

Chapter 10

Basic IP Traffic Management with Access Lists

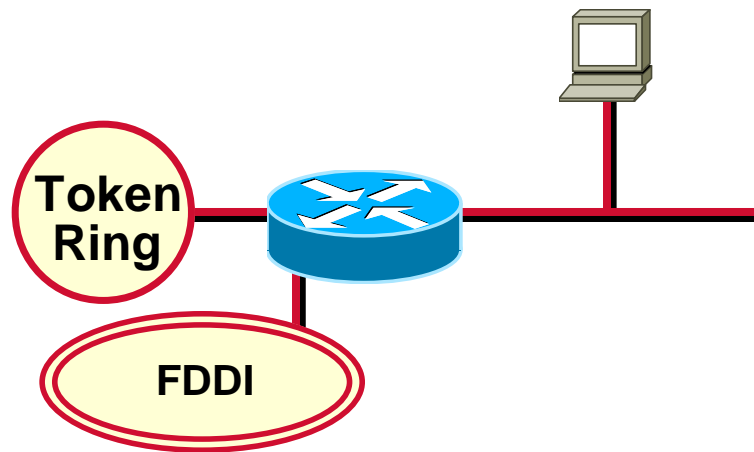


Objectives

Upon completion of this chapter, you will be able to perform the following tasks:

