clients A through D authenticate through the same port (B1) on the 8212zl switch.

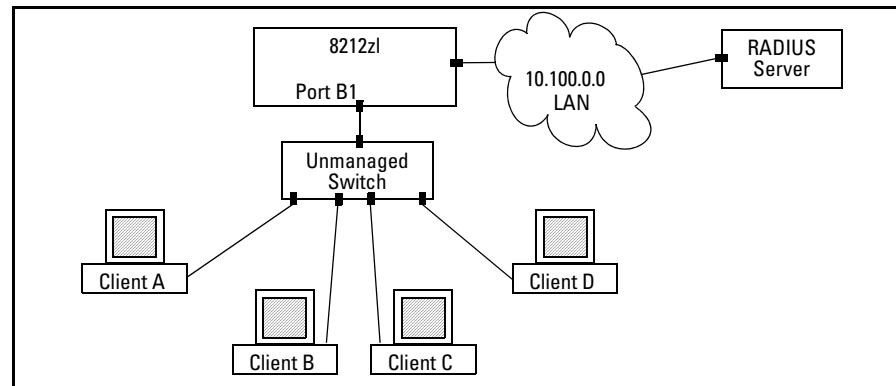


Figure 10-3. Example of Multiple Clients Authenticating Through a Single Port

In this case, the RADIUS server must be configured to assign a dynamic port ACL to port B1 each time any of the clients authenticates on the port.

802.1X User-Based and Port-Based Applications. User-Based 802.1X access control allows up to 32 individually authenticated clients on a given port. However, port-based access control does not set a client limit, and requires only one authenticated client to open a given port (and is recommended for applications where only one client at a time can connect to the port).

- If you configure 802.1X *user-based* security on a port and the RADIUS response includes a dynamic port ACL for at least one authenticated client, then the RADIUS response for all other clients authenticated on the port must also include a dynamic port ACL. Traffic on the port from any client that authenticates without the RADIUS server including a dynamic port ACL in its response will be dropped and the client will be de-authenticated.
- Using 802.1X *port-based* security on a port where the RADIUS response includes a dynamic port ACL, only the first client to authenticate can use the port. Traffic from other clients will be dropped.

Multiple ACLs on an Interface

Multiple ACL Assignments Allowed. The switch allows multiple ACL applications on an interface (subject to internal resource availability). This means that a port belonging to a given VLAN “X” can simultaneously be subject to all of the following:

- One VACL for any IP traffic for VLAN “X” entering the switch through the port.
- One static port ACL for any IP traffic entering the switch on the port.
- One dynamic (RADIUS-assigned) port ACL applied to inbound IP traffic for each authenticated client on the port
- One connection-rate ACL for inbound IP traffic for VLAN “X” on the port (if the port is configured for connection-rate filtering). (Refer to chapter 3, “Virus Throttling”.)
- ACL mirroring per VLAN, port, and trunk interface (Refer to “Local and Remote Traffic Mirroring” in the appendix titled “Monitoring and Analyzing Switch Operation” in the *Management and Configuration Guide* for your switch.)

- One inbound and one outbound RACL filtering routed IP traffic moving through the port for VLAN “X”. (Also applies to inbound, switched traffic on VLAN “X” that has a destination on the switch itself.”

Note

In cases where an RACL and any type of port or VLAN ACL are filtering traffic entering the switch, the *switched* traffic explicitly permitted by the port or VLAN ACL is not filtered by the RACL (except when the traffic has a destination on the switch itself). However, *routed* traffic explicitly permitted by the port or VLAN ACL (and any switched traffic having a destination on the switch itself) must also be explicitly permitted by the RACL, or it will be dropped.

Also, a switched packet is not affected by an outbound RACL assigned to the VLAN on which the packet exits from the switch.

A Packet Must Have a Match with a “Permit” ACE in All Applicable ACLs Assigned to an Interface. On a given interface where multiple ACLs apply to the same traffic, a packet having a match with a **deny** ACE in any applicable ACL on the interface (including an implicit **deny any**) will be dropped.

For example, suppose the following is true:

- Port A10 belongs to VLAN 100.
- A static port ACL is configured on port A10.
- A VACL is configured on VLAN 100.
- An RACL is also configured for inbound, routed traffic on VLAN 100.

An inbound, *switched* packet entering on port A10, with a destination on port A12, will be screened by the static port ACL and the VACL, regardless of a match with any **permit** or **deny** action. A match with a **deny** action (including an implicit deny) in either ACL will cause the switch to drop the packet. (If the packet has a match with explicit **deny** ACEs in multiple ACLs and the log option is included in these ACEs, then a separate log event will occur for each match.) The switched packet will not be screened by the RACL.

However, suppose that VLAN 2 in figure 10-4 (page 10-22) is configured with the following:

- A VACL permitting IP traffic having a destination on the 10.28.10.0 subnet

- An RACL that denies inbound IP traffic having a destination on the 10.28.10.0 subnet

In this case, no IP traffic received on the switch from clients on the 10.28.20.0 subnet will reach the 10.28.10.0 subnet, even though the VACL allows such traffic. This is because the **deny** in the RACL causes the switch to drop the traffic regardless of whether any other VACLs permit the traffic.

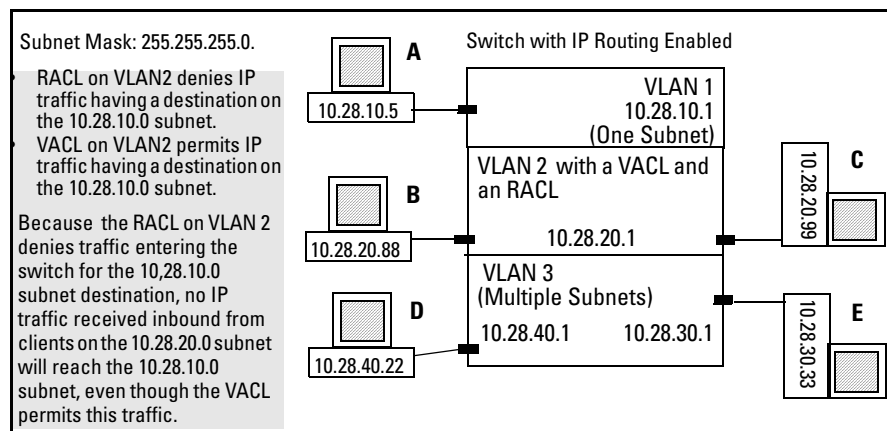


Figure 10-4. Example of Order of Application for Multiple ACLs on an Interface

Exception for Mirrored IP Traffic. If ACL mirroring is configured along with one or more of the above ACL applications on the same interface, the mirroring action occurs regardless of the effect of other ACLs on the packets that match the mirror criteria. This means, for example, that if a dynamic port ACL denies a packet that also meets the mirror ACL criteria for forwarding to the configured mirror destination, the packet will be mirrored even though it will not be forwarded to its intended destination.

Exception for Connection-Rate Filtering. Connection-rate filtering can be configured along with one or more other ACL applications on the same interface. In this case, a connection-rate match for a **filter** action is carried out according to the configured policy, regardless of whether any other ACLs on the interface have a match for a **deny** action. Also, if a connection-rate filter permits (**ignore** action) a packet, it can still be denied by another ACL on the interface.

Features Common to All ACL Applications

- Any ACL can have multiple entries (ACEs).

- You can apply any one ACL to multiple interfaces.
- All ACEs in an ACL configured on the switch are automatically sequenced (numbered). For an existing ACL, entering an ACE without specifying a sequence number automatically places the ACE at the end of the list. Specifying a sequence number inserts the ACE into the list at the correct sequential location.
 - Automatic sequence numbering begins with “10” and increases in increments of 10. You can renumber the ACEs in an ACL and also change the sequence increment between ACEs.
 - The CLI **remark** command option allows you to enter a separate comment for each ACE.
- A source or destination IP address and a mask, together, can define a single host, a range of hosts, or all hosts.
- Every ACL populated with one or more explicit ACEs includes an Implicit Deny as the last entry in the list. The switch applies this action to any packets that do not match other criteria in the ACL. (For standard ACLs, the Implicit Deny is **deny any**. For extended ACLs, it is **deny ip any any**.)
- In any ACL, you can apply an ACL log function to ACEs that have an explicit “deny” action. The logging occurs when there is a match on a “deny” ACE (except when the ACL is used for mirroring). The switch sends ACL logging output to Syslog, if configured, and, optionally, to a console session.

You can create ACLs for the switch configuration using either the CLI or a text editor. The text-editor method is recommended when you plan to create or modify an ACL that has more entries than you can easily enter or edit using the CLI alone. Refer to “Creating or Editing ACLs Offline” on page 10-108.

General Steps for Planning and Configuring ACLs

1. Identify the ACL application to apply. As part of this step, determine the best points at which to apply specific ACL controls. For example, you can improve network performance by filtering unwanted IP traffic at the edge of the network instead of in the core. Also, on the switch itself, you can improve performance by filtering unwanted IP traffic where it is inbound to the switch instead of outbound.

IP Traffic Source	ACL Application
IP traffic from a specific, authenticated client	dynamic port ACL (RADIUS-assigned ACL) for inbound IP traffic from an authenticated client on a port*
IP traffic entering the switch on a specific port	static port ACL (static-port assigned) for any inbound IP traffic on a port from any source
switched or routed IP traffic entering the switch on a specific VLAN	VACL (VLAN ACL)
routed IP traffic entering or leaving the switch on a specific VLAN	RACL (routed ACL)

*For more on this option, refer to chapter 7, "Configuring RADIUS Server Support for Switch Services", and also to the documentation for your RADIUS server.)

2. Identify the IP traffic types to filter:
 - The SA and/or the DA of IP traffic you want to permit or deny. This can be a single host, a group of hosts, a subnet, or all hosts.
 - Any IP traffic of a specific protocol type (0-255)
 - Any TCP traffic (only) for a specific TCP port or range of ports, including optional control of connection traffic based on whether the initial request should be allowed
 - Any UDP traffic (only) or UDP traffic for a specific UDP port
 - Any ICMP traffic (only) or ICMP traffic of a specific type and code
 - Any IGMP traffic (only) or IGMP traffic of a specific type
 - Any of the above with specific precedence and/or ToS settings
3. Design the ACLs for the control points (interfaces) you have selected. Where you are using explicit "deny" ACEs, you can optionally use the ACL logging feature for notification that the switch is denying unwanted packets.
4. Configure the ACLs on the selected switches.

5. Assign the ACLs to the interfaces you want to filter, using the ACL application (static port ACL, VACL, or RACL) appropriate for each assignment. (For RADIUS-assigned ACLs, refer to the Note in the table in step 1 on page 10-24.)
6. If you are using an RACL, ensure that IP routing is enabled on the switch.
7. Test for desired results.

For more details on ACL planning considerations, refer to “Planning an ACL Application” on page 10-30.

Notes on IP Routing

To activate a RACL to screen inbound IP traffic for routing between subnets, assign the RACL to the statically configured VLAN on which the traffic enters the switch. Also, ensure that IP routing is enabled. Similarly, to activate a RACL to screen routed, outbound IP traffic, assign the RACL to the statically configured VLAN on which the traffic exits from the switch. A RACL configured to screen inbound IP traffic with a destination IP address on the switch itself does not require routing to be enabled. (ACLs do not screen outbound IP traffic generated by the switch, itself.) Refer to “ACL Screening of IP Traffic Generated by the Switch” on page 10-117.)

Caution Regarding the Use of Source Routing

Source routing is enabled by default on the switch and can be used to override ACLs. For this reason, if you are using ACLs to enhance network security, the recommended action is to use the **no ip source-route** command to disable source routing on the switch. (If source routing is disabled in the running-config file, the **show running** command includes “**no ip source-route**” in the running-config file listing.)

ACL Operation

Introduction

An ACL is a list of one or more Access Control Entries (ACEs), where each ACE consists of a matching criteria and an action (permit or deny). An ACL applies only to the switch in which it is configured. ACLs operate on assigned interfaces, and offer these traffic filtering options:

- Any IP traffic inbound on a port.
- Any IP traffic inbound on a VLAN.
- Routed IP traffic entering or leaving the switch on a VLAN. (Note that ACLs do not screen traffic at the internal point where traffic moves between VLANs or subnets within the switch. Refer to “ACL Applications” on page 10-15.)

The following table lists the range of interface options:

Interface	ACL Application	Application Point	Filter Action
Port	Static Port ACL (switch configured)	inbound on the switch port	any inbound IP traffic
	Dynamic Port ACL ¹ (RADIUS assigned)	inbound on the switch port used by authenticated client	any inbound IP traffic from the authenticated client
VLAN	VACL	entering the switch on the VLAN	any inbound IP traffic
		entering the switch on the VLAN	routed IP traffic entering the switch and any IP traffic with a destination on the switch itself
	exitting from the switch on the VLAN	routed IP traffic exiting from the switch	

¹This chapter describes ACLs statically configured on the switch. For information on dynamic port ACLs assigned by a RADIUS server, refer to the chapter 7, “Configuring RADIUS Server Support for Switch Services”.

²Supports one inbound and/or one outbound RACL. When both are used, one RACL can be assigned to filter both inbound and outbound, or different RACLs can be assigned to filter inbound and outbound.

Note

After you assign an ACL to an interface, the default action on the interface is to implicitly deny any IP traffic that is not specifically permitted by the ACL. (This applies only in the direction of traffic flow filtered by the ACL.)

The Packet-filtering Process

Sequential Comparison and Action. When an ACL filters a packet, it sequentially compares each ACE's filtering criteria to the corresponding data in the packet until it finds a match. The action indicated by the matching ACE (deny or permit) is then performed on the packet.

Implicit Deny. If a packet does not have a match with the criteria in any of the ACEs in the ACL, the ACL denies (drops) the packet. If you need to override the implicit deny so that a packet that does not have a match will be permitted, then you can use the “permit any” option as the last ACE in the ACL. This directs the ACL to permit (forward) packets that do not have a match with any earlier ACE listed in the ACL, and prevents these packets from being filtered by the implicit “deny any”.

Example. Suppose the ACL in figure 10-5 is assigned to filter the IP traffic from an authenticated client on a given port in the switch:

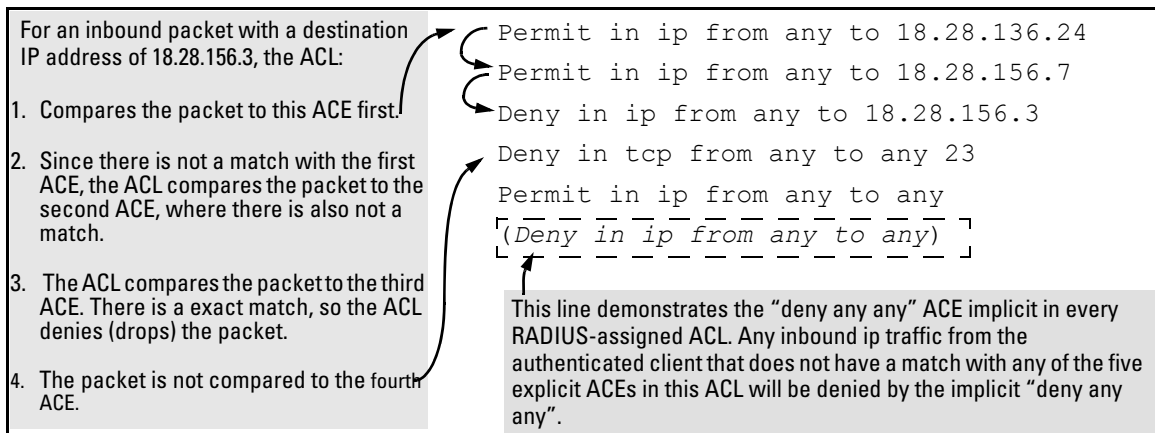


Figure 10-5. Example of Sequential Comparison

As shown above, the ACL tries to apply the first ACE in the list. If there is not a match, it tries the second ACE, and so on. When a match is found, the ACL invokes the configured action for that entry (permit or drop the packet) and

no further comparisons of the packet are made with the remaining ACEs in the list. This means that when an ACE whose criteria matches a packet is found, the action configured for that ACE is invoked, and any remaining ACEs in the ACL are ignored. *Because of this sequential processing, successfully implementing an ACL depends in part on configuring ACEs in the correct order for the overall policy you want the ACL to enforce.*

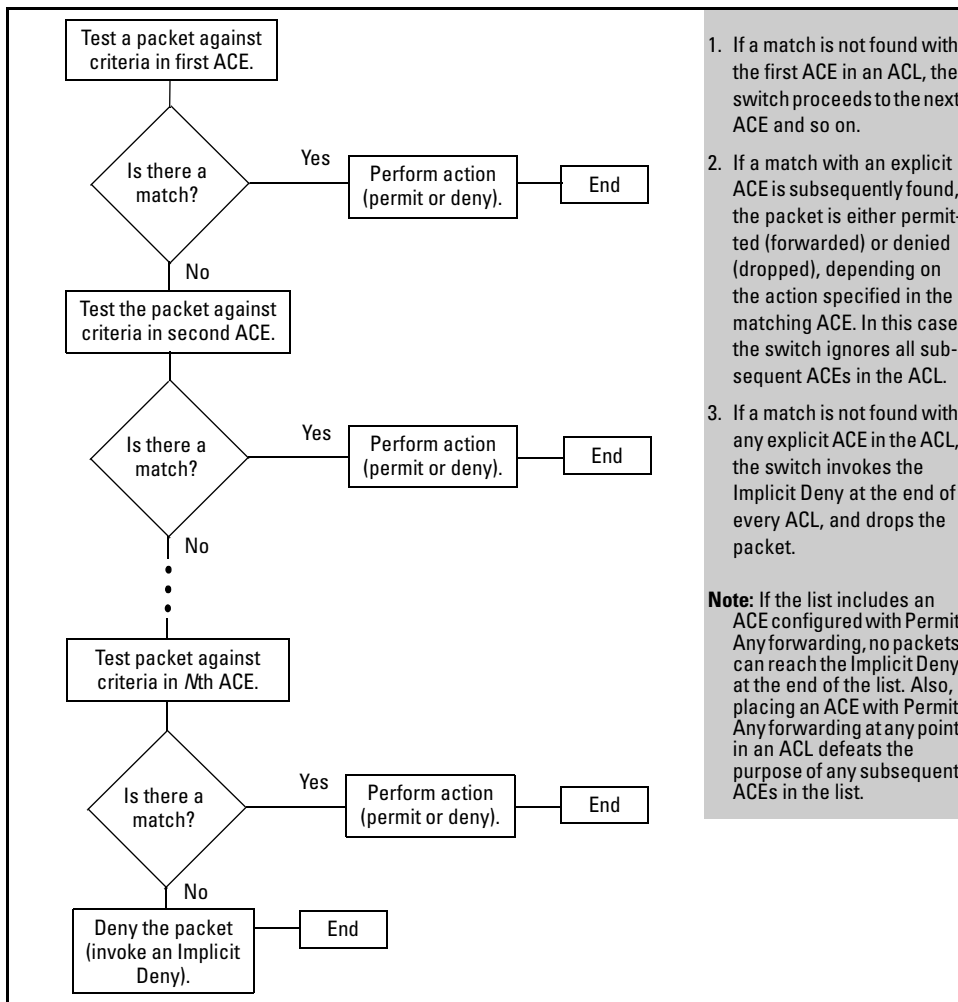


Figure 10-6. The Packet-Filtering Process in an ACL with *N* Entries (ACEs)

Note

The order in which an ACE occurs in an ACL is significant. For example, if an ACL contains six ACEs, but the first ACE allows Permit Any forwarding, then the ACL permits all IP traffic, and the remaining ACEs in the list do not apply, even if they specify criteria that would make a match with any of the IP traffic permitted by the first ACE.

For example, suppose you want to configure an ACL on the switch (with an ID of “Test-02”) to invoke these policies for routed IP traffic entering the switch on VLAN 12 :

1. Permit all inbound IP traffic from IP address 10.11.11.42.
2. Deny *only* the inbound Telnet traffic from address 10.11.11.101.
3. Permit *only* inbound Telnet traffic from IP address 10.11.11.33.
4. Deny *all other* inbound IP traffic.

The following ACL model , when assigned to inbound filtering on an interface, supports the above case:

```

ip access-list extended "Test-02"

  1 10 permit ip 10.11.11.42 0.0.0.0 0.0.0.0 255.255.255.255

  2 20 deny tcp 10.11.11.101 0.0.0.0 0.0.0.0 255.255.255.255 eq 23

  3 30 permit ip 10.11.11.101 0.0.0.0 0.0.0.0 255.255.255.255

  4 40 permit tcp 10.11.11.33 0.0.0.0 0.0.0.0 255.255.255.255 eq 23

  5 < Implicit Deny >
exit
ProCurve(config)# vlan 12 ip access-group Test-02 in

```

<p>1. Permits IP traffic from source address 10.11.11.42. Packets matching this criterion are permitted and will not be compared to any later ACE in the list. Packets not matching this criterion will be compared to the next entry in the list.</p>	<p>4. Permits Telnet traffic from source address 10.11.11.33. Packets matching this criterion are permitted and are not compared to any later criteria in the list. Packets not matching this criterion are compared to the next entry in the list.</p>
<p>2. Denies Telnet traffic from source address 10.11.11.101. Packets matching this criterion are dropped and are not compared to later criteria in the list. Packets not matching this criterion are compared to the next entry in the list.</p>	<p>5. This entry does not appear in an actual ACL, but is implicit as the last entry in every ACL. Any packets that do not match any of the criteria in the ACL's preceding entries will be denied (dropped), and will not cross VLAN 12.</p>
<p>3. Permits any IP traffic from source address 10.11.11.101. Any packets matching this criterion will be permitted and will not be compared to any later criteria in the list. Because this entry comes after the entry blocking Telnet traffic from this same address, there will not be any Telnet packets to compare with this entry; they have already been dropped as a result of matching the preceding entry.</p>	

Figure 10-7. Example of How an ACL Filters Packets

It is important to remember that all ACLs configurable on the switch include an implicit **deny ip any any**. That is, IP packets that the ACL does not *explicitly* permit or deny will be *implicitly* denied, and therefore dropped instead of forwarded on the interface. If you want to preempt the implicit deny so that packets not explicitly denied by other ACEs in the ACL will be permitted, insert an explicit “permit any” as the last ACE in the ACL. Doing so permits any packet not explicitly denied by earlier entries. (Note that this solution does not apply in the preceding example, where the intention is for the switch to forward only explicitly permitted packets routed on VLAN 12.

Planning an ACL Application

Before creating and implementing ACLs, you need to define the policies you want your ACLs to enforce, and understand how the ACL assignments will impact your network users.

Note

All IP traffic entering the switch on a given interface is filtered by all ACLs configured for inbound traffic on that interface. For this reason, an inbound packet will be denied (dropped) if it has a match with either an implicit or explicit **deny** in *any* of the inbound ACLs applied to the interface. (This does not apply to IP traffic leaving the switch because only one type of ACL—an RACL—can be applied, and only to routed IP traffic.)

(Refer to “Multiple ACLs on an Interface” on page 10-20.)

IP Traffic Management and Improved Network Performance

You can use ACLs to block IP traffic from individual hosts, workgroups, or subnets, and to block access to VLANs, subnets, devices, and services. Traffic criteria for ACLs include:

- Switched and/or routed IP traffic
- Any IP traffic of a specific protocol type (0-255)

- Any TCP traffic (only) for a specific TCP port or range of ports, including optional control of connection traffic based on whether the initial request should be allowed
- Any UDP traffic (only) or UDP traffic for a specific UDP port
- Any ICMP traffic (only) or ICMP traffic of a specific type and code
- Any IGMP traffic (only) or IGMP traffic of a specific type
- Any of the above with specific precedence and/or ToS settings

Depending on the source and/or destination of a given IP traffic type, you must also determine the ACL application(s) (RACL, VACL, or static port ACL) needed to filter the traffic on the applicable switch interfaces. Answering the following questions can help you to design and properly position ACLs for optimum network usage.

- What are the logical points for minimizing unwanted IP traffic, and what ACL application(s) should be used? In many cases it makes sense to prevent unwanted IP traffic from reaching the core of your network by configuring ACLs to drop unwanted IP traffic at or close to the edge of the network. (The earlier in the network path you can block unwanted IP traffic, the greater the benefit for network performance.)
- From where is the traffic coming? The source and destination of IP traffic you want to filter determines the ACL application to use (RACL, VACL, static port ACL, and dynamic port ACL).
- What IP traffic should you explicitly block? Depending on your network size and the access requirements of individual hosts, this can involve creating a large number of ACEs in a given ACL (or a large number of ACLs), which increases the complexity of your solution.
- What IP traffic can you implicitly block by taking advantage of the implicit **deny IP any** to deny IP traffic that you have not explicitly permitted? This can reduce the number of entries needed in an ACL.
- What IP traffic should you permit? In some cases you will need to explicitly identify permitted IP traffic. In other cases, depending on your policies, you can insert an ACE with “permit any” forwarding at the end of an ACL. This means that all IP traffic not specifically matched by earlier entries in the list will be permitted.

Security

ACLs can enhance security by blocking IP traffic carrying an unauthorized source IP address (SA). This can include:

- blocking access from specific devices or interfaces (port or VLAN)
- blocking access to or from subnets in your network
- blocking access to or from the internet
- blocking access to sensitive data storage or restricted equipment
- preventing specific IP, TCP, UDP, IGMP, and ICMP traffic types, including unauthorized access using functions such as Telnet, SSH, and web browser

You can also enhance switch management security by using ACLs to block IP traffic that has the switch itself as the destination address (DA).

Caution

ACLs can enhance network security by blocking selected IP traffic, and can serve as one aspect of maintaining network security. *However, because ACLs do not provide user or device authentication, or protection from malicious manipulation of data carried in IP packet transmissions, they should not be relied upon for a complete security solution.*

Note

ACLs in the switches covered by this guide do not filter non-IP traffic such as AppleTalk and IPX.

Guidelines for Planning the Structure of an ACL

After determining the filtering type (standard or extended) and ACL application (RACL, VACL, or static port ACL) to use at a particular point in your network, determine the order in which to apply individual ACEs to filter IP traffic (For information on ACL applications, refer to “ACL Applications” on page 10-15.).

- The sequence of ACEs is significant. When the switch uses an ACL to determine whether to permit or deny a packet on a particular VLAN, it compares the packet to the criteria specified in the individual

Access Control Entries (ACEs) in the ACL, beginning with the first ACE in the list and proceeding sequentially until a match is found. When a match is found, the switch applies the indicated action (permit or deny) to the packet.

- The first match in an ACL dictates the action on a packet. Subsequent matches in the same ACL are ignored. However, if a packet is permitted by one ACL assigned to an interface, but denied by another ACL assigned to the same interface, the packet will be denied on the interface.
- On any ACL, the switch implicitly denies IP packets that are not explicitly permitted or denied by the ACEs configured in the ACL. If you want the switch to forward a packet for which there is not a match in an ACL, append an ACE that enables Permit Any forwarding as the last ACE in an ACL. This ensures that no packets reach the Implicit Deny case for that ACL.
- Generally, you should list ACEs from the most specific (individual hosts) to the most general (subnets or groups of subnets) unless doing so permits IP traffic that you want dropped. For example, an ACE allowing a small group of workstations to use a specialized printer should occur earlier in an ACL than an entry used to block widespread access to the same printer.

ACL Configuration and Operating Rules

- **RACLs and Routed IP Traffic:** Except for any IP traffic with a DA on the switch itself, RACLs filter only routed IP traffic that is entering or leaving the switch on a given VLAN. Thus, if routing is not enabled on the switch, there is no routed IP traffic for RACLs to filter. For more on routing, refer to the chapter titled “IP Routing Features” in the *Multicast and Routing Guide* for your switch.
- **VACLs and Switched or Routed IP Traffic:** A VACL filters any IP traffic entering the switch on the VLAN(s) to which it is assigned.
- **Static Port ACLs:** A static port ACL filters any IP traffic entering the switch on the port(s) or trunk(s) to which it is assigned.
- **Per Switch ACL Limits for All ACL Types.** At a minimum an ACL must have one, explicit “permit” or “deny” Access Control Entry. You can configure up to 2048 ACL assignments, as follows:
 - Named (Extended or Standard) ACLs: Up to 2048 (minus any numeric standard or extended ACL assignments)

- Numeric Standard ACLs: Up to 99; numeric range: 1 - 99
 - Numeric Extended ACLs: Up to 100; numeric range: 100 - 199
 - Total ACEs in all ACLs: Depends on the combined resource usage by ACL, QoS, IDM, Virus-Throttling, ICMP, and Management VLAN features (For more on this topic, refer to “Monitoring Shared Resources” on page 10-118.)
-
- **Implicit Deny:** In any ACL, the switch automatically applies an implicit “deny IP any” that does not appear in **show** listings. This means that the ACL denies any packet it encounters that does not have a match with an entry in the ACL. Thus, if you want an ACL to permit any packets that you have not expressly denied, you must enter a **permit any** or **permit ip any any** as the last ACE in an ACL. Because, for a given packet the switch sequentially applies the ACEs in an ACL until it finds a match, any packet that reaches the **permit any** or **permit ip any any** entry will be permitted, and will not encounter the “deny ip any” ACE the switch automatically includes at the end of the ACL. For an example, refer to figure 10-7 on page 10-29.
 - **Explicitly Permitting Any IP Traffic:** Entering a **permit any** or a **permit ip any any** ACE in an ACL permits all IP traffic not previously permitted or denied by that ACL. Any ACEs listed after that point do not have any effect.
 - **Explicitly Denying Any IP Traffic:** Entering a **deny any** or a **deny ip any any** ACE in an ACL denies all IP traffic not previously permitted or denied by that ACL. Any ACEs listed after that point have no effect.
 - **Replacing One ACL with Another Using the Same Application:** For a specific interface, the most recent ACL assignment using a given application replaces any previous ACL assignment using the same application on the same interface. For example, if you configured an RACL named “100” to filter inbound routed IP traffic on VLAN 20, but later, you configured another RACL named 112 to filter inbound routed IP traffic on this same VLAN, RACL 112 replaces RACL 100 as the ACL to use.
 - **Static Port ACLs:** These are applied per-port, per port-list, or per static trunk. Adding a port to a trunk applies the trunk’s ACL configuration to the new member. If a port is configured with an ACL, the ACL must be removed before the port is added to the trunk. Also, removing a port from an ACL-configured trunk removes the ACL configuration from that port.

- **VACLs:** These filter any IP traffic entering the switch through any port belonging to the designated VLAN. VACLs do not filter IP traffic leaving the switch or being routed from another VLAN.
- **VACLs and RACLs Operate On Static VLANs:** You can assign an ACL to any VLAN that is statically configured on the switch. ACLs do not operate with dynamic VLANs.
- **A VACL or RACL Affects All Physical Ports in a Static VLAN:** A VACL or RACL assigned to a VLAN applies to all physical ports on the switch belonging to that VLAN, including ports that have dynamically joined the VLAN.
- **RACLs Screen Routed IP Traffic Entering or Leaving the Switch on a Given VLAN Interface:** This means that the following traffic is subject to ACL filtering:
 - IP traffic arriving on the switch through one VLAN and leaving the switch through another VLAN
 - IP traffic arriving on the switch through one subnet and leaving the switch through another subnet within the same, multinetted VLAN

Filtering the desired, routed IP traffic requires assigning an RACL to screen IP traffic inbound or outbound on the appropriate VLAN(s). In the case of a multinetted VLAN, it means that IP traffic inbound from different subnets in the same VLAN is screened by the same inbound RACL, and IP traffic outbound from different subnets is screened by the same outbound RACL. (Refer to figure 10-1 on page 10-17.)

- **RACLs Do Not Filter Switched IP Traffic Unless the Switch Itself is the SA or DA:** RACLs do *not* filter IP traffic moving between ports belonging to the same VLAN or subnet (in the case of a subnetted VLAN). (IP traffic moving between ports in different subnets of the same VLAN can be filtered.)

Note

RACLs do filter routed *or* switched IP traffic having an SA or DA on the switch itself.

How an ACE Uses a Mask To Screen Packets for Matches

When the switch applies an ACL to IP traffic, each ACE in the ACL uses an IP address and *ACL mask* to enforce a selection policy on the packets being screened. That is, the mask determines the range of IP addresses (SA only or SA/DA) that constitute a match between the policy and a packet being screened.

What Is the Difference Between Network (or Subnet) Masks and the Masks Used with ACLs?

In common IP addressing, a network (or subnet) mask defines which part of the IP address to use for the network number and which part to use for the hosts on the network. For example:

IP Address	Mask	Network Address	Host Address
10.38.252.195	255.255.255.0	first three octets	The fourth octet.
10.38.252.195	255.255.248.0	first two octets and the left-most five bits of the third octet	The right most three bits of the third octet and all bits in the fourth octet.

Thus, the bits set to 1 in a network mask define the part of an IP address to use for the network number, and the bits set to 0 in the mask define the part of the address to use for the host number.

In an ACL, IP addresses and masks provide criteria for determining whether to deny or permit a packet, or to pass it to the next ACE in the list. If there is a match, the configured deny or permit action occurs. If there is not a match, the packet is compared with the next ACE in the ACL. Thus, where a standard network mask defines how to identify the network and host numbers in an IP address, the mask used with ACEs defines which bits in a packet's IP address must match the corresponding bits in the IP address listed in an ACE, and which bits can be *wildcards*.

Rules for Defining a Match Between a Packet and an Access Control Entry (ACE)

- For a given ACE, when the switch compares an IP address and corresponding mask in the ACE to an IP address carried in a packet:
 - **A mask-bit setting of 0 (“off”)** requires that the corresponding bit in the packet’s IP address and in the ACE’s IP address must be the same. That is, if a bit in the ACE’s IP address is set to 1 (“on”), the same bit in the packet’s IP address must also be 1.
 - **A mask-bit setting of 1 (“on”)** means the corresponding bit in the packet’s IP address and in the ACE’s IP address do not have to be the same. That is, if a bit in the ACE’s IP address is set to 1, the same bit in the packet’s IP address can be either 1 or 0 (“on” or “off”).

For an example, refer to “Example of How the Mask Bit Settings Define a Match” on page 10-39.

- In any ACE, a mask of all ones means *any* IP address is a match. Conversely, a mask of all zeros means the *only* match is an IP address identical to the host IP address specified in the ACE.
- Depending on your network, a single ACE that allows a match with more than one source or destination IP address may allow a match with multiple subnets. For example, in a network with a prefix of 31.30.240 and a subnet mask of 255.255.240.0 (the leftmost 20 bits), applying an ACL mask of 0.0.31.255 causes the subnet mask and the ACL mask to overlap one bit, which allows matches with hosts in two subnets: 31.30.224.0 and 31.30.240.0.

Bit Position in the Third Octet of Subnet Mask 255.255.240.0								
Bit Values	128	64	32	16	8	4	2	1
Subnet Mask Bits	1	1	1	1	n/a	n/a	n/a	n/a
Mask Bit Settings Affecting Subnet Addresses	0	0	0	1 or 0	n/a	n/a	n/a	n/a

This ACL supernetting technique can help to reduce the number of ACLs you need. You can apply it to a multinetted VLAN and to multiple VLANs. However, ensure that you exclude subnets that do not belong in the policy. If this creates a problem for your network, you can eliminate the unwanted match by making the ACEs in your ACL as specific as possible, and using multiple ACEs carefully ordered to eliminate unwanted matches.

- Every IP address and mask pair (source or destination) used in an ACE creates one of the following policies:

- **Any IP address fits the matching criteria.** In this case, the switch automatically enters the IP address and mask in the ACE. For example:

```
access-list 1 deny any
```

produces this policy in an ACL listing:

IP Address	Mask
0.0.0.0	255.255.255.255

This policy states that every bit in every octet of a packet's SA is a wildcard, which covers any IP address.

- **One IP address fits the matching criteria.** In this case, you provide the IP address and the switch provides the mask. For example:

```
access-list 1 permit host 10.28.100.15
```

produces this policy in an ACL listing:

IP Address	Mask
10.28.100.15	0.0.0.0

This policy states that every bit in every octet of a packet's SA must be the same as the corresponding bit in the SA defined in the ACE.

- **A group of IP addresses fits the matching criteria.** In this case you provide both the IP address and the mask. For example:

```
access-list 1 permit 10.28.32.1 0.0.0.31
```

IP Address	Mask
10.28.32.1	0.0.0.31

This policy states that:

- In the first three octets of a packet's SA, every bit must be set the same as the corresponding bit in the SA defined in the ACE.
- In the last octet of a packet's SA, the first three bits must be the same as in the ACE, but the last five bits are wildcards and can be any value.

- Unlike subnet masks, the wildcard bits in an ACL mask need not be contiguous. For example, 0.0.7.31 is a valid ACL mask. However, a subnet mask of 255.255.248.224 is not a valid subnet mask.

Example of How the Mask Bit Settings Define a Match . Assume an ACE where the second octet of the mask for an SA is 7 (the rightmost three bits are “on”, or “1”) and the second octet of the corresponding SA in the ACE is 31 (the rightmost five bits). In this case, a match occurs when the second octet of the SA in a packet being filtered has a value in the range of 24 to 31. Refer to table 10-4, below.

Table 10-4. Example of How the Mask Defines a Match

Location of Octet	Bit Position in the Octet							
	128	64	32	16	8	4	2	1
SA in ACE	0	0	0	1	1	1	1	1
Mask for SA	0	0	0	0	0	1	1	1
Corresponding Octet of a Packet's SA	0	0	0	1	1	0/1	0/1	0/1

The shaded area indicates bits in the packet that must exactly match the bits in the source IP in the ACE. Wherever the mask bits are ones (wildcards), the IP bits in the packet can be any value, and where the mask bits are zeros, the IP bits in the packet must be the same as the IP bits in the ACE. **Note:** This example covers only one octet of an IP address. An actual ACE applies this method to all four octets of an IP address.

Example of Allowing Only One IP Address (“Host” Option). Suppose, for example, that you have configured the ACL in figure 10-8 to filter inbound packets on VLAN 20. Because the mask is all zeros, the ACE policy dictates that a match occurs only when the source IP address on such packets is identical to the IP address configured in the ACE.

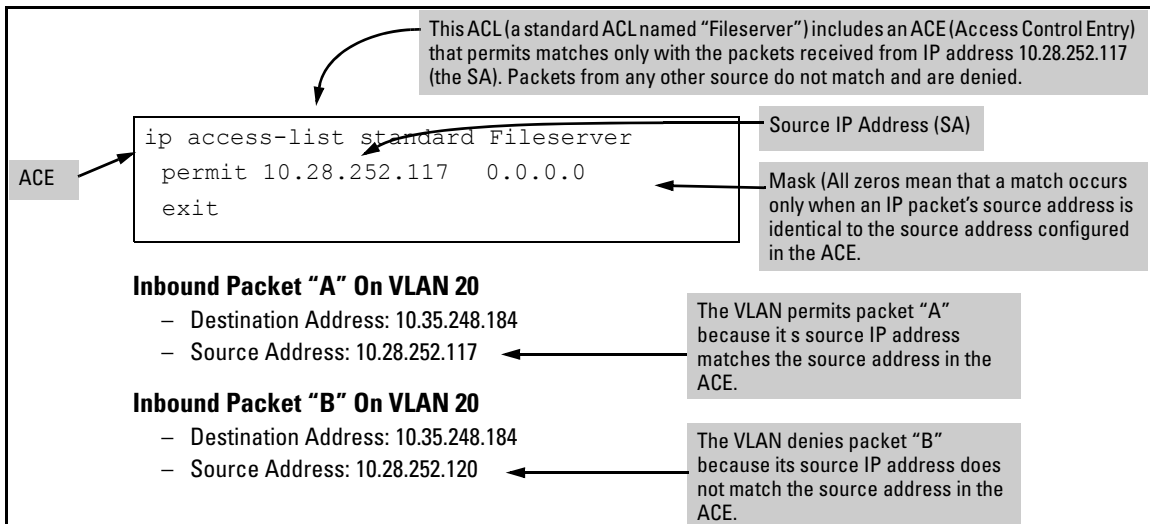


Figure 10-8. Example of an ACL with an Access Control Entry (ACE) that Allows Only One Source IP Address

Examples Allowing Multiple IP Addresses. Table 10-5 provides examples of how to apply masks to meet various filtering requirements.

Table 10-5. Example of Using an IP Address and Mask in an Access Control Entry

IP Address in the ACE	Mask	Policy for a Match Between a Packet and the ACE	Allowed IP Addresses
A: 10.38.252.195	0.0.0.255	Exact match in first three octets only.	10.38.252.< 0-255 > (See row A in table 10-6, below.)
B: 10.38.252.195	0.0.7.255	Exact match in the first two octets and the leftmost five bits (248) of the third octet.	10.38.< 248-255 >.< 0-255 > (In the third octet, only the rightmost three bits are wildcard bits. The leftmost five bits must be a match, and in the ACE, these bits are all set to 1. See row B in table 10-6, below.)
C: 10.38.252.195	0.0.0.0	Exact match in all octets.	10.38.252.195 (There are no wildcard bits in any of the octets. See row C in table 10-6, below.)
D: 10.38.252.195	0.15.255.255	Exact match in the first octet and the leftmost four bits of the second octet.	10.< 32-47 >.< 0-255 >.< 0-255 > (In the second octet, the rightmost four bits are wildcard bits. See row D in table 10-6, below.)

Table 10-6. Mask Effect on Selected Octets of the IP Addresses in Table 10-5

IP Addr	Octet	Mask	Octet Range	128	64	32	16	8	4	2	1
A	3	0 all bits	252	1	1	1	1	1	1	0	0
B	3	7 last 3 bits	248-255	1	1	1	1	1	0 or 1	0 or 1	0 or 1
C	4	0 all bits	195	1	1	0	0	0	0	1	1
D	2	15 last 4 bits	32-47	0	0	1	0	0 or 1	0 or 1	0 or 1	0 or 1

Shaded areas indicate bit settings that must be an exact match.

If there is a match between the policy in the ACE and the IP address in a packet, then the packet is either permitted or denied, according to how the ACE is configured. If there is not a match, the next ACE in the ACL is then applied to the packet. The same operation applies to a destination IP address (DA) used in an extended ACE. (Where an ACE includes both source and destination IP addresses, there is one IP-address/ACL-mask pair for the source address, and another IP-address/ACL-mask pair for the destination address. See “Configuring and Assigning an ACL” on page 10-41.)

CIDR Notation. For information on using CIDR notation to specify ACL masks, refer to “Using CIDR Notation To Enter the ACL Mask” on page 10-50.

Configuring and Assigning an ACL

ACL Feature	Page
Configuring and Assigning a Standard ACL	10-51
Configuring and Assigning an Extended ACL	10-60
Enabling or Disabling ACL Filtering	10-81

Overview

General Steps for Implementing ACLs

1. Configure one or more ACLs. This creates and stores the ACL(s) in the switch configuration.
2. Assign an ACL. This step uses one of the following applications to assign the ACL to an interface:
 - RACL (routed IP traffic entering or leaving the switch on a given VLAN)
 - VACL (any IP traffic entering the switch on a given VLAN)
 - Static Port ACL (any IP traffic entering the switch on a given port, port list, or static trunk)
3. If the ACL is applied as an RACL, enable IP routing. Except for instances where the switch is the traffic source or destination, assigned RACLs filter IP traffic only when routing is enabled on the switch.

Caution Regarding the Use of Source Routing

Source routing is enabled by default on the switch and can be used to override ACLs. For this reason, if you are using ACLs to enhance network security, the recommended action is to disable source routing on the switch. To do so, execute **no ip source-route**.

Options for Permit/Deny Policies

The permit or deny policy for IP traffic you want to filter can be based on source IP address alone, or on source IP address plus other IP factors.

- **Standard ACL:** Uses only a packet's source IP address as a criterion for permitting or denying the packet. For a standard ACL ID, use either a unique numeric string in the range of 1-99 or a unique name string of up to 64 alphanumeric characters.
- **Extended ACL:** Offers the following criteria as options for permitting or denying a packet:
 - source IP address
 - destination IP address
 - IP protocol options:
 - Any IP traffic
 - Any IP traffic of a specific protocol type (0-255)
 - Any TCP traffic (only) for a specific TCP port or range of ports, including optional control of connection traffic based on whether the initial request should be allowed
 - Any UDP traffic (only) or UDP traffic for a specific UDP port
 - Any ICMP traffic (only) or ICMP traffic of a specific type and code
 - Any IGMP traffic (only) or IGMP traffic of a specific type
 - Any of the above with specific precedence and/or ToS settings

For an extended ACL ID, use either a unique number in the range of 100-199 or a unique name string of up to 64 alphanumeric characters.

Carefully plan ACL applications before configuring specific ACLs. For more on this topic, refer to “Planning an ACL Application” on page 10-30.

ACL Configuration Structure

After you enter an ACL command, you may want to inspect the resulting configuration. This is especially true where you are entering multiple ACEs into an ACL. Also, it is helpful to understand the configuration structure when using later sections in this chapter.

The basic ACL structure includes four elements:

1. ACL identity and type: This identifies the ACL as **standard** or **extended** and shows the ACL name or number.
2. Optional **remark** entries.

3. One or more deny/permit list entries (ACEs): One entry per line.

Element	Notes
Type	Standard or Extended
Identifier	<ul style="list-style-type: none"> • Alphanumeric; Up to 64 Characters, Including Spaces • Numeric: 1 - 99 (Standard) or 100 - 199 (Extended)
Remark	Allows up to 100 alphanumeric characters, including blank spaces. (If any spaces are used, the remark must be enclosed in a pair of single or double quotes.) A remark is associated with a particular ACE and will have the same sequence number as the ACE. (One remark is allowed per ACE.) Refer to "Attaching a Remark to an ACE" on page 10-96.
Maximum ACEs Per per Switch	The upper limit on ACEs supported by the switch depends on the concurrent resource usage by configured QoS, ICMP rate-limiting, management VLAN, and virus-throttling features. Refer to "Monitoring Shared Resources" on page 10-118.

4. **Implicit Deny:** Where an ACL is in use, it denies any packets that do not have a match with the ACEs explicitly configured in the list. The Implicit Deny does not appear in ACL configuration listings, but always functions when the switch uses an ACL to filter packets. (You cannot delete the Implicit Deny, but you can supersede it with a **permit any** or **permit ip any any** statement.)

Standard ACL Structure

Individual ACEs in a standard ACL include only a permit/deny statement, the source IP addressing, and an optional **log** command (available with "deny" statements).

```

ip access-list standard < identifier >
  [ [ seq-# ] remark < remark-str > ]
  < permit | deny > < SA > [ log ]
  .
  .
  .
  < Implicit Deny >
exit

```

Note: The optional **log** function is available only for explicit "deny" ACEs.

Figure 10-9. Example of the General Structure for a Standard ACL

Access Control Lists (ACLs)

Configuring and Assigning an ACL

For example, figure 10-10 shows how to interpret the entries in a standard ACL.

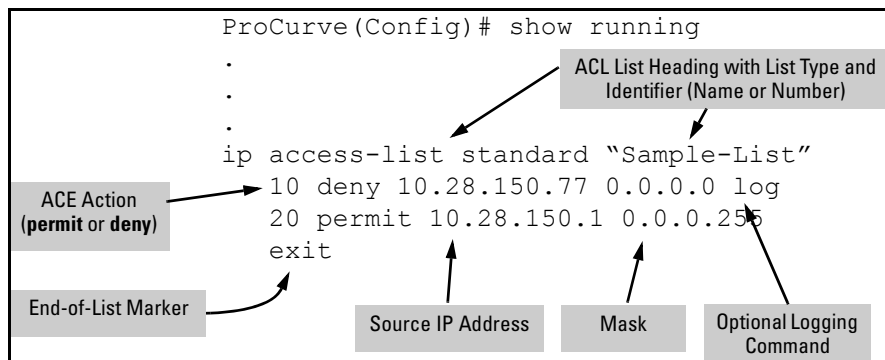


Figure 10-10. Example of a Displayed Standard ACL Configuration with Two ACEs

Extended ACL Configuration Structure

Individual ACEs in an extended ACL include:

- A permit/deny statement
- Source and destination IP addressing
- Choice of IP criteria, including optional precedence and ToS
- Optional ACL **log** command (for **deny** entries)
- Optional remark statements

```
ip access-list extended < identifier >
[[ seq-#] remark < remark-str >]
< permit | deny > < ip-type > < SA > < src-acl-mask > < DA > < dest-acl-mask > [log]

    < permit | deny > tcp
        < SA > < src-acl-mask > [< operator > < port-id >]
        < DA > < desti-acl-mask > [< operator > < port-id >] [log]
        [ established ]

    < permit | deny > udp
        < SA > < src-acl-mask > [< operator > < port-id >]
        < DA > < dest-acl-mask > [< operator > < port-id >] [log]

    < permit | deny > icmp
        < SA > < src-acl-mask > < DA > < dest-acl-mask > [ icmp-type ] [log]

    < permit | deny > igmp
        < SA > < SA-mask > < DA > < dest-acl-mask > [ igmp-type ] [log]

    [ precedence < priority >]
    [ tos < tos-setting >]
    . . .
    < Implicit Deny >
    exit
```

Note: The optional **log** function appears only with “deny” ACEs.

Figure 10-11. Example of General Structure Options for an Extended ACL

Access Control Lists (ACLs) Configuring and Assigning an ACL

For example, figure 10-12 shows how to interpret the entries in an extended ACL.

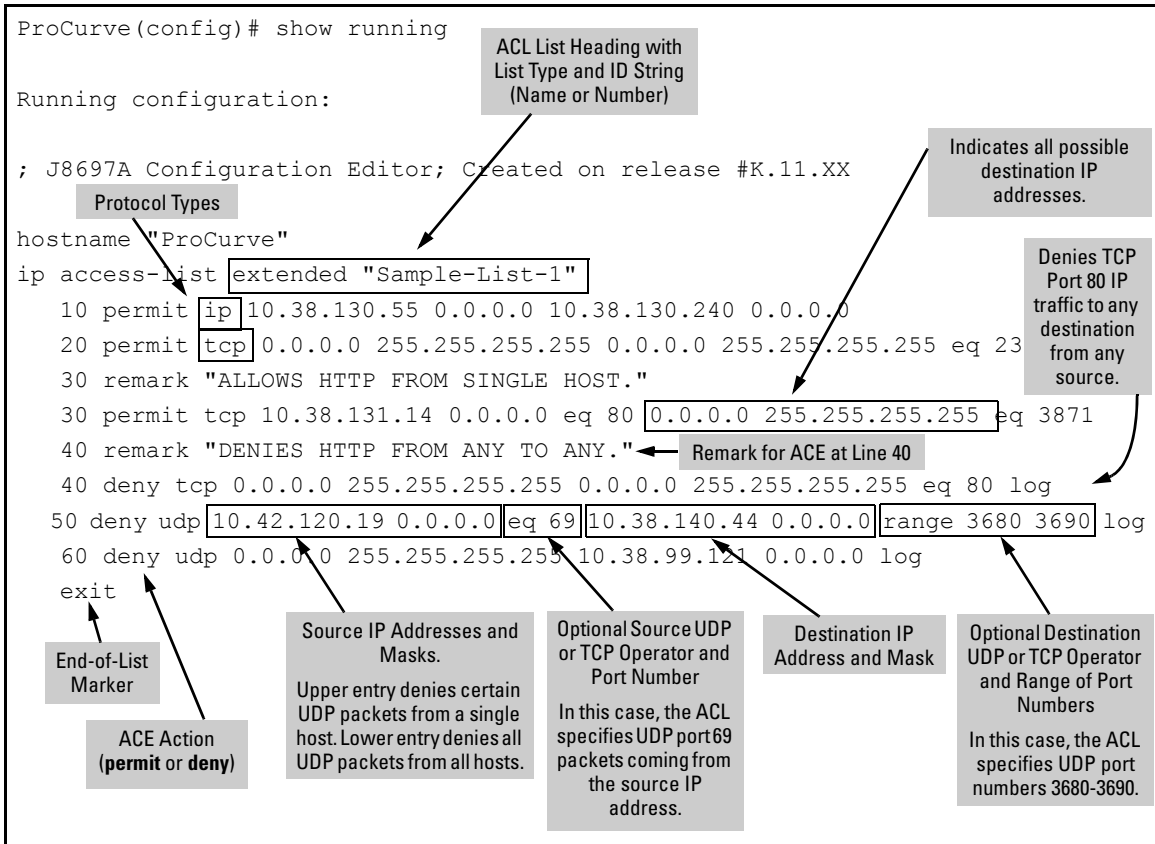


Figure 10-12. Example of a Displayed Extended ACL Configuration

ACL Configuration Factors

The Sequence of Entries in an ACL Is Significant

When the switch uses an ACL to determine whether to permit or deny a packet, it compares the packet to the criteria specified in the individual Access Control Entries (ACEs) in the ACL, beginning with the first ACE in the list and proceeding sequentially until a match is found. When a match is found, the switch applies the indicated action (permit or deny) to the packet. This is

significant because, once a match is found for a packet, subsequent ACEs in the same ACL will not be applied to that packet, regardless of whether they match the packet.

For example, suppose that you have applied the ACL shown in figure 10-13 to inbound IP traffic on VLAN 1 (the default VLAN):

```

ip access-list extended "Sample-List-2"
 10 deny ip 10.28.235.10 0.0.0.0 0.0.0.0 255.255.255.255
 20 deny ip 10.28.245.89 0.0.0.0 0.0.0.0 255.255.255.255
 30 permit tcp 10.28.18.100 0.0.0.0 10.28.237.1 0.0.0.0
 40 deny tcp 10.28.18.100 0.0.0.0 0.0.0.0 255.255.255.255
 50 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255

(implicit deny) ←
exit
  
```

The diagram shows the ACL configuration with labels for Source IP, Mask, Destination IP, and Mask pointing to the corresponding fields in the ACL entries. The labels are placed above the configuration and arrows point to the respective fields in the ACL entries.

Figure 10-13. Example of a Standard ACL that Permits All IP Traffic Not Implicitly Denied

Table 10-7. Effect of the Above ACL on Inbound IP Traffic in the Assigned VLAN

Line #	Action
n/a	Shows type (extended) and ID (Sample-List-2).
10	A packet from IP source address 10.28.235.10 will be denied (dropped). This ACE filters out all packets received from 10.28.235.10. As a result, IP traffic from that device will not be allowed and packets from that device will not be compared against any later entries in the list.
20	A packet from IP source 10.28.245.89 will be denied (dropped). This ACE filters out all packets received from 10.28.245.89. As the result, IP traffic from that device will not be allowed and packets from that device will not be compared against any later entries in the list.
30	A TCP packet from SA 10.28.18.100 with a DA of 10.28.237.1 will be permitted (forwarded). Since no earlier ACEs in the list have filtered TCP packets from 10.28.18.100 and destined for 10.28.237.1, the switch will use this ACE to evaluate such packets. Any packets that meet this criteria will be forwarded. (Any packets that do not meet this TCP source-destination criteria are not affected by this ACE.)
40	A TCP packet from source address 10.28.18.100 to any destination address will be denied (dropped). Since, in this example, the intent is to block TCP traffic from 10.28.18.100 to any destination except the destination stated in the ACE at line 30, this ACE must follow the ACE at line 30. (If their relative positions were exchanged, all TCP traffic from 10.28.18.100 would be dropped, including the traffic for the 10.28.18.1 destination.)
50	Any packet from any IP source address to any destination address will be permitted (forwarded). The only traffic to reach this ACE will be IP packets not specifically permitted or denied by the earlier ACEs.

Line #	Action
<i>n/a</i>	The <i>Implicit Deny</i> is a function the switch automatically adds as the last action in all ACLs. It denies (drops) any IP traffic from any source to any destination that has not found a match with earlier entries in the ACL. In this example, the ACE at line 50 permits (forwards) any IP traffic not already permitted or denied by the earlier entries in the list, so there is no traffic remaining for action by the Implicit Deny function.
exit	Marks the end of the ACL.

Allowing for the Implied Deny Function

In any ACL having one or more ACEs there will always be a packet match. This is because the switch automatically applies an Implicit Deny as the last ACE in any ACL. This function is not visible in ACL listings, but is always present. (Refer to figure 10-13.) This means that if you configure the switch to use an ACL for filtering either inbound or outbound IP traffic on a VLAN, any packets not specifically permitted or denied by the explicit entries you create will be denied by the Implicit Deny action. If you want to preempt the Implicit Deny (so that IP traffic not specifically addressed by earlier ACEs in a given ACL will be permitted), insert an explicit **permit any** (for standard ACLs) or **permit ip any any** (for extended ACLs) as the last explicit ACE in the ACL.

A Configured ACL Has No Effect Until You Apply It to an Interface

The switch stores ACLs in the configuration file. Thus, until you actually assign an ACL to an interface, it is present in the configuration, but not used (and does not use any of the monitored resources described in the appendix titled “Monitored Resources” in the *Management and Configuration Guide* for your switch.)

You Can Assign an ACL Name or Number to an Interface Even if the ACL Does Not Exist in the Switch’s Configuration

In this case, if you subsequently create an ACL with that name or number, the switch automatically applies each ACE as soon as you enter it in the running-config file. Similarly, if you modify an existing ACE in an ACL you already applied to an interface, the switch automatically implements the new ACE as soon as you enter it. (See “General ACL Operating Notes” on page 10-117.) The switch allows a maximum of 2048 ACLs in any combination of numeric and alphanumeric names, and determines the total from the number of unique ACL names in the configuration. For example, if you configure two ACLs, but assign only one of them to a VLAN, the ACL total is two, for the two unique ACL names. If you then assign the name of a nonexistent ACL to a VLAN, the new ACL total is three, because the switch now has three unique ACL names in its configuration.

Using the CLI To Create an ACL

Command	Page
access-list (standard ACLs)	10-51
access-list (extended ACLs)	10-60

You can use either the switch CLI or an offline text editor to create an ACL. This section describes the CLI method, which is recommended for creating short ACLs. (To use the offline method, refer to “Creating or Editing ACLs Offline” on page 10-108.)

General ACE Rules

These rules apply to all ACEs you create or edit using the CLI:

- Inserting or adding an ACE to an ACL:
 - **Named ACLs:** Add an ACE to the end of a named ACE by using the **ip access-list** command to enter the Named ACL (**nacl**) context and entering the ACE without the sequence number. For example, if you wanted to add a “permit” ACL at the end of a list named “List-1” to allow IP traffic from the device at 10.10.10.100:

```
ProCurve(config)# ip access-list standard List-1
ProCurve(config-std-nacl)# permit host
10.10.10.100
```

Insert an ACE anywhere in a named ACL by specifying a sequence number. For example, if you wanted to insert a new ACE as line 15 between lines 10 and 20 in an existing ACL named “List-2” to deny IP traffic from the device at 10.10.10.77:

```
ProCurve(config)# ip access-list standard List-2
ProCurve(config-std-nacl)# 15 deny host 10.10.10.77
```

- **Numbered ACLs:** Add an ACE to the end of a numbered ACL by using the **access-list <1-99|100-199>** command. For example, if you wanted to add a “permit” ACE at the end of a list identified with the number “11” to allow IP traffic from the device at 10.10.10.100:

```
ProCurve(config)# access-list 11 permit host
10.10.10.100
```

To insert an ACE *anywhere* in a numbered ACL, use the same process as described above for inserting an ACE anywhere in a *named* ACL. For example, to insert an ACE denying IP traffic from the host at 10.10.10.77 as line 52 in an existing ACL identified (named) with the number 11:

```
ProCurve(config)# ip access-list standard 99
ProCurve(config-std-nacl)# 52 deny host 10.10.10.77
```

Note

After a numbered ACL has been created (using **access-list < 1 - 99 | 100 - 199 >**), it can be managed as either a named or numbered ACL, as shown above.

- Deleting an ACE: Enter the ACL context and delete the sequence number for the unwanted ACE. (To view the sequence numbers of the ACEs in a list, use **show access-list < acl-name-str >**.)
- Duplicate ACEs are not allowed in the same ACL. Attempting to enter a duplicate ACE displays the **Duplicate access control entry** message.

Using CIDR Notation To Enter the ACL Mask

You can use CIDR (Classless Inter-Domain Routing) notation to enter ACL masks. The switch interprets the bits specified with CIDR notation as the IP address bits in an ACL and the corresponding IP address bits in a packet that must match. The switch then converts the mask to inverse notation for ACL use.

Table 10-8. Examples of CIDR Notation for Masks

IP Address Used In an ACL with CIDR Notation	Resulting ACL Mask	Meaning
10.38.240.125/15	0.1.255.255	The leftmost 15 bits must match; the remaining bits are wildcards.
10.38.240.125/20	0.0.15.255	The leftmost 20 bits must match; the remaining bits are wildcards.
10.38.240.125/21	0.0.7.255	The leftmost 21 bits must match; the remaining bits are wildcards.
10.38.240.125/24	0.0.0.255	The leftmost 24 bits must match; the remaining bits are wildcards.
18.38.240.125/32	0.0.0.0	All bits must match.

A standard ACL uses only source IP addresses in its ACEs. This type of ACE is useful when you need to:

- Permit or deny any IP traffic based on source IP address only.
- Quickly control the IP traffic from a specific address. This allows you to isolate IP traffic problems generated by a specific device, group of devices, or a subnet threatening to degrade network performance. This gives you an opportunity to troubleshoot without sacrificing performance for users outside of the problem area.

A *named*, standard ACL is identified by an alphanumeric string of up to 64 characters and is created by entering the Named ACL (**nacl**) context. A *numbered*, standard ACL is identified by a number in the range of 1 - 99 and is created without having to leave the global config context. Note that the CLI command syntax for creating a named ACL differs from the command syntax for creating a numbered ACL. For example, the first pair of entries below illustrate how to create (or enter) a named, standard ACL and enter an ACE. The next entry illustrates creating a numbered, standard ACL with the same ACE.

```
ProCurve(config)# ip access-list standard Test-List
ProCurve(config-std-nacl)# permit host 10.10.10.147
```

```
ProCurve(config)# access-list 1 permit host 10.10.10.147
```

Note that once a numbered ACL has been created, it can be accessed using the named ACL method. This is useful if it becomes necessary to edit a numbered ACL by inserting or removing individual ACEs. (Inserting or deleting an ACE is done by sequence number, and requires the Named ACL (**nacl**) context.) The switch allows a maximum of 2048 unique ACL identities; standard and extended combined.

Note

For a summary of standard ACL commands, refer to table 10-9 on page 10-51. For a summary of all ACL commands, refer to tables 10-1 and 10-2 on pages 10-6 and 10-8.

Configuring Named, Standard ACLs

This section describes the commands for performing the following:

- creating and/or entering the context of a named, standard ACL
- appending an ACE to the end of an existing list or entering the first ACE in a new list

For other ACL topics, refer to the following:

Topic	Page
configuring numbered, standard ACLs	10-56
configuring named, extended ACLs	10-62
configuring numbered, extended ACLs	10-74
applying or removing an ACL on an interface	10-81
deleting an ACL	10-89
editing an ACL	10-90
sequence numbering in ACLs	10-91
including remarks in an ACL	10-96
displaying ACL configuration data	10-100
creating or editing ACLs offline	10-108
enabling ACL "Deny" logging	10-113

Entering the "Named ACL" (nacl) Context. This command is a prerequisite to entering or editing ACEs in a named ACL.

Syntax: ip access-list standard < name-str >

Places the CLI in the "Named ACL" (nacl) context specified by the < name-str > alphanumeric identifier. This enables entry of individual ACEs in the specified ACL. If the ACL does not already exist, this command creates it.

< name-str >: Specifies an identifier for the ACL. Consists of an alphanumeric string of up to 64 case-sensitive characters. Including spaces in the string requires that you enclose the string in single or double quotes. For example: "Accounting ACL".

Refer also to table 10-9 on page 10-51.

Configuring ACEs in a Named, Standard ACL. Configuring ACEs is done after using the **ip access-list standard < name-str >** command described above to enter the “Named ACL” (**nacl**) context of an access list. *For a standard ACL syntax summary, refer to table 10-9 on page 10-51.*

Syntax: < deny | permit >
< any | host < SA > | SA < mask | SA / mask-length >> [log]

*Executing this command appends the ACE to the end of the list of ACEs in the current ACL. In the default ACL configuration, ACEs are automatically assigned consecutive sequence numbers in increments of 10 and can be renumbered using **resequence** (page 10-95).*

Note: *To insert a new ACE between two existing ACEs, precede **deny** or **permit** with an appropriate sequence number. (Refer to “Inserting an ACE in an Existing ACL” on page 10-92.)*

< deny | permit >

*For named ACLs, used in the “Named ACL” (**nacl**) context to configure an ACE. Specifies whether the ACE denies or permits a packet matching the criteria in the ACE, as described below.*

< any | host < SA > | SA < mask > | SA / mask-length >

Defines the source IP address (SA) a packet must carry for a match with the ACE.

- **any** — Allows IP packets from any SA.
- **host < SA >** — Specifies only packets having < SA > as the source. Use this criterion when you want to match the IP packets from a single source IP address.
- **SA < mask >** or **SA / mask-length** — Specifies packets received from either a subnet or a group of IP addresses. The mask format can be in either dotted-decimal format or CIDR format (number of significant bits). (Refer to “Using CIDR Notation To Enter the ACL Mask” on page 10-50).

Mask Application: *The mask is applied to the IP address in the ACE to define which bits in a packet’s source IP address must exactly match the IP address configured in the ACE and which bits need not match. For example: **10.10.10.1/24** and **10.10.10.1 0.0.0.255** both define any IP address in the range of 10.10.10.(1 - 255).*

Note: *Specifying a group of contiguous IP addresses may require more than one ACE. For more on how masks operate, refer to “How an ACE Uses a Mask To Screen Packets for Matches” on page 10-36.*

[log]

This option generates an ACL log message if:

- *The action is deny.*
- *There is a match.*
- *ACL logging is enabled on the switch. (Refer to “Enable ACL “Deny” Logging” on page 10-113.)*

(Use the debug command to direct ACL logging output to the current console session and/or to a Syslog server. Note that you must also use the logging < ip-addr > command to specify the IP addresses of Syslog servers to which you want log messages sent. See also “Enable ACL “Deny” Logging” on page 10-113.)

Example of Creating and Listing a Standard, Named ACL. This example illustrates how to create a standard, named ACL with several ACEs. This example creates an ACL that:

1. permits IP traffic from a host with the IP address of 10.10.10.104
2. creates another ACE that blocks all other IP traffic from the same subnet
3. allows all other IP traffic

<pre>ProCurve(config)# ip access-list standard Sample-List ProCurve(config-std-nacl)# permit host 10.10.10.104 ProCurve(config-std-nacl)# deny 10.10.10.1/24 log ProCurve(config-std-nacl)# permit any ProCurve(config-std-nacl)# exit ProCurve(config)# _</pre>	<p>Creates the “Sample-List” ACL and enters the “Named ACL” context for this list.</p> <p>Appends three ACEs to the list in the order shown.</p> <p>Exits from the nacl context.</p>
--	--

Figure 10-14. Example of Commands Used To Create a Standard, Named ACL

```
ProCurve(config)# show access-list Sample-List

Access Control Lists

Name: Sample-List
Type: Standard
Applied: No

SEQ  Entry
-----
10   Action: permit
     IP      : 10.10.10.104      Mask: 0.0.0.0
20   Action: deny (log)
     IP      : 10.10.10.1       Mask: 0.0.0.255
30   Action: permit
     IP      : 0.0.0.0          Mask: 255.255.255.255
```

Note that each ACE is automatically assigned a sequence number.

Figure 10-15. Screen Output Listing the “Sample-List” ACL Content

Creating Numbered, Standard ACLs

Use the following general steps to create or add to a numbered, standard ACL:

1. Create a numbered, standard ACL by entering the first ACE in the list
2. Append a new ACE to the end of an existing, standard ACL

This section describes the commands for performing these steps. For other ACL topics, refer to the following:

Topic	Page
configuring named, standard ACLs	10-53
configuring named, extended ACLs	10-62
configuring numbered, extended ACLs	10-74
applying or removing an ACL on an interface	10-81
deleting an ACL	10-89
editing an ACL	10-90
sequence numbering in ACLs	10-91
including remarks in an ACL	10-96
displaying ACL configuration data	10-100
creating or editing ACLs offline	10-108
enabling ACL “Deny” logging	10-113

Creating or Adding to a Standard, Numbered ACL. *This command is an alternative to using `ip access-list standard < name-str >` and does not use the “Named ACL” (**nacl**) context. For a standard ACL syntax summary, refer to table 10-9 on page 10-51.*

Syntax: `access-list < 1-99 > < deny | permit >
< any | host < SA > | SA < mask | SA/mask-length >> [log]`

*Appends an ACE to the end of the list of ACEs in the current standard, numbered ACL. If the ACL does not already exist, creates both the ACL and its first ACE. In the default configuration, ACEs are automatically assigned consecutive sequence numbers in increments of 10 and can be renumbered using **resequence** (page 10-95).*

Note: *To insert a new ACE between two existing ACEs in a standard, numbered ACL:*

- a. *Use **ip access list extended < 1 - 99 >** to open the ACL as a named ACL.*
- b. *Enter the desired sequence number along with the ACE keywords and variables you want.*

(After a numbered ACL has been created, it can be managed as either a named or numbered ACL. Refer to the “Numbered ACLs” list item on page 10-49.)

`< 1-99 >`

*Specifies the ACL identifier as a number. The switch interprets an ACL with a value in this range as a standard ACL (which filters all IP traffic on the basis of SA). (To create a standard access list with an alphanumeric name (**name-str**) instead of a number, refer to “Configuring Named, Standard ACLs” on page 10-53.)*

`< deny | permit >`

Specifies whether the ACE denies or permits a packet matching the criteria in the ACE, as described next.

< any | host < SA > | SA < mask | SA/mask-length >>

Defines the source IP address (SA) a packet must carry for a match with the ACE.

- **any** — Allows IP packets from any SA.
- **host < SA >** — Specifies only packets having < SA > as the source. Use this criterion when you want to match only the IP packets from a single SA.
- **SA < mask > or SA /mask-length** — Specifies packets received from an SA, where the SA is either a subnet or a group of IP addresses. The mask format can be in either dotted-decimal format or CIDR format (number of significant bits). (Refer to “Using CIDR Notation To Enter the ACL Mask” on page 10-50).

SA Mask Application: *The mask is applied to the SA in the ACE to define which bits in a packet’s SA must exactly match the SA configured in the ACL and which bits need not match.*

Example: **10.10.10.1/24** and **10.10.10.1 0.0.0.255** both define any IP address in the range of 10.10.10.(1 - 255).

Note: *Specifying a group of contiguous IP addresses may require more than one ACE. For more on how masks operate in ACLs, refer to “How an ACE Uses a Mask To Screen Packets for Matches” on page 10-36.*

Example of Creating and Viewing a Standard ACL. This example creates a standard, numbered ACL with the same ACE content as show in figure 10-14 on page 10-55.

```
ProCurve(config)# access-list 17 permit host 10.10.10.104
ProCurve(config)# access-list 17 deny 10.10.10.1/24 log
ProCurve(config)# access-list 17 permit any
ProCurve(config)# show access-list 17
```

Access Control Lists

Name: 17
Type: Standard
Applied: No

SEQ Entry

```
-----
10  Action: permit
    IP      : 10.10.10.104      Mask: 0.0.0.0

20  Action: deny (log)
    IP      : 10.10.10.1       Mask: 0.0.0.255

30  Action: permit
    IP      : 0.0.0.0          Mask: 255.255.255.255
```

Note that each ACE is automatically assigned a sequence number.

Figure 10-16. Standard, Numbered ACL with the Same ACEs as the Standard, Named ACL in Figure 10-14

Configuring Extended ACLs

Table 10-10. Command Summary for Extended ACLs

Action	Command(s)	Page
Create an Extended, Named ACL <i>or</i> Add an ACE to the End of an Existing, Extended ACL	<pre>ProCurve(config)# ip access-list extended < name-str 100-199 > ProCurve(config-std-nacl)# < deny permit > < ip ip-protocol ip-protocol-nbr > < any host <SA> SA/< mask-length > SA < mask >>¹ < any host <DA> DA/< mask-length > DA < mask >>¹ [tcp udp] < any host <SA> SA/< mask-length > SA < mask >>¹ [comparison-operator < value >] < any host <DA> DA/< mask-length > DA < mask >>¹ [comparison-operator < value >] [established] < igmp > < any host <SA> SA/< mask-length > SA < mask >>¹ < any host <DA> DA/< mask-length > DA < mask >>¹ [igmp-packet-type] < icmp > < any host <SA> SA/< mask-length > SA < mask >>¹ < any host <DA> DA/< mask-length > DA < mask >>¹ [[< 0 - 255 > [0 - 255]] icmp-message] [precedence < priority >] [tos < tos- setting >] [log]²</pre>	10-62
Create an Extended, Numbered ACL <i>or</i> Add an ACE to the End of an Existing, Numbered ACL	<pre>ProCurve(config)# access-list < 100-199 > < deny permit > < ip-options tcp/udp-options igmp-options icmp-options > [log]² [precedence < priority >] [tos < tos- setting >]</pre> <p>Note: Uses the same IP, TCP/UDP, IGMP, and ICMP options as shown above for "Create an Extended, Named ACL".</p>	10-74
Insert an ACE by Assigning a Sequence Number	<pre>ProCurve(config)# ip access-list extended < name-str 100-199 > ProCurve(config-ext-nacl)# 1-2147483647 < deny permit ></pre> <p><i>Uses the options shown above for "Create an Extended, Named ACL".</i></p>	10-92
Use Sequence Number To Delete an ACE	<pre>ProCurve(config)# ip access-list extended < name-str 100-199 > ProCurve(config-std-nacl)# no < 1-2147483647 ></pre>	10-94
Resequence the ACEs in an ACL	<pre>ProCurve(config)# ip access-list resequence < name-str 100-199 > < 1-2147483647 > < 1-2147483646 ></pre>	10-95

¹The mask can be in either dotted-decimal notation (such as 0.0.15.255) or CIDR notation (such as /20).

²The [log] function applies only to "deny" ACLs, and generates a message only when there is a "deny" match.

Table continues on the next page.

Action	Command(s)	Page
Enter or Remove a Remark	ProCurve(config)# ip access-list extended < name-str 100-199 >	10-96
	ProCurve(config-ext-nacl)# [remark < remark-str > no < 1 - 2147483647 > remark]	10-98
<p><i>For numbered, extended ACLs only, the following remark commands can be substituted for the above:</i></p> <p style="padding-left: 40px;">ProCurve(config)# access-list < 100 - 199 > remark < remark-str ></p> <p style="padding-left: 40px;">ProCurve(config)# [no] access-list < 100 - 199 > remark</p>		
Delete an Extended ACL	ProCurve(config)# no ip access-list extended < name-str 100-199 >	10-89
<p><i>For numbered, extended ACLs only, the following command can also be used:</i></p> <p style="padding-left: 40px;">ProCurve(config)# no access-list < 100 - 199 ></p>		

Standard ACLs use only source IP addresses for filtering criteria, extended ACLs use multiple filtering criteria. This enables you to more closely define your IP packet-filtering. Extended ACLs enable filtering on the following:

- Source and destination IP addresses (required), in one of the following options:
 - specific host IP
 - subnet or group of IP addresses
 - any IP address
- choice of any IP protocol
- optional packet-type criteria for IGMP, and ICMP traffic
- optional source and/or destination TCP or UDP port, with a further option for comparison operators and (for TCP) an option for establishing connections
- filtering for TCP traffic based on whether the subject traffic is initiating a connection (“established” option)
- optional IP precedence and ToS criteria

The switch allows up to 2048 ACLs in any combination of numeric and alphanumeric identifiers, and determines the total from the number of unique identifiers in the configuration. For example, configuring two ACLs results in an ACL total of two, even if neither is assigned to an interface. If you then assign a nonexistent ACL to an interface, the new ACL total is three, because the switch now has three unique ACL names in its configuration. (For more on ACL limits, refer to “Monitoring Shared Resources” on page 10-118.)

Configuring Named, Extended ACLs

For a match to occur with an ACE in an extended ACL, a packet must have the source and destination IP address criteria specified by the ACE, as well as any IP protocol-specific criteria included in the command.

Use the following general steps to create or add to a named, extended ACL:

1. Create and/or enter the context of a named, extended ACL.
2. Enter the first ACE in a new, extended ACL or append an ACE to the end of an existing, extended ACL.

This section describes the commands for performing these steps. For other ACL topics, refer to the following:

Topic	Page
configuring named, standard ACLs	10-53
configuring numbered, standard ACLs	10-56
configuring numbered, extended ACLs	10-74
applying or removing an ACL on an interface	10-81
deleting an ACL	10-89
editing an ACL	10-90
sequence numbering in ACLs	10-91
including remarks in an ACL	10-96
displaying ACL configuration data	10-100
creating or editing ACLs offline	10-108
enabling ACL "Deny" logging	10-113

Creating a Named, Extended ACL and/or Entering the “Named ACL” (nacl) Context. This command is a prerequisite to entering or editing ACEs in a named, extended ACL. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: ip access-list extended < name-str >

Places the CLI in the “Named ACL” (nacl) context specified by the < name-str > alphanumeric identifier. This enables entry of individual ACEs in the specified ACL. If the ACL does not already exist, this command creates it.

< name-str >: *Specifies an alphanumeric identifier for the ACL. Consists of an alphanumeric string of up to 64 case-sensitive characters. Including spaces in the string requires that you enclose the string in single or double quotes. For example: “Accounting ACL”. You can also use this command to access an existing, numbered ACL. Refer to “Using the CLI To Edit ACLs” on page 10-90*

```
ProCurve(config)# ip access-list extended Sample-List
ProCurve(config-ext-nacl)#
```

Figure 10-17. Example of Entering the Named ACL Context

Configure ACEs in a Named, Extended ACL and/or Enter the “Named ACL” (nacl) Context. Configuring ACEs is done after using the **ip access-list standard < name-str >** command described on page 10-63 to enter the “Named ACL” (**nacl**) context of an ACL. For an extended ACL syntax summary, refer to table 10-10 on page 10-60.

Syntax: < deny | permit > < ip | ip-protocol | ip-protocol-nbr >
(nacl
context) < any | host < SA > | SA / mask-length | SA < mask > >
< any | host < DA > | DA / mask-length | DA < mask > >
[precedence] [tos] [log]

*Appends an ACE to the end of the list of ACEs in the current ACL. In the default configuration, ACEs are automatically assigned consecutive sequence numbers in increments of 10 and can be renumbered using **resequence** (page 10-95).*

Note: *To insert a new ACE between two existing ACEs in an extended, named ACL, precede **deny** or **permit** with an appropriate sequence number along with the ACE keywords and variables you want. (Refer to “Inserting an ACE in an Existing ACL” on page 10-92.)*

For a match to occur, a packet must have the source and destination IP addressing criteria specified in the ACE, as well as:

- *the protocol-specific criteria configured in the ACE, including any included, optional elements (described later in this section)*
- *any (optional) precedence and/or ToS settings configured in the ACE*

< deny | permit >

*For named ACLs, these keywords are used in the “Named ACL” (**nacl**) context to specify whether the ACE denies or permits a packet matching the criteria in the ACE, as described below.*

< ip | ip-protocol | ip-protocol-nbr >

Used after **deny** or **permit** to specify the packet protocol type required for a match. An extended ACL must include one of the following:

- **ip** — any IP packet.
- **ip-protocol** — any one of the following IP protocol names:

ip-in-ip	ipv6-in-ip	gre	esp	ah
ospf	pim	vrrp	sctp	tcp*
udp*	icmp*	igmp*		
- **ip-protocol-nbr** — the IPv4 IP protocol number of an IP packet type, such as “8” for Exterior Gateway Protocol or 121 for Simple Message Protocol. (For a listing of IP protocol numbers and their corresponding protocol names, refer to the IANA “Protocol Number Assignment Services” at www.iana.com.) (Range: 0 - 255)

* For TCP, UDP, ICMP, and IGMP, additional criteria can be specified, as described on pages 10-68 through 10-72.

< any | host < SA > | SA < mask > | SA / mask-length

This is the first instance of IP addressing in an extended ACE. It follows the protocol specifier and defines the source IP address (SA) a packet must carry for a match with the ACE.

- **any** — Allows IP packets from any SA.
- **host < SA >** — Specifies only packets having a single address as the SA. Use this criterion when you want to match only the IP packets from a single SA.
- **SA < mask >** or **SA / mask-length** — Specifies packets received from an SA, where the SA is either a subnet or a group of IP addresses. The mask can be in either dotted-decimal format or CIDR format (number of significant bits). Refer to “Using CIDR Notation To Enter the ACL Mask” on page 10-50.

SA Mask Application: The mask is applied to the SA in the ACL to define which bits in a packet’s SA must exactly match the SA configured in the ACL and which bits need not match.

Example: 10.10.10.1/24 and 10.10.10.1 0.0.0.255 both define any IP address in the range of 10.10.10.(1 - 255).

Note: Specifying a group of contiguous IP addresses may require more than one ACE. For more on how masks operate in ACLs, refer to “How an ACE Uses a Mask To Screen Packets for Matches” on page 10-36.

< any | host < DA > | DA/mask-length | DA/ < mask >>

This is the second instance of IP addressing in an extended ACE. It follows the first (SA) instance, described earlier, and defines the destination IP address (DA) that a packet must carry in order to have a match with the ACE.

- **any** — Allows routed IP packets to any DA.
- **host < DA >** — Specifies only packets having **DA** as the destination address. Use this criterion when you want to match only the IP packets for a single DA.
- **DA/mask-length** or **DA < mask >** — Specifies packets intended for a destination address, where the address is either a subnet or a group of IP addresses. The mask format can be in either dotted-decimal format or CIDR format (number of significant bits). Refer to “Using CIDR Notation To Enter the ACL Mask” on page 10-50.

DA Mask Application: *The mask is applied to the DA in the ACL to define which bits in a packet’s DA must exactly match the DA configured in the ACL and which bits need not match. See also the above example and note.*

[precedence < 0 - 7 | precedence-name >]

This option can be used after the DA to cause the ACE to match packets with the specified IP precedence value. Values can be entered as the following IP precedence numbers or alphanumeric names:

0	or	routine
1	“	priority
2	“	immediate
3	“	flash
4	“	flash-override
5	“	critical
6	“	internet (for internetwork control)
7	“	network (for network control)

Note: *The precedence criteria described in this section are applied in addition to any other selection criteria configured in the same ACE.*

[tos < tos-setting >]

This option can be used after the DA to cause the ACE to match packets with the specified IP Type-of-Service (ToS) setting. ToS values can be entered as the following numeric settings or, in the case of 0, 2, 4, and 8, as alphanumeric names:

0	or	normal
2	“	max-reliability
4	“	max-throughput
6		
8	“	minimize-delay
10		
12		
14		

Note: *The ToS criteria in this section are applied in addition to any other criteria configured in the same ACE.*

[log]

This option can be used after the DA to generate an Event Log message if:

- *The action is **deny**. (Not applicable to **permit**.)*
- *There is a match.*
- *ACL logging is enabled. (Refer to “Enabling ACL Logging on the Switch” on page 10-115.)*

Options for TCP and UDP Traffic in Extended ACLs. An ACE designed to permit or deny TCP or UDP traffic can optionally include port number criteria for either the source or destination, or both. Use of TCP criteria also allows the **established** option for controlling TCP connection traffic. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: < deny | permit > < tcp | udp >
< SA > [comparison-operator < tcp/udp-src-port >]
< DA >
[comparison-operator < tcp-dest-port >] [established]
[comparison-operator < udp-dest-port >]

*In an extended ACL using either **tcp** or **udp** as the IP packet protocol type, you can optionally use TCP or UDP source and/or destination port numbers or ranges of numbers to further define the criteria for a match. For example:*

```
#deny tcp host 10.20.10.17 eq 23 host 10.20.10.155
  established
#permit tcp host 10.10.10.100 host 10.20.10.17
  eq telnet
#deny udp 10.30.10.1/24 host 10.20.10.17 range
  161 162
```

[comparison-operator < tcp/udp-src-port >]

To specify a TCP or UDP source port number in an ACE, (1) select a comparison operator from the following list and (2) enter the port number or a well-known port name.

Comparison Operators:

- **eq** < tcp/udp-port-nbr > — “Equal To”; to have a match with the ACE entry, the TCP or UDP source port number in a packet must be equal to < tcp/udp-port-nbr >.
- **gt** < tcp/udp-port-nbr > — “Greater Than”; to have a match with the ACE entry, the TCP or UDP source port number in a packet must be greater than < tcp/udp-port-nbr >.
- **lt** < tcp/udp-port-nbr > — “Less Than”; to have a match with the ACE entry, the TCP or UDP source port number in a packet must be less than < tcp/udp-port-nbr >.
- **neq** < tcp/udp-port-nbr > — “Not Equal”; to have a match with the ACE entry, the TCP or UDP source port number in a packet must not be equal to < tcp/udp-port-nbr >.
- **range** < start-port-nbr > < end-port-nbr > — For a match with the ACE entry, the TCP or UDP source-port number in a packet must be in the range < start-port-nbr > < end-port-nbr >.

Port Number or Well-Known Port Name:

Use the TCP or UDP port number required by your application. The switch also accepts these well-known TCP or UDP port names as an alternative to their port numbers:

- **TCP:** bgp, dns, ftp, http, imap4, ldap, nntp, pop2, pop3, smtp, ssl, telnet
- **UDP:** bootpc, bootps, dns, ntp, radius, radius-old, rip, snmp, snmp-trap, tftp

To list the above names, press the **[Shift] [?]** key combination after entering an operator. For a comprehensive listing of port numbers, visit www.iana.org/assignments/port-numbers.

[comparison-operator < tcp-dest-port >] [established]

[comparison-operator < udp-dest-port >]

This option, if used, is entered immediately after the < DA > entry. To specify a TCP or UDP port number, (1) select a comparison operator and (2) enter the port number or a well-known port name.

Comparison Operators and Well-Known Port Names —

These are the same as are used with the TCP/UDP source-port options, and are listed earlier in this command description.

[established] — *This option applies only where TCP is the configured IP protocol type. It blocks the synchronizing packet associated with establishing a TCP connection in one direction on a VLAN while allowing all other IP traffic for the same type of connection in the opposite direction. For example, a Telnet connect requires TCP traffic to move both ways between a host and the target device. Simply applying a Deny to inbound Telnet traffic on a VLAN would prevent Telnet sessions in either direction because responses to outbound requests would be blocked. However, by using the **established** option, inbound Telnet traffic arriving in response to outbound Telnet requests would be permitted, but inbound Telnet traffic trying to establish a connection would be denied.*

Options for ICMP Traffic in Extended ACLs. This option is useful where it is necessary to permit some types of ICMP traffic and deny other types, instead of simply permitting or denying all types of ICMP traffic. That is, an ACE designed to permit or deny ICMP traffic can optionally include an ICMP type and code value to permit or deny an individual type of ICMP packet while not addressing other ICMP traffic types in the same ACE. As an optional alternative, the ACE can include the name of an ICMP packet type. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: < deny | permit > icmp < SA > < DA > [icmp-type [icmp-code]
< deny | permit > icmp < SA > < DA > [icmp-type-name]

[] []

*In an extended ACL using **icmp** as the packet protocol type (see above), you can optionally specify an individual ICMP packet type or packet type/code pair to further define the criteria for a match. This option, if used, is entered immediately after the destination IP address (DA) entry. The following example shows two ACEs entered in a Named ACL context:*

```
#permit icmp any any host-unknown  
#permit icmp any any 3 7
```

[icmp-type [icmp-code]]

This option identifies an individual ICMP packet type as criteria for permitting or denying that type of ICMP traffic in an ACE.

- **icmp-type** — This value is in the range of 0 - 255 and corresponds to an ICMP packet type.
- **icmp-code** — This value is in the range of 0 - 255 and corresponds to an ICMP code for an ICMP packet type.

For more information on ICMP type names, visit the Internet Assigned Numbers Authority (IANA) website at www.iana.com, click on “Protocol Number Assignment Services”, and then go to the selections under “Internet Control Message Protocol (ICMP) Parameters”.

[*icmp-type-name*]

These name options are an alternative to the [icmp-type [icmp-code]] methodology described above. For more information, visit the IANA website cited above.

administratively-prohibited	net-tos-unreachable
alternate-address	net-unreachable
conversion-error	network-unknown
dod-host-prohibited	no-room-for-option
dod-net-prohibited	option-missing
echo	packet-too-big
echo-reply	parameter-problem
general-parameter-problem	port-unreachable
host-isolated	precedence-unreachable
host-precedence-unreachable	protocol-unreachable
host-redirect	reassembly-timeout
host-tos-redirect	redirect
host-tos-unreachable	router-advertisement
host-unknown	router-solicitation
host-unreachable	source-quench
information-reply	source-route-failed
information-request	time-exceeded
mask-reply	timestamp-reply
mask-request	timestamp-request
mobile-redirect	traceroute
net-redirect	ttl-exceeded
net-tos-redirect	unreachable

Option for IGMP in Extended ACLs. This option is useful where it is necessary to permit some types of IGMP traffic and deny other types instead of simply permitting or denying all types of IGMP traffic. That is, an ACE designed to permit or deny IGMP traffic can optionally include an IGMP packet type to permit or deny an individual type of IGMP packet while not addressing other IGMP traffic types in the same ACE. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: < permit | deny > igmp < SA > < DA > [igmp-type]

*In an extended ACL using **igmp** as the packet protocol type, you can optionally specify an individual IGMP packet type to further define the criteria for a match. This option, if used, is entered immediately after the destination IP addressing entry. The following example shows an IGMP ACE entered in the Named ACL context:*

```
ProCurve(config-ext-nacl)# permit igmp any  
any host-query
```

[igmp-type]

The complete list of IGMP packet-type options includes:

dvmrp	trace	mtrace-request
host-query	v2-host-report	v3-host-report
host-report	v2-host-leave	
pim	mtrace-reply	

For more information on IGMP packet types, visit the Internet Assigned Numbers Authority (IANA) website at www.iana.com, click on “Protocol Number Assignment Services”, and then go to the selections under “Internet Group Management Protocol (IGMP) Type Numbers”.

Example of a Named, Extended ACL. Suppose that you want to implement these policies on a switch configured for IP routing and membership in VLANs 10, 20, and 30:

- A. Permit Telnet traffic from 10.10.10.44 to 10.10.20.78, deny all other IP traffic from network 10.10.10.0 (VLAN 10) to 10.10.20.0 (VLAN 20), and permit all other IP traffic from any source to any destination. (See “A” in figure 10-18, below.)
- B. Permit FTP traffic from IP address 10.10.20.100 (on VLAN 20) to 10.10.30.55 (on VLAN 30). Deny FTP traffic from other hosts on network 10.10.20.0 to any destination, but permit all other IP traffic.

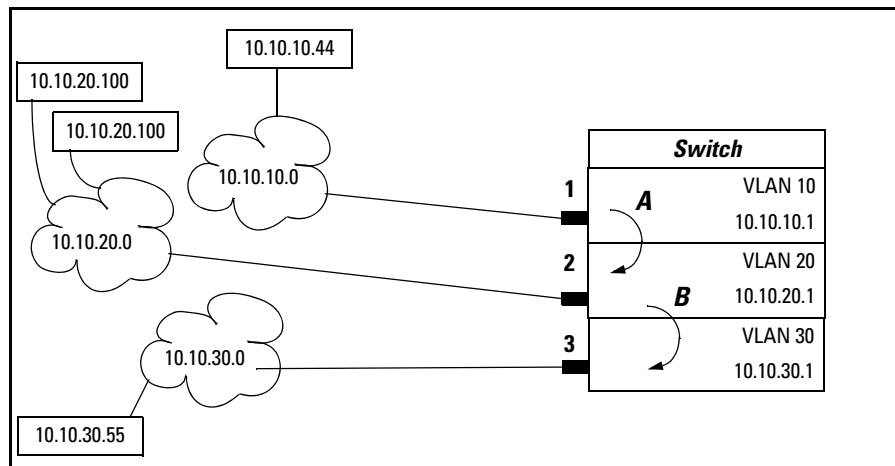


Figure 10-18. Example of an Extended ACL

```
A (Refer to figure 10-18 on page  
ProCurve(config)# ip access-list extended Extended-List-01  
| ProCurve(config-ext-nacl)# permit tcp host 10.10.10.44 host  
| 10.10.20.78 eq telnet  
| ProCurve(config-ext-nacl)# deny ip 10.10.10.1/24 10.10.20.1/24  
| ProCurve(config-ext-nacl)# permit ip any any  
| ProCurve(config-ext-nacl)# exit  
| ProCurve(config)# vlan 10 ip access-group Extended-List in  
B (Refer to figure 10-18 on page  
ProCurve(config)# ip access-list extended Extended-List-02  
| ProCurve(config-ext-nacl)# permit tcp host 10.10.20.100 host  
| 10.10.30.55 eq ftp  
| ProCurve(config-ext-nacl)# deny tcp 10.10.20.1/24 any eq ftp log  
| ProCurve(config-ext-nacl)# permit ip any any  
| ProCurve(config-ext-nacl)# exit  
| ProCurve(config)# vlan 20 ip access-group Extended-List-02 in
```

Figure 10-19. Example of Configuration Commands for Extended ACLs

Configuring Numbered, Extended ACLs

This section describes the commands for performing the following in a numbered, extended ACL:

- creating the ACL by entering the first ACE in the list
- appending a new ACE to the end of an existing ACL

For other ACL topics, refer to the following:

Topic	Page
configuring named, standard ACLs	10-53
configuring numbered, standard ACLs	10-56
configuring named, extended ACLs	10-62
applying or removing an ACL on an interface	10-81
deleting an ACL	10-89
editing an ACL	10-90
sequence numbering in ACLs	10-91
including remarks in an ACL	10-96
displaying ACL configuration data	10-100
creating or editing ACLs offline	10-108
enabling ACL "Deny" logging	10-113

Creating or Adding to an Extended, Numbered ACL. This command is an alternative to using **ip access-list extended < name-str >** and does not use the Named ACL (**nacl**) context. (For an extended ACL syntax summary, refer to table 10-10 on page 10-60.)

Syntax: access-list < 100-199 > < deny | permit > < ip | ip-protocol | ip-protocol-nbr >
< any | host < SA > | SA/mask-length | SA < mask >>
< any | host < DA > | DA/mask-length | DA < mask >>
[precedence < 0 - 7 | precedence-name >]
[tos < tos-bit-setting >]
[log]

*If the ACL does not already exist, this command creates the specified ACL and its first ACE. If the ACL already exists, the new ACE is appended to the end of the configured list of explicit ACEs. In the default configuration, the ACEs in an ACL will automatically be assigned consecutive sequence numbers in increments of 10 and can be renumbered with **resequence** (page 10-95).*

Note: To insert a new ACE between two existing ACEs in an extended, numbered ACL:

- a. Use **ip access list extended < 100 - 199 >** to open the ACL as a named ACL.
- b. Enter the desired sequence number along with the ACE statement you want.

(Refer to the “Numbered ACLs” list item on page 10-49.)

For a match to occur, a packet must have the source and destination IP addressing criteria specified in the ACE, as well as:

- *the protocol-specific criteria configured in the ACE, including any included, optional elements (described later in this section)*
- *any (optional) precedence and/or ToS settings configured in the ACE*

< 100-199 >

Specifies the ACL ID number. The switch interprets a numeric ACL with a value in this range as an extended ACL.

< deny | permit >

*Specifies whether to deny (**drop**) or permit (forward) a packet that matches the criteria specified in the ACE, as described below.*

< ip | ip-protocol | ip-protocol-nbr >

Specifies the packet protocol type required for a match. An extended ACL must include one of the following:

- **ip** — any IP packet.
 - **ip-protocol** — any one of the following IP protocol names:

ip-in-ip	ipv6-in-ip	gre	esp	ah
ospf	pim	vrrp	sctp	tcp*
udp*	icmp*	igmp*		
 - **ip-protocol-nbr** — the IPv4 IP protocol number of an IP packet type, such as “8” for Exterior Gateway Protocol or 121 for Simple Message Protocol. (For a listing of IP protocol numbers and their corresponding protocol names, refer to the IANA “Protocol Number Assignment Services” at www.iana.com.) (Range: 0 - 255)
- * For TCP, UDP, ICMP, and IGMP, additional criteria can be specified, as described later in this section.

< any | host < SA > | SA/mask-length | SA < mask >>

In an extended ACL, this parameter defines the source IP address (SA) that a packet must carry in order to have a match with the ACE.

- **any** — Specifies all inbound IP packets.
- **host < SA >** — Specifies only inbound packets from a single IP address. Use this option when you want to match only the IP packets from one source IP address.
- **SA/mask-length** or **SA < mask >** — Specifies packets received from an SA, where the SA is either a subnet or a group of IP addresses. The mask can be in either dotted-decimal format or CIDR format with the number of significant bits. Refer to “Using CIDR Notation To Enter the ACL Mask” on page 10-50.

SA Mask Application: *The mask is applied to the SA in the ACL to define which bits in a packet's source SA must exactly match the IP address configured in the ACL and which bits need not match.*

Example: *10.10.10.1/24 and 10.10.10.1 0.0.0.255 both define any IP address in the range of 10.10.10.(1-255).*

Note: *Specifying a group of contiguous IP addresses may require more than one ACE. For more on how masks operate in ACLs, refer to "How an ACE Uses a Mask To Screen Packets for Matches" on page 10-36.*

< any | host < DA > | DA/mask-length >

This is the second instance of IP addressing in an extended ACE. It follows the first (SA) instance, described earlier, and defines the destination IP address (DA) that a packet must carry in order to have a match with the ACE. The options are the same as shown for < SA >.

- **any** — *Allows routed IP packets to any DA.*
- **host < DA >** — *Specifies only packets having DA as the destination IP address. Use this criterion when you want to match only the IP packets for a single DA.*
- **DA/mask-length** or **DA < mask >** — *Specifies packets intended for a destination address, where the address is either a subnet or a group of IP addresses. The mask format can be in either dotted-decimal format or CIDR format (number of significant bits). Refer to "Using CIDR Notation To Enter the ACL Mask" on page 10-50.*

DA Mask Application: *The mask is applied to the DA in the ACL to define which bits in a packet's DA must exactly match the DA configured in the ACL and which bits need not match. See also the above example and note.*

[precedence < 0 - 7 | precedence-name >]

This option causes the ACE to match packets with the specified IP precedence value. Values can be entered as the following IP precedence numbers or alphanumeric names:

0	or	routine
1	“	priority
2	“	immediate
3	“	flash
4	“	flash-override
5	“	critical
6	“	internet (for internetwork control)
7	“	network (for network control)

Note: *The precedence criteria described in this section are applied in addition to any other selection criteria configured in the same ACE.*

[tos]

This option can be used after the DA to cause the ACE to match packets with the specified IP Type-of-Service (ToS) setting. ToS values can be entered as the following numeric settings or, in the case of 0, 2, 4, and 8, as alphanumeric names:

0	or	normal
2	“	max-reliability
4	“	max-throughput
6		
8	“	minimize-delay
10		
12		
14		

Note: *The ToS criteria in this section are applied in addition to any other criteria configured in the same ACE.*

[log]

Optional; generates an Event Log message if:

- *The action is **deny**. (This option is not configurable for Permit.)*
- *There is a match.*
- *ACL logging is enabled on the switch. (Refer to “Enabling ACL Logging on the Switch” on page 10-115)*

Additional Options for TCP and UDP Traffic. An ACE designed to permit or deny TCP or UDP traffic can optionally include port number criteria for either the source or destination, or both. Use of TCP criteria also allows the **established** option for controlling TCP connection traffic. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: access-list < 100 - 199 > < deny | permit > < tcp | udp >
 < SA > [comparison-operator < tcp/udp-src-port >]

 < DA > [comparison-operator < tcp-dest-port >] [established]
 < DA > [comparison-operator < udp-dest-port >]

This source-port and destination-port TCP/UDP criteria is identical to the criteria described for TCP/UDP use in named, extended ACLs, beginning on page 10-68.

Additional Options for ICMP Traffic. This option is useful where it is necessary to permit some types of ICMP traffic and deny other types, instead of simply permitting or denying all types of ICMP traffic. That is, an ACE designed to permit or deny ICMP traffic can optionally include an ICMP type and code value to permit or deny an individual type of ICMP packet while not addressing other ICMP traffic types in the same ACE. As an optional alternative, the ACE can include the name of an ICMP packet type. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: access-list < 100 - 199 > < deny | permit > icmp < SA > < DA >
 [[icmp-type [icmp-code]] | [icmp-type-name]]

The ICMP “type” and “code” criteria are identical to the criteria described for ICMP in named, extended ACLs, beginning on page 10-70.

Additional Option for IGMP. This option is useful where it is necessary to permit some types of IGMP traffic and deny other types, instead of simply permitting or denying all types of IGMP traffic. That is, an ACE designed to permit or deny IGMP traffic can optionally include an IGMP packet type to permit or deny an individual type of IGMP packet while not addressing other IGMP traffic types in the same ACE. (For a summary of the extended ACL syntax options, refer to table 10-10 on page 10-60.)

Syntax: access-list < 100 - 199 >
 < deny | permit > igmp < src-ip > < dest-ip > [igmp-type]

The IGMP “type” criteria is identical to the criteria described for IGMP in named, extended ACLs, beginning on page 10-72.

Adding or Removing an ACL Assignment On an Interface

Filtering Routed IP Traffic

For a given VLAN interface on a switch configured for routing, you can assign an ACL as a RACL to filter inbound IP traffic and another ACL as a RACL to filter outbound IP traffic. You can also assign one ACL for both inbound and outbound RACLs, and for assignment to multiple VLANs. For limits and operating rules, refer to “ACL Configuration and Operating Rules” on page 10-33.

Syntax: [no] vlan < vid > ip access-group < identifier > < in | out >
where: < identifier > = either a ACL name or an ACL ID number.

Assigns an ACL to a VLAN as an RACL to filter routed IP traffic entering or leaving the switch on that VLAN. You can use either the global configuration level or the VLAN context level to assign or remove an RACL.

Note: *The switch allows you to assign a nonexistent ACL name or number to a VLAN. In this case, if you subsequently configure an ACL with that name or number, it automatically becomes active on the assigned VLAN. Also, if you delete an assigned ACL from the switch without subsequently using the “no” form of this command to remove the assignment to a VLAN, the ACL assignment remains and will automatically activate any new ACL you create with the same identifier (name or number).*

Access Control Lists (ACLs)

Adding or Removing an ACL Assignment On an Interface

ProCurve(config)# vlan 20 ip access-group My-List in	←	Enables an RACL from the Global Configuration Level
ProCurve(config)# vlan 20		
ProCurve(vlan-20)# ip access-group 155 out	←	Enables an RACL from a VLAN Context.
ProCurve(vlan-20)# exit		
ProCurve(config)# no vlan 20 ip access-group My-List in	←	Disables an RACL from the Global Configuration Level
ProCurve(config)# vlan 20		
ProCurve(vlan-20)# no ip access-group 155 out	←	Disabling an RACL from a VLAN Context.
ProCurve(vlan-20)# exit		

Figure 10-20. Methods for Enabling and Disabling RACLs

Filtering IP Traffic Inbound on a VLAN

For a given VLAN interface, you can assign an ACL as a VACL to filter any IP traffic entering the switch on that VLAN. You can also use the same ACL for assignment to multiple VLANs. For limits and operating rules, refer to “ACL Configuration and Operating Rules” on page 10-33.

Syntax: [no] vlan < vid > ip access-group < identifier > vlan
where: < identifier > = either a ACL name or an ACL ID number.

Assigns an ACL as a VACL to a VLAN to filter any IP traffic entering the switch on that VLAN. You can use either the global configuration level or the VLAN context level to assign or remove a VACL.

Note: *The switch allows you to assign a nonexistent ACL name or number to a VLAN. In this case, if you subsequently configure an ACL with that name or number, it automatically becomes active on the assigned VLAN. Also, if you delete an assigned ACL from the switch without subsequently using the “no” form of this command to remove the assignment to a VLAN, the ACL assignment remains and will automatically activate any new ACL you create with the same identifier (name or number).*

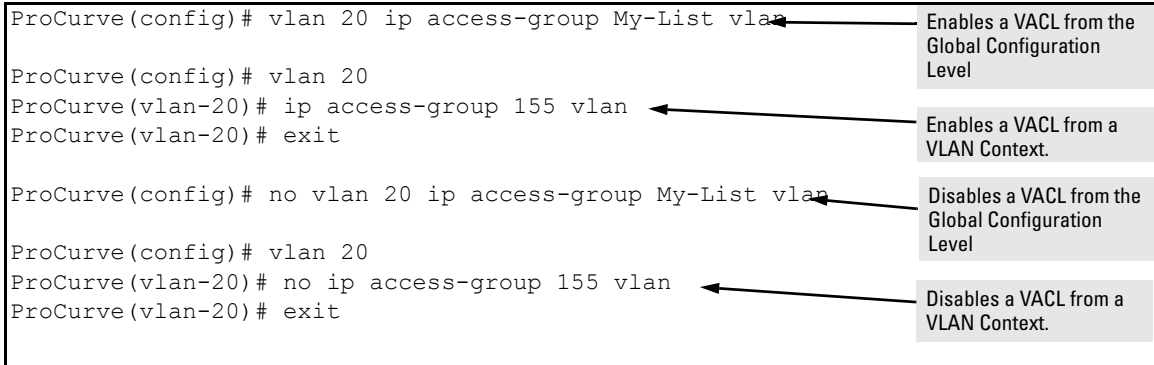


Figure 10-21. Methods for Enabling and Disabling VACLs

Filtering Inbound IP Traffic Per Port

For a given port, port list, or static port trunk, you can assign an ACL as a static port ACL to filter any IP traffic entering the switch on that interface. You can also use the same ACL for assignment to multiple interfaces. For limits and operating rules, refer to “ACL Configuration and Operating Rules” on page 10-33.

Syntax: [no] interface < port-list | Trkx > ip access-group < identifier > in
where: < identifier > = either a ACL name or an ACL ID number.

Assigns an ACL as a static port ACL to a port, port list, or static trunk to filter any IP traffic entering the switch on that interface. You can use either the global configuration level or the interface context level to assign or remove a static port ACL.

Note: *The switch allows you to assign a nonexistent ACL name or number to an interface. In this case, if you subsequently configure an ACL with that name or number, it automatically becomes active on the assigned interface. Also, if you delete an assigned ACL from the switch without subsequently using the “no” form of this command to remove the assignment to an interface, the ACL assignment remains and will automatically activate any new ACL you create with the same identifier (name or number).*

ProCurve(config)# interface b10 ip access-group My-List in	← Enables a static port ACL from the Global Configuration level.
ProCurve(config)# interface b10	
ProCurve(eth-b10)# ip access-group 155 in	← Enables a static port ACL from a port context.
ProCurve(eth-b10)# exit	
ProCurve(config)# no interface b10 ip access-group My-List in	← Disables a static port ACL from the Global Configuration level.
ProCurve(config)# interface b10	
ProCurve(eth-b10)# no ip access-group 155 in	← Uses a VLAN context to disable a static port ACL.
ProCurve(eth-b10)# exit	

Figure 10-22. Methods for Enabling and Disabling ACLs

Classifier-Based Rate-Limiting with RL-PACLs

Classifier-Based Rate Limiting (also known as Rate-Limit Port ACLs or RL-PACLs) allows you to create an ACL and apply it on a per-port basis to rate-limit network traffic. When packets entering the port match a PERMIT statement in the ACL, they are rate-limited at the rate you have configured. If the packet matches a DENY statement, it is not rate-limited. There is an implicit DENY ALL at the end of the ACL, which means packets with no match are not rate-limited.

RL-PACLs use one meter per port and that meter is shared among all PERMIT statements for the RL-PACL on that port. The meter is not shared with other ports. For a 24-port interface module, 24 meters potentially can be used. (There are up to 256 meters available per module.)

Operating features include:

- Each port individually enforces the rate limit specified for that port.
- Only one rate-limiting ACL is allowed per port. Rate limits can be applied to a range of ports.
- Rate limits set on one port do not affect the traffic on any other port
- If you want to rate-limit some classes of traffic and drop others, the RL-PACL must be used in combination with another CLI-configured PACL; this feature does not provide that functionality.
- The rate-limiting configuration information is stored in the config file
- Rate-limiting ACLs cannot be configured on a port that is part of a trunk.

Caution

All rate-limit values are in Kbps (1000 bits per second); some other rate-limiting features use bps (bits per second). Be careful to enter the rates correctly.

CLI Command for Rate Limiting

To set a per-port rate-limit, use this command:

Syntax: [no] interface <port-list> rate-limit ip access-group <name> in kbps <rate>

Applies the access-group name specified to the ports selected in <port-list>. All packets that match a permit statement in access-group <name> are rate-limited and all packets that match a deny statement in access-group <name> are not rate-limited.

in Kbps <rate>: *The range is 1-10,000,000 kilobits per second.*

```
ProCurve(config)# interface 4-8 rate-limit ip access-group Group_A in kbps 10000
```

Figure 10-23. Example of Rate-Limiting RL-PACL on a Range of Ports

In the example in Figure 10-23 the ACL named Group_A is applied to ports 4 through 8 with a limit of 10000 kbps (10 Mbps). When the packets enter the configured ports, all packets that match a PERMIT statement in Group_A are rate-limited and all packets that match a DENY statement in Group_A are not rate-limited. The ACL does not have to be defined before being applied as an RL-PACL.

Note

RL-PACLs and other ACL types (normal PACLs, VACLs, RACLs, etc.) operate on packets at the same time. If any ACL indicates the packet should be dropped, the packet is dropped. However, the packet will still count towards RL-PACL meter usage.

To turn off the rate limiting use the **no** version of the command:

```
ProCurve(config)# no interface 4-8 rate-limit ip  
access-group
```

Viewing the RL-PACL Information

The **show rate-limit** CLI command displays information about the rate-limiting configured on each port for each group.

Syntax: show rate-limit ip access-group <[ethernet] port-list>

```
ProCurve(config)# show rate-limit ip access-group

Inbound access-group rate-limits:

Port  kbps      Access-group name
-----
4      10000      Group_A
5      10000      Group_A
6      10000      Group_A
7      10000      Group_A
8      10000      Group_A
```

Ports with no RL-PACL applied are not displayed in the output.

Figure 10-24. Example of show rate-limit Command for RL-PACLs

Show Access Lists by Port

Figure 10-25 shows the rate limit information for a specific port.

```
ProCurve(config)# show access-list ports 4-6

Access Lists for Port 4

RateLimit : Group_A
Type      : None

Access Lists for Port 5

RateLimit : Group_A
Type      : None

Access Lists for Port 6

RateLimit : Group_A
Type      : None
```

Figure 10-25. Access-List Information for Selected Ports

Troubleshooting RL-PACLs

The following situations may occur when using RL-PACLs.

Problem	Resolution
You try to apply an RL-PACL to a port, but are informed that there are insufficient resources.	Hardware resources have been consumed by some combination of RL-PACLs, other ACLS, other QoS or rate-limiting features or other features on the switch. Enter the commands show access-list resources or show qos resources to see what features are using resources.
The switch is unexpectedly dropping traffic entering a port and you suspect the RL-PACL is the issue.	The command show access-list ports <port-list> shows which ACLs are applied to a port, including regular ACLS and RL-PACLs. If these are not causing the problem, enter the command show access-list vlan <vlan-id> to check VLANs. Additionally, check the configured rate-limit using the show rate-limit command.
The switch is not rate-limiting traffic even though an RL-PACL is configured.	Verify that the RL-PACL is configured correctly by entering the show rate-limit ip access-group <portnum> command. Check that the ACL name and rate (in kbps) are correctly configured for the port. Traffic is only dropped when the rate limit for packets matching permit statements is exceeded. For example, if an RL-PACL is configured with a rate-limit of 100 kbps and 100 kbps of traffic matches permit statements and 100 kbps of traffic matches deny statements, all 200 kbps of traffic is allowed to pass through the port.

Deleting an ACL

Syntax: no ip access-list standard < name-str | 1-99 >

no ip access-list extended < name-str | 100-199 >

no access-list < 1 - 99 | 100 - 199 >

Removes the specified ACL from the switch's running-config file.

Note: *Deleting an ACL does not delete any assignment of that ACL's identifier on a specific interface. Creating a new ACL using an identifier that is already configured on an interface causes the switch to automatically activate that ACL. If you need to remove an ACL identifier assignment on an interface, refer to "Adding or Removing an ACL Assignment On an Interface" on page 10-81*

Editing an Existing ACL

The CLI provides the capability for editing in the switch by using sequence numbers to insert or delete individual ACEs. An offline method is also available. This section describes using the CLI for editing ACLs. To use the offline method for editing ACLs, refer to “Creating or Editing ACLs Offline” on page 10-108.

Using the CLI To Edit ACLs

You can use the CLI to delete individual ACEs from anywhere in an ACL, append new ACEs to the end of an ACL, and insert new ACEs anywhere within an ACL.

General Editing Rules

- **Named ACLs:**
 - When you enter a new ACE in a named ACL without specifying a sequence number, the switch inserts the ACE as the last entry in the ACL.
 - When you enter a new ACE in a named ACL and include a sequence number, the switch inserts the ACE according to the position of the sequence number in the current list of ACEs.
- **Numbered ACLs:** When using the **access-list < 1 - 99 | 100 - 199 >** command to create or add ACEs to a numbered ACL, each new ACE you enter is added to the end of the current list. (This command does not offer a **< seq-# >** option for including a sequence number to enable inserting an ACE at other points in the list.) Note, however, that once a numbered list has been created, you have the option of accessing it in the same way as a named list by using the **ip access-list < standard | extended >** command. This enables you to edit a numbered list in the same way that you would edit a named list. (See the next item in this list.)
- You can delete any ACE from any ACL (named or numbered) by using the **ip access-list** command to enter the ACLs context, and then using the **no < seq-# >** command (page 10-94).

- Deleting the last ACE from an ACL leaves the ACL in memory. In this case, the ACL is “empty” and cannot perform any filtering tasks. (In any ACL the Implicit Deny does not apply unless the ACL includes at least one explicit ACE.)

Sequence Numbering in ACLs

The ACEs in any ACL are sequentially numbered. In the default state, the sequence number of the first ACE in a list is “10” and subsequent ACEs are numbered in increments of 10. For example, the following **show run** output lists three ACEs with default numbering in a list named “My-List”:

```
ip access-list standard "My-List"  
 10 permit 10.10.10.25 0.0.0.0  
 20 permit 10.20.10.117 0.0.0.0  
 30 deny 10.20.10.1 0.0.0.255  
exit
```

Figure 10-26. Example of the Default Sequential Numbering for ACEs

You can add an ACE to the end of a named or numbered ACL by using either **access-list** for numbered ACLs or **ip access-list** for named ACLs:

```
ProCurve(config)# access-list 2 permit any  
  
ProCurve(Config)# ip access-list standard My-list  
ProCurve(Config-ext-nacl)# permit ip any host 10.10.10.125
```

← Appends an ACE to the end of a standard, numbered ACL.

↗ Enters the context of an extended ACL and appends an ACE to the end of the list.

Figure 10-27. Examples of Adding an ACE to the end of Numbered or Named ACLs

For example, to append a fourth ACE to the end of the ACL in figure 10-26:

```
ProCurve(config)# ip access-list standard My-List
ProCurve(config-std-nacl)# permit any
ProCurve(config-std-nacl)# show run
.
.
.
ip access-list standard "My-List"
 10 permit 10.10.10.25 0.0.0.0
 20 permit 10.20.10.117 0.0.0.0
 30 deny 10.20.10.1 0.0.0.255
 40 permit 0.0.0.0 255.255.255.255
exit
```

Figure 10-28. Example of Appending an ACE to an Existing List

Note

When using the **access-list < 1 - 99 | 100 - 199 > < permit | deny > < SA >** command to create an ACE for a numbered ACL, the ACE is always added to the end of the current list and given the appropriate sequence number. However, once a numbered list has been created, you can use the **ip access-list** command to open it as a named ACL and specify a nondefault sequence number, as described in the next section.

Inserting an ACE in an Existing ACL

This action uses a sequence number to specify where to insert a new ACE into an existing sequence of ACLs.

Syntax: ip access-list < standard | extended > < name-str | 1 - 99 | 100 - 199 >

```
<1-2147483647> < permit | deny > < standard-acl-ip-criteria > [ log ]
<1-2147483647> < permit | deny > < extended-acl-ip-criteria > [ options ]
```

The first command enters the “Named-ACL” context for the specified ACL. The remaining two commands insert a new ACE in a standard or extended ACL, respectively. (For details on these criteria and options, refer to table 10-1, “Command Summary for Standard ACLs” —page 10-6, and table 10-2, “Command Summary for Extended ACLs” —page 10-8.)

To insert a new ACE between existing ACEs in a list:

1. Use **ip access-list** to enter the “Named-ACL” (**nacl**) context of the ACE. *This applies regardless of whether the ACE was originally created as a numbered ACL or a named ACL.*

2. Begin the ACE command with a sequence number that identifies the position you want the ACE to occupy. (The sequence number range is 1-2147483647.)
3. Complete the ACE with the command syntax appropriate for the type of ACL you are editing.

For example, inserting a new ACE between the ACEs numbered 10 and 20 in figure 10-28 requires a sequence number in the range of 11-19 for the new ACE.

```
ProCurve(config)# ip access-list standard My-List
ProCurve(config-std-nacl)# 15 deny 10.10.10.1/24
ProCurve(config-std-nacl)# show run
.
.
.
ip access-list standard "My-List"
 10 permit 10.10.10.25 0.0.0.0
 15 deny 10.10.10.1 0.0.0.255
 20 permit 10.20.10.117 0.0.0.0
 30 deny 10.20.10.1 0.0.0.255
 40 permit 0.0.0.0 255.255.255.255
exit
```

Figure 10-29. Example of Inserting an ACE in an Existing ACL

In the following example, the first two ACEs entered become lines 10 and 20 in the list. The third ACE entered is configured with a sequence number of 15 and is inserted between lines 10 and 20.

```
ProCurve(config)# ip access-list standard List-01
ProCurve(config-std-nacl)# permit 10.10.10.1/24
ProCurve(config-std-nacl)# deny 10.10.1.1/16
ProCurve(config-std-nacl)# 15 permit 10.10.20.1/24
ProCurve(config-std-nacl)# show run

Running configuration:
. . .
ip access-list standard "List-01"
 10 permit 10.10.10.1 0.0.0.255
 15 permit 10.10.20.1 0.0.0.255
 20 deny 10.10.1.1 0.0.255.255
exit
```

Figure 10-30. Example of Inserting an ACE into an Existing Sequence

Deleting an ACE from an Existing ACL

This action uses ACL sequence numbers to delete ACEs from an ACL.

Syntax: ip access-list < standard | extended > < name-str | 1 - 99 | 100 - 199 >
no < seq-# >

The first command enters the “Named-ACL” context for the specified ACL. The no command deletes the ACE corresponding to the sequence number entered. (Range: 1 - 2147483647)

1. To find the sequence number of the ACE you want to delete, use **show run** or **show access-list < name-str | 1 - 99 | 100-199 >** to view the ACL.
2. Use **ip access-list** to enter the “Named-ACL” (**nacl**) context of the ACE. *This applies regardless of whether the ACE was originally created as a numbered ACL or a named ACL.*
3. In the “Named-ACL” context, type **no** and enter the sequence number of the ACE you want to delete.

Figure 10-31 illustrates the process for deleting an ACE from a list:

```
ProCurve(config)# show run
. . .
ACL Before Deleting an ACE
ip access-list standard "My-List"
 10 permit 10.10.10.25 0.0.0.0
 15 deny 10.10.10.1 0.0.0.255
 20 permit 10.20.10.117 0.0.0.0
 30 deny 10.20.10.1 0.0.0.255
 40 permit 0.0.0.0 255.255.255.255
  exit
ProCurve(config)# ip access-list standard My-List
ProCurve(config-std-nacl)# no 20
ProCurve(config-std-nacl)# show run
. . .
ACL After Deleting the ACE at Line 20
ip access-list standard "My-List"
 10 permit 10.10.10.25 0.0.0.0
 15 deny 10.10.10.1 0.0.0.255
 30 deny 10.20.10.1 0.0.0.255
 40 permit 0.0.0.0 255.255.255.255
  exit
```

This command enters the “Named-ACL” (nacl) context for “My-List”.

This command deletes the ACE at line 20.

The ACE at line 20 has been removed.

Figure 10-31. Example of Deleting an ACE from Any ACL

Resequencing the ACEs in an ACL

This action reconfigures the starting sequence number for ACEs in an ACL, and resets the numeric interval between sequence numbers for ACEs configured in the ACL.

Syntax: `ip access-list resequence < name-str | 1 - 99 | 100 - 199 >
< starting-seq-# > < interval >`

Resets the sequence numbers for all ACEs in the ACL.

< starting-seq-# > : Specifies the sequence number for the first ACE in the list. (Default: 10; Range: 1 - 2147483647)

< interval > : Specifies the interval between sequence numbers for the ACEs in the list. (Default: 10; Range: 1 - 2147483647)

1. To view the current sequence numbering in an ACE, use **show run** or **show access-list < name-str | 1 - 99 | 100-199 >**.
2. Use the command syntax (above) to change the sequence numbering.

This example resequences the “My-List” ACL at the bottom of figure 10-31 so that the list begins with line 100 and uses a sequence interval of 100.

```
ProCurve(config)# show run
. . .
ip access-list standard "My-List"
 10 permit 10.10.10.25 0.0.0.0
 15 deny 10.10.10.1 0.0.0.255
 30 deny 10.20.10.1 0.0.0.255
 40 permit 0.0.0.0 255.255.255.255
exit
. . .
ProCurve(config)# ip access-list resequence My-List 100 100
ProCurve(config)# show run
. . .
ip access-list standard "My-List"
 100 permit 10.10.10.25 0.0.0.0
 200 deny 10.10.10.1 0.0.0.255
 300 deny 10.20.10.1 0.0.0.255
 400 permit 0.0.0.0 255.255.255.255
exit
```

Figure 10-32. Example of Viewing and Resequencing an ACL

Attaching a Remark to an ACE

A remark is numbered in the same way as an ACE, and uses the same sequence number as the ACE to which it refers. This operation requires that the remark for a given ACE be entered prior to entering the ACE itself.

Syntax: access-list < 1 - 99 | 100 - 199 > remark < remark-str >

This syntax appends a remark to the end of a numbered ACL and automatically assigns a sequence number to the remark. The next command entry should be the ACE to which the remark belongs. (The new ACE will automatically be numbered with the same sequence number as was used for the preceding remark.)

Syntax: ip access-list < standard | extended > < name-str | 1-99 | 100-199 >
[seq-#] remark < remark-str >
no < seq-# > remark

*This syntax applies to both named and numbered ACLs. Without an optional sequence number, the remark is appended to the end of the list and automatically assigned a sequence number. When entered with an optional sequence number, the remark is inserted in the list according to the numeric precedence of the sequence number. The **no** form of the command deletes the indicated remark, but does not affect the related ACE.*

To associate a remark with a specific ACE, enter the remark first, and then enter the ACE.

- Entering a remark without a sequence number and then entering an ACE without a sequence number results in the two entries being automatically paired with the same sequence number and appended to the end of the current ACL.*
- Entering a remark with a sequence number and then entering an ACE with the same sequence number results in the two entries being paired together and positioned in the list according to the sequence number they share.*

Note

After a numbered ACL has been created (using **access-list < 1 - 99 | 100 - 199 >**), it can be managed as either a named or numbered ACL. For example, in an existing ACL with a numeric identifier of “115”, either of the following command sets adds an ACE denying IP traffic from any IP source to a host at 10.10.10.100:

```
ProCurve(config)# access-list 115 deny ip host
10.10.10.100

ProCurve(config)# ip access-list extended 115
ProCurve(config-ext-nacl)# deny ip any 10.10.10.100
```

Appending Remarks and Related ACEs to the End of an ACL. To include a remark for an ACE that will be appended to the end of the current ACL, enter the remark first, then enter the related ACE. This results in the remark and the subsequent ACE having the same sequence number. For example, to add remarks using the “Named-ACL” (**nacl**) context:

```
ProCurve(config)# ip access-list standard My-List
ProCurve(config-std-nacl)# permit host 10.10.10.15
ProCurve(config-std-nacl)# deny 10.10.10.1/24
ProCurve(config-std-nacl)# remark HOST-10.20.10.34
ProCurve(config-std-nacl)# permit host 10.20.10.34
ProCurve(config-std-nacl)# show run
. . .
hostname "ProCurve"
ip access-list standard "My-List"
 10 permit 10.10.10.15 0.0.0.0
 20 deny 10.10.10.1 0.0.0.255
 30 remark "HOST-10.20.10.34"
 30 permit 10.20.10.34 0.0.0.0
exit
```

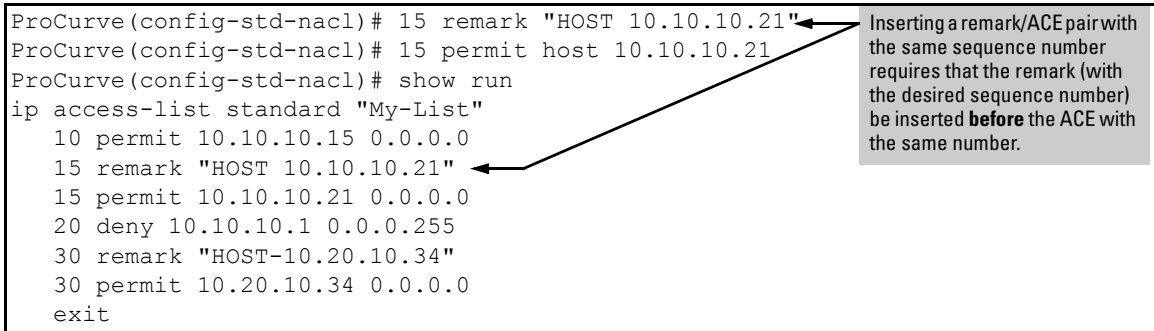
The remark is assigned the same number that the immediately following ACE (“30” in this example) is assigned when it is automatically appended to the end of the list. This operation applies where new remarks and ACEs are appended to the end of the ACL and are automatically assigned a sequence number.

Figure 10-33. Example of Appending a Remark and Its Related ACE to the End of an ACL

(You can also perform the operation illustrated in figure 10-33 by using the numbered, **access-list < 1 - 99 | 100 - 199 >** syntax shown at the beginning of this section.)

Inserting Remarks and Related ACEs Within an Existing List. To insert an ACE with a remark within an ACL by specifying a sequence number, insert the numbered remark first, then, using the same sequence number, insert the ACE. (This operation applies only to ACLs accessed using the “Named-ACL” (**nacl**) context.) For example:

```
ProCurve(config-std-nacl)# 15 remark "HOST 10.10.10.21"
ProCurve(config-std-nacl)# 15 permit host 10.10.10.21
ProCurve(config-std-nacl)# show run
ip access-list standard "My-List"
 10 permit 10.10.10.15 0.0.0.0
 15 remark "HOST 10.10.10.21"
 15 permit 10.10.10.21 0.0.0.0
 20 deny 10.10.10.1 0.0.0.255
 30 remark "HOST-10.20.10.34"
 30 permit 10.20.10.34 0.0.0.0
exit
```



Inserting a remark/ACE pair with the same sequence number requires that the remark (with the desired sequence number) be inserted **before** the ACE with the same number.

Inserting a Remark for an ACE that Already Exists in an ACL. If a sequence number is already assigned to an ACE in a list, you cannot insert a remark by assigning it to the same number. (To configure a remark with the same number as a given ACE, the remark must be configured first.) To assign a remark to the same number as an existing ACE:

1. Delete the ACE.
2. Configure the remark with the number you want assigned to the pair.
3. Re-Enter the deleted ACE with the number used to enter the remark.

Removing a Remark from an Existing ACE. If you want to remove a remark, but want to retain the ACE, do the following:

1. Use the Named ACL context to enter the ACL.
2. Using **show run** or **show access-list <list-name >**, note the sequence number and content of the ACE having a remark you want to remove.
3. Delete the ACE.
4. Using the same sequence number, re-enter the ACE.

Operating Notes for Remarks

- The **resequence** command ignores “orphan” remarks that do not have an ACE counterpart with the same sequence number. For example, if:
 - a remark numbered “55” exists in an ACE
 - there is no ACE numbered “55” in the same ACL
 - **resequence** is executed on an ACL

then the remark retains “55” as its sequence number and will be placed in the renumbered version of the ACL according to that sequence number.

- Entering an unnumbered remark followed by a numbered ACE, or the reverse, creates an “orphan” remark. The unnumbered entry will be assigned a sequence number that is an increment from the last ACE in the list. The numbered entry will then be placed sequentially in the list according to the sequence number used.
- Configuring two remarks without either sequence numbers or an intervening, unnumbered ACE results in the second remark overwriting the first.

```
ProCurve(config)# ip access-list standard Accounting
ProCurve(config-std-nacl)# permit host 10.10.10.115
ProCurve(config-std-nacl)# deny 10.10.10.1/24
ProCurve(config-std-nacl)# remark Marketing
ProCurve(config-std-nacl)# remark Channel_Mktg
ProCurve(config-std-nacl)# show run
.
.
.
ip access-list standard "Accounting"
 10 permit 10.10.10.115 0.0.0.0
 20 deny 10.10.10.1 0.0.0.255
 30 remark "Channel_Mktg"
exit
```

Where multiple remarks are sequentially entered for automatic inclusion at the end of an ACL, each successive remark replaces the previous one until an ACE is configured for automatic inclusion at the end of the list.

Figure 10-34. Example of Overwriting One Remark with Another

Displaying ACL Configuration Data

ACL Commands	Function	Page
show access-list	Displays a brief listing of all ACLs on the switch.	10-10 1
show access-list config	Display the type, identifier, and content of all ACLs configured in the switch.	10-10 2
show access-list vlan < vid >	List the name and type for each ACL application assigned to a particular VLAN on the switch.	10-10 3
show access-list ports < all port-list >	Lists the ACL static port assignment for either all ports and trunks, or for the specified ports and/or trunks.	
show access-list < acl-id >	Display detailed content information for a specific ACL.	10-10 5
show access-list resources	Displays the currently available per-slot resource availability. Refer to the appendix titled "Monitoring Resources" in the current <i>Management and Configuration Guide</i> for your switch.	
show access-list radius < all port-list >	Lists the RADIUS ACL(s) currently assigned for either all ports and trunks, or for the specified ports and/or trunks. For more on this topic, refer to chapter 7, "Configuring RADIUS Server Support for Switch Services".	
show config	show config includes configured ACLs and assignments existing in the startup-config file.	
show running	show running includes configured ACLs and assignments existing in the running-config file.	

Display an ACL Summary

This command lists the configured ACLs, regardless of whether they are assigned to any VLANs.

Syntax: show access-list

List a summary table of the name, type, and application status of all ACLs configured on the switch.

For example:

```
ProCurve(config)# show access-list

Access Control Lists

Type  Appl  Name
----  -
std   yes   List-01-Inbound
ext   no    List-02-Outbound
std   yes   55
```

In this switch, the ACL named "List-02-Outbound" exists in the configuration but is not applied to any VLANs and thus does not affect traffic.

Figure 10-35. Example of a Summary Table of Access lists

Term	Meaning
Type	Shows whether the listed ACL is std (Standard; source-address only) or ext (Extended; protocol, source, and destination data).
Appl	Shows whether the listed ACL has been applied to a VLAN (yes/no).
Name	Shows the identifier (name or number) assigned to each ACL configured in the switch.

Display the Content of All ACLs on the Switch

This command lists the configuration details for every ACL in the running-config file, regardless of whether any are actually assigned to filter IP traffic on specific VLANs.

Syntax: show access-list config

List the configured syntax for all ACLs currently configured on the switch.

Note

Notice that you can use the output from this command for input to an offline text file in which you can edit, add, or delete ACL commands. Refer to “Creating or Editing ACLs Offline” on page 10-108.

This information also appears in the **show running** display. If you executed **write memory** after configuring an ACL, it appears in the **show config** display.

For example, with two ACLs configured in the switch, you will see results similar to the following:

```
ProCurve(config)# show access-list config

ip access-list standard "List-43"
 10 deny 10.28.236.77 0.0.0.0
 20 deny 10.29.140.107 0.0.0.0
 30 permit 0.0.0.0 255.255.255.255
 exit
ip access-list extended "111"
 10 permit tcp 10.30.133.27 0.0.0.0 0.0.0.0 255.255.255.255
 20 permit tcp 10.30.155.101 0.0.0.0 0.0.0.0 255.255.255.255
 30 deny ip 10.30.133.1 0.0.0.0 0.0.0.0 255.255.255.255 log
 40 deny ip 10.30.155.1 0.0.0.255 0.0.0.0 255.255.255.255
 exit
```

Figure 10-36. Example of an ACL Configured Syntax Listing

Display the RACL and VACL Assignments for a VLAN

This command briefly lists the identification and type(s) of RACLs and VACLs currently assigned to a particular VLAN in the running-config file. (The switch allows one inbound and one outbound RACL assignment per VLAN, plus one VACL assignment.)

Syntax: show access-list vlan < vid >

Lists any RACL and/or VACL assignments to a VLAN in the running config file.

Note

This information also appears in the **show running** display. If you execute **write memory** after configuring an ACL, it also appears in the **show config** display.

For example, if you assigned an extended ACL with an ACL-ID of “List-43” to filter routed IP traffic exiting from the switch on VLAN 10 and a standard VACL with an ACL-ID of “List-12” to filter all IP traffic entering the switch on VLAN 10, you could verify these assignments as shown in figure 10-37:

```
ProCurve(config)# show access-list vlan 10

Access Lists for VLAN 10

  Inbound Access List: None
  Outbound Access List: List-43
  Type: Extended
  Vlan Access List : List-12
  Type: Standard
  Connection Rate Filter Access List: None
```

Indicates that:

- There is no ACL assignment to filter routed IP traffic entering the switch on VLAN 10.
- An extended ACL with the ID of “List-43” is assigned to filter routed IP traffic exiting the switch on VLAN 10.
- A standard ACL with the ID of “List-12” is assigned to filter all IP traffic entering the switch on VLAN 10.

Applies to Connection Rate Filter ACLs. (Refer to chapter 3, Virus Throttling”).

Figure 10-37. Example of Listing the ACL Assignments for a VLAN

Display Static Port ACL Assignments

This command briefly lists the identification and type(s) of current static port ACL assignments to individual switch ports and trunks, as configured in the running-config file. (The switch allows one static port ACL assignment per port.)

Syntax: `show access-list ports <all | interface >`

Lists the current static port ACL assignments for ports and trunks in the running config file.

Note

This information also appears in the **show running** display. If you execute **write memory** after configuring an ACL, it appears in the **show config** display.

For example, if you assigned a standard ACL with an ACL-ID of “Port-10” to filter inbound IP traffic on switch ports B10-B11 and trunk trk1, you could verify these assignments as shown in figure 10-38.

```
ProCurve(config)# show access-list ports all

Access Lists for Port B10

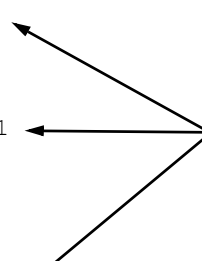
  Inbound  : 15
  Type     : Standard

Access Lists for Port B11

  Inbound  : 15
  Type     : Standard

Access Lists for Port Trk1

  Inbound  : 15
  Type     : Standard
```



Indicates that a standard ACL with the ID of “15” is assigned to filter traffic entering the switch on ports B10 and B11, and on trunk Trk1.

Figure 10-38. Example of Listing the ACL Assignments for Ports and Trunks

Displaying the Content of a Specific ACL

This command displays a specific ACL configured in the running config file in an easy-to-read tabular format.

Note

This information also appears in the **show running** display. If you execute **write memory** after configuring an ACL, it also appears in the **show config** display.

Syntax: show access-list < acl-id >

Display detailed information on the content of a specific ACL configured in the running-config file.

For example, suppose you configured the following two ACLs in the switch:

ACL ID	Type	Desired Action
1	Standard	<ul style="list-style-type: none">Deny IP traffic from 18.28.236.77 and 18.29.140.107.Permit IP traffic from all other sources.
105	Extended	<ul style="list-style-type: none">Permit any TCP traffic from 18.30.133.27 to any destination.Deny any other IP traffic from 18.30.133.(1-255).Permit all other IP traffic from any source to any destination.

Inspect the ACLs as follows:

```
ProCurve(config)# show access-list 1

Access Control Lists

Name: 1
Type: Standard
Applied: Yes

SEQ  Entry
-----
10   Action: deny (log)
     IP      : 10.28.236.77      Mask: 0.0.0.0

20   Action: deny
     IP      : 10.29.140.107    Mask: 0.0.0.0

30   Action: permit
     IP      : 0.0.0.0          Mask: 255.255.255.255
```

Indicates whether the ACL is applied to an interface.

Figure 10-39. Example of a Listing a Standard ACL

Access Control Lists (ACLs)
Displaying ACL Configuration Data

```
ProCurve(config)# show access-list List-120

Access Control Lists

Name: List-120
Type: Extended
Applied: No

SEQ  Entry
-----
10   Action: permit
     Remark: Telnet Allowed
     Src IP: 10.30.133.27      Mask: 0.0.0.0      Port (s): eq 23
     Dst IP: 0.0.0.0         Mask: 255.255.255.255  Port (s):
     Proto : TCP (Established)
     TOS   : -                Precedence: routine

20   Action: deny (log)
     Src IP: 10.30.133.1      Mask: 0.0.0.255    Port (s):
     Dst IP: 0.0.0.0         Mask: 255.255.255.255  Port (s):
     Proto : IP
     TOS   : -                Precedence: -

30   Action: permit
     Src IP: 0.0.0.0          Mask: 255.255.255.255  Port (s):
     Dst IP: 0.0.0.0         Mask: 255.255.255.255  Port (s):
     Proto : IP
     TOS   : -                Precedence: -
```

The diagram includes three callout boxes with arrows pointing to specific parts of the ACL output:

- A box labeled "Indicates whether the ACL is applied to an interface." points to the "Applied: No" line.
- A box labeled "Indicates source and destination entries in the ACL." points to the "SEQ" and "Entry" headers.
- A box labeled "Empty field indicates that the destination TCP port can be any value." points to the empty "Port (s):" field in entry 20.

Figure 10-40. Examples of Listings Showing the Content of Standard and Extended ACLs

Table 10-11. Descriptions of Data Types Included in Show Access-List < acl-id > Output

Field	Description
Name	The ACL identifier. Can be a number from 1 to 199, or a name.
Type	Standard or Extended. The former uses only source IP addressing. The latter uses both source and destination IP addressing and also allows TCP or UDP port specifiers.
Applied	“Yes” means the ACL has been applied to a port or VLAN interface. “No” means the ACL exists in the switch configuration, but has not been applied to any interface, and is therefore not in use.
SEQ	The sequential number of the Access Control Entry (ACE) in the specified ACL.
Entry	Lists the content of the ACEs in the selected ACL.
Action	Permit (forward) or deny (drop) a packet when it is compared to the criteria in the applicable ACE and found to match. Includes the optional log option, if used, in deny actions.
Remark	Displays any optional remark text configured for the selected ACE.
IP	Used for Standard ACLs: The source IP address to which the configured mask is applied to determine whether there is a match with a packet.
Src IP	Used for Extended ACLs: Same as above.
Dst IP	Used for Extended ACLs: The source and destination IP addresses to which the corresponding configured masks are applied to determine whether there is a match with a packet.
Mask	The mask configured in an ACE and applied to the corresponding IP address in the ACE to determine whether a packet matches the filtering criteria.
Proto	Used only in extended ACLs to specify the packet protocol type to filter. Must be either IP, TCP, or UDP. For TCP protocol selections, includes the established option, if configured.
Port(s)	Used only in extended ACLs to show any TCP or UDP operator and port number(s) included in the ACE.
TOS	Used only in extended ACLs to indicate Type-of-Service setting, if any.
Precedence	Used only in extended ACLs to indicate the IP precedence setting, if any.

Display All ACLs and Their Assignments in the Routing Switch Startup-Config File and Running-Config File

The **show config** and **show running** commands include in their listings any configured ACLs and any ACL assignments to VLANs. Refer to figure 10-12 (page 10-46) for an example. Remember that **show config** lists the startup-config file and **show running** lists the running-config file.

Creating or Editing ACLs Offline

The section titled “Editing an Existing ACL” on page 10-90 describes how to use the CLI to edit an ACL, and is most applicable in cases where the ACL is short or there is only a minor editing task to perform. The offline method provides a useful alternative to using the CLI for creating or extensively editing a large ACL. This section describes how to:

- move an existing ACL to a TFTP server
- use a text (.txt) file format to create a new ACL or edit an existing ACL offline
- use TFTP to load an offline ACL into the switch’s running-config

For longer ACLs that may be difficult or time-consuming to accurately create or edit in the CLI, you can use the offline method described in this section.

Note

Beginning with software release K_12_XX or later, **copy** commands that used either **tftp** or **xmodem**, also include an option to use **usb** as a source or destination device for file transfers. So although the following example highlights tftp, bear in mind that **xmodem** or **usb** can also be used to transfer ACLs to and from the switch.

Creating or Editing an ACL Offline

The Offline Process

1. Begin by doing one of the following:
 - To edit one or more existing ACLs, use **copy command-output tftp** to copy the current version of the ACL configuration to a file in your TFTP server. For example, to copy the ACL configuration to a file named **acl-02.txt** in the TFTP directory on a server at 10.28.227.2:

```
ProCurve# copy command-output 'show access-list config' tftp 10.28.227.2 acl02.txt pc
```
 - To create a new ACL, just open a text (.txt) file in the appropriate directory on a TFTP server accessible to the switch.
2. Use a text editor to create or edit the ACL(s) in the ***.txt** ASCII file format.

If you are replacing an ACL on the switch with a new ACL that uses the same number or name syntax, begin the command file with a **no ip access-list** command to remove the earlier version of the ACL from the switch's running-config file. Otherwise, the switch will append the new ACEs in the ACL you download to the existing ACL. For example, if you planned to use the **copy** command to *replace* ACL "List-120", you would place this command at the beginning of the edited file:

```
no ip access-list extended List-120
```

<pre>no ip access-list extended List-120 ip access-list extended "List-120" 10 remark "THIS ACE ALLOWS TELNET" 10 permit tcp 10.30.133.27 0.0.0.0 eq 23 0.0.0.0 255.255.255. 20 deny ip 10.30.133.1 0.0.0.255 0.0.0.0 255.255.255.255 30 deny ip 10.30.155.1 0.0.0.255 0.0.0.0 255.255.255.255 40 remark "THIS IS THE FINAL ACE IN THE LIST" 40 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255 exit</pre>	<p>←</p> <p>←</p> <p>Removes an existing ACL and replaces it with a new version with the same identity. To append new ACEs to an existing ACL instead of replacing it, you would omit the first line and ensure that the sequence numbering for the new ACEs begin with a number greater than the highest number in the existing list.</p>
--	--

Figure 10-41. Example of an Offline ACL File Designed To Replace An Existing ACL

3. Use **copy tftp command-file** to download the file as a list of commands to the switch.

Example of Using the Offline Process

For example, suppose that you wanted to create an extended ACL for an RACL application to fulfill the following requirements (Assume a subnet mask of 255.255.255.0 and a TFTP server at 10.10.10.1):

- ID: "LIST-20-IN"
- Deny Telnet access to a server at 10.10.10.100 on VLAN 10 from these three IP addresses on VLAN 20 (with ACL logging):
 - 10.10.20.17
 - 10.10.20.23
 - 10.10.20.40
- Allow any access to the server from all other addresses on VLAN 20:
- Permit internet access to these two IP address on VLAN 20, but deny access to all other addresses on VLAN 20 (without ACL logging).
 - 10.10.20.98
 - 10.10.20.21

- Deny all other IP traffic from VLAN 20 to VLAN 10.
 - Deny all IP traffic from VLAN 30 (10.10.30.0) to the server at 10.10.10.100 on VLAN 10 (without ACL logging), but allow any other IP traffic from VLAN 30 to VLAN 10.
 - Deny all other inbound IP traffic to VLAN 20. (Hint: The Implicit Deny can achieve this objective.)
1. You would create a **.txt** file with the content shown in figure 10-43.

```
ip access-list extended LIST-20-IN

; CREATED ON JUNE 27

10 remark "THIS ACE APPLIES INBOUND ON VLAN 20"
10 permit tcp any host 10.10.20.98 eq http
20 permit tcp any host 10.10.20.21 eq http
30 deny tcp any 10.10.20.1/24 eq http

; VLAN 20 SOURCES TO VLAN 10 DESTINATIONS.

40 deny tcp host 10.10.20.17 host 10.10.10.100 eq telnet log
50 deny tcp host 10.10.20.23 host 10.10.10.100 eq telnet log
60 deny tcp host 10.10.20.40 host 10.10.10.100 eq telnet log
70 permit ip 10.10.20.1/24 host 10.10.10.100
80 remark "VLAN 30 POLICY."
80 deny ip 10.10.30.1/24 host 10.10.10.100
90 permit ip 10.10.30.1/24 10.10.10.1/24
exit
vlan 20 ip access-group "LIST-20-in" in
```

The ";" enables a comment in the file.

Note: You can use the ";" character to denote a comment. The file stored on your TFTP server retains comments, and they appear when you use **copy** to download the ACL command file. (Comments are not saved in the switch configuration.)

Figure 10-42. Example of a .txt File Designed for Creating an ACL

2. After you copy the above .txt file to a TFTP server the switch can access, you would then execute the following command:

copy tftp command-file 10.10.10.1 LIST-20-IN.txt pc

In this example, the CLI would show the following output to indicate that the ACL was successfully downloaded to the switch:

Note

If a transport error occurs, the switch does not execute the command and the ACL is not configured.

```
ProCurve(config)# copy tftp command-file 10.10.10.1 LIST-20-IN.txt pc
Running configuration may change, do you want to continue [y/n]? Y
 1. ip access-list extended LIST-20-IN
 3. ; CREATED ON JUNE 27
 5. 10 remark "THIS ACE APPLIES INBOUND ON VLAN 20"
 6. 10 permit tcp any host 10.10.20.98 eq http
 7. 20 permit tcp any host 10.10.20.21 eq http
 8. 30 deny tcp any 10.10.20.1/24 eq http
10. ; VLAN 20 SOURCES TO VLAN 10 DESTINATIONS.
12. 40 deny tcp host 10.10.20.17 host 10.10.10.100 eq telnet log
13. 50 deny tcp host 10.10.20.23 host 10.10.10.100 eq telnet log
14. 60 deny tcp host 10.10.20.40 host 10.10.10.100 eq telnet log
15. 70 permit ip 10.10.20.1/24 host 10.10.10.100
16. 80 remark "VLAN 30 POLICY."
17. 80 deny ip 10.10.30.1/24 host 10.10.10.100
18. 90 permit ip 10.10.30.1/24 10.10.10.1/24
19. exit
20. vlan 20 ip access-group "LIST-20-in" in
```

As illustrated here, blank lines in the .txt file in figure 10-41 cause breaks in the displayed line-numbering sequence when you copy the command file to the switch. This is normal operation. (See also figure 10-44 for the configuration resulting from this output.)

Figure 10-43. Example of Using “copy tftp command-file” To Configure an ACL in the Switch

3. In this example, the command to assign the ACL to a VLAN was included in the .txt command file. If this is not done in your applications, then the next step is to manually assign the new ACL to the intended VLAN.

vlan < vid > ip access-group < identifier > in

4. You can then use the **show run** or **show access-list config** command to inspect the switch configuration to ensure that the ACL was properly downloaded.

Access Control Lists (ACLs) Creating or Editing ACLs Offline

```
ProCurve(config)# show run
. . .
ip access-list extended "LIST-20-IN"
 10 remark "THIS ACE APPLIES INBOUND ON VLAN 20"
 10 permit tcp 0.0.0.0 255.255.255.255 10.10.20.98 0.0.0.0 eq 80
 20 permit tcp 0.0.0.0 255.255.255.255 10.10.20.21 0.0.0.0 eq 80
 30 deny tcp 0.0.0.0 255.255.255.255 10.10.20.1 0.0.0.255 eq 80
 40 deny tcp 10.10.20.17 0.0.0.0 10.10.10.100 0.0.0.0 eq 23 log
 50 deny tcp 10.10.20.23 0.0.0.0 10.10.10.100 0.0.0.0 eq 23 log
 60 deny tcp 10.10.20.40 0.0.0.0 10.10.10.100 0.0.0.0 eq 23 log
 70 permit ip 10.10.20.1 0.0.0.255 10.10.10.100 0.0.0.0
 80 remark "VLAN 30 POLICY."
 80 deny ip 10.10.30.1 0.0.0.255 10.10.10.100 0.0.0.0
 90 permit ip 10.10.30.1 0.0.0.255 10.10.10.1 0.0.0.255
 exit
. . .
vlan 20
 name "VLAN20"
 no ip address
 ip access-group "LIST-20-in" in
 exit
```

Note that the comments preceded by ";" in the .txt source file for this configuration do not appear in the ACL configured in the switch.

As a part of the instruction set included in the .txt file, the ACL is assigned to inbound IP traffic on VLAN 20.

Figure 10-44. Example of Verifying the .txt File Download to the Switch

5. If the configuration appears satisfactory, save it to the startup-config file:

```
ProCurve(config)# write memory
```


Enable ACL “Deny” Logging

ACL logging enables the switch to generate a message when IP traffic meets the criteria for a match with an ACE that results in an explicit “deny” action. You can use ACL logging to help:

- Test your network to ensure that your ACL configuration is detecting and denying the IP traffic you do not want forwarded
- Receive notification when the switch detects attempts to forward IP traffic you have designed your ACLs to reject (deny)

The switch sends ACL messages to Syslog and optionally to the current console, Telnet, or SSH session. You can use **logging < >** to configure up to six Syslog server destinations.

Requirements for Using ACL Logging

- The switch configuration must include an ACL (1) assigned to a port, trunk, or static VLAN interface and (2) containing an ACE configured with the **deny** action and the **log** option.
- If the RACL application is used, then IP routing must be enabled on the switch.
- For ACL logging to a Syslog server:
 - The server must be accessible to the switch and identified in the running configuration.
 - The logging facility must be enabled for Syslog.
 - Debug must be configured to:
 - support ACL messages
 - send debug messages to the desired debug destination

These requirements are described in more detail under “Enabling ACL Logging on the Switch” on page 10-115.

ACL Logging Operation

When the switch detects a packet match with an ACE and the ACE includes both the **deny** action and the optional **log** parameter, an ACL log message is sent to the designated debug destination. The first time a packet matches an ACE with **deny** and **log** configured, the message is sent immediately to the destination and the switch starts a wait-period of approximately five minutes. (The exact duration of the period depends on how the packets are internally routed.) At the end of the collection period, the switch sends a single-line summary of any additional "deny" matches for that ACE (and any other "deny" ACEs for which the switch detected a match). If no further log messages are generated in the wait-period, the switch suspends the timer and resets itself to send a message as soon as a new "deny" match occurs. The data in the message includes the information illustrated in figure 10-45.

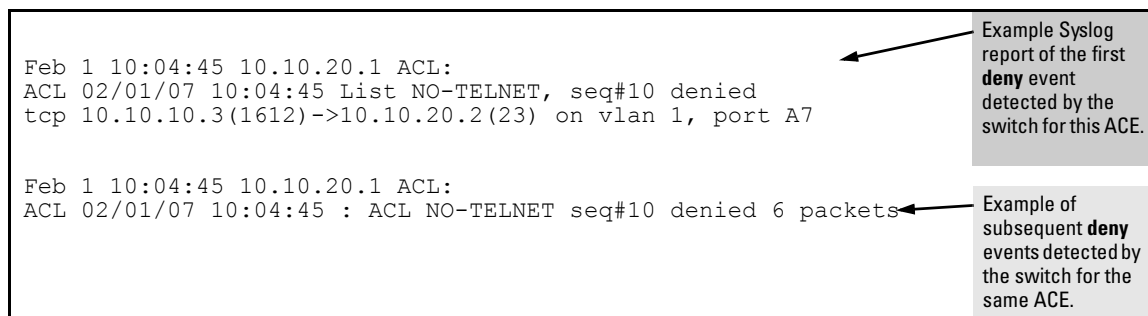


Figure 10-45. Content of a Message Generated by an ACL-Deny Action

Enabling ACL Logging on the Switch

1. If you are using a Syslog server, use the **logging < ip-addr >** command to configure the Syslog server IP address(es). Ensure that the switch can access any Syslog server(s) you specify.
2. Use **logging facility syslog** to enable the logging for Syslog operation.
3. Use the **debug destination** command to configure one or more log destinations. (Destination options include **logging**, **session**, and **windshell**. For more information on debug, refer to "Debug and Syslog Messaging Operation" in appendix C, "Troubleshooting", in the *Management and Configuration Guide* for your switch.)
4. Use **debug acl** or **debug all** to configure the debug operation to include ACL messages.
5. Configure one or more ACLs with the **deny** action and the **log** option.

For example, suppose that you want to configure the following operation:

- On VLAN 10 configure an extended ACL with an ACL-ID of "NO-TELNET" and use the RACL **in** option to deny Telnet traffic entering the switch from IP address 10.10.10.3 to any routed destination. (Note that this assignment will not filter Telnet traffic from 10.10.10.3 to destinations on VLAN 10 itself.)
- Configure the switch to send an ACL log message to the current console session and to a Syslog server at IP address 10.10.20.3 on VLAN 20 if the switch detects a packet match denying a Telnet attempt from 10.10.10.3.

(This example assumes that IP routing is already configured on the switch.)

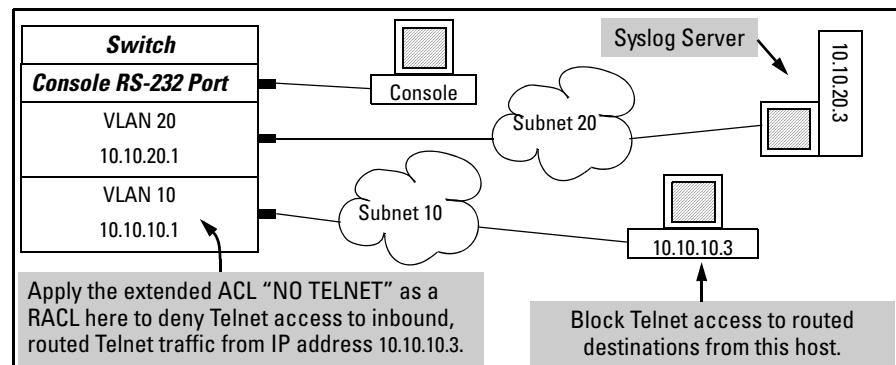


Figure 10-46. Example of an ACL Log Application

Access Control Lists (ACLs) Enable ACL "Deny" Logging

```
ProCurve(config)# ip access-list extended NO-TELNET
ProCurve(config-ext-nacl)# remark "DENY 10.10.10.3 TELNET TRAFFIC IN"
ProCurve(config-ext-nacl)# deny tcp host 10.10.10.3 any eq telnet log
ProCurve(config-ext-nacl)# permit ip any any
ProCurve(config-ext-nacl)# exit
ProCurve(config)# vlan 10 ip access-group NO-TELNET in
ProCurve(config)# logging 10.10.20.3
ProCurve(config)# logging facility syslog
ProCurve(config)# debug destination logging
ProCurve(config)# debug destination session
ProCurve(config)# debug acl
ProCurve(config)# write mem
ProCurve(config)# show debug

Debug Logging

Destination:
Logging --
  10.10.20.3
  Facility = syslog
  Session

Enabled debug types:
  event
  acl log

ProCurve(config)# show access-list config

ip access-list extended "NO-TELNET"
  10 remark "DENY 10.10.10.3 TELNET TRAFFIC"
  10 deny tcp 10.10.10.5 0.0.0.0 0.0.0.0 255.255.255.255 eq 23 log
  20 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
  exit
```



Figure 10-47. Commands for Applying an ACL with Logging to Figure 10-46

General ACL Operating Notes

ACLs do not provide DNS hostname support. ACLs cannot be configured to screen hostname IP traffic between the switch and a DNS.

ACLs Do Not Affect Serial Port Access. ACLs do not apply to the switch's serial port.

ACL Screening of IP Traffic Generated by the Switch. Outbound RACL applications on a switch do not screen IP traffic (such as broadcasts, Telnet, Ping, and ICMP replies) *generated by the switch itself*. Note that all ACLs applied on the switch do screen this type of IP traffic when other devices generate it. Similarly, all ACL applications can screen responses from other devices to unscreened IP traffic the switch generates.

ACL Logging.

- The ACL logging feature generates a message only when packets are explicitly denied as the result of a match, and not when explicitly permitted or implicitly denied. To help test ACL logging, configure the last entry in an ACL as an explicit **deny** statement with a **log** statement included, and apply the ACL to an appropriate VLAN.
- Logging enables you to selectively test specific devices or groups. However, excessive logging can affect switch performance. For this reason, ProCurve recommends that you remove the logging option from ACEs for which you do not have a present need. Also, avoid configuring logging where it does not serve an immediate purpose. (Note that ACL logging is not designed to function as an accounting method.) See also “Apparent Failure To Log All ‘Deny’ Matches” in the section titled “ACL Problems”, found in appendix C, “Troubleshooting” of the *Management and Configuration Guide* for your switch.
- When configuring logging, you can reduce excessive resource use by configuring the appropriate ACEs to match with specific hosts instead of entire subnets. (For more on resource usage, refer to “Monitoring Shared Resources” on page 10-118.)

Minimum Number of ACEs in an ACL. Any ACL must include at least one ACE to enable IP traffic screening. A numbered ACL cannot be created without at least one ACE. A named ACL can be created “empty”; that is, without any ACEs. However in an empty ACL applied to an interface, the Implicit Deny function does not operate, and the ACL has no effect on traffic.

Monitoring Shared Resources. Applied ACLs share internal switch resources with several other features. The switch provides ample resources for all features. However, if the internal resources become fully subscribed, additional ACLs cannot be applied until the necessary resources are released from other applications. For information on determining current resource availability and usage, refer to appendix E, “Monitoring Resources” in the *Management and Configuration Guide* for your switch.

Protocol Support . ACL criteria does not include use of MAC information or QoS.

Replacing or Adding To an Active ACL Policy. If you assign an ACL to an interface and subsequently add or replace ACEs in that ACL, each new ACE becomes active when you enter it. If the ACL is configured on multiple interfaces when the change occurs, then the switch resources must accommodate all applications of the ACL. If there are insufficient resources to accommodate one of several ACL applications affected by the change, then the change is not applied to any of the interfaces and the previous version of the ACL remains in effect. Refer to “Monitoring Shared Resources”, above.

“Strict” TCP and UDP. When the ACL configuration includes TCP or UDP options, the switch operates in “strict” TCP and UDP mode for increased control. In this case, the switch compares all TCP and UDP packets against the ACLs. (In the ProCurve 9300m and 9404sl Routing Switches, the Strict TCP and Strict UDP modes are optional and must be specifically invoked.)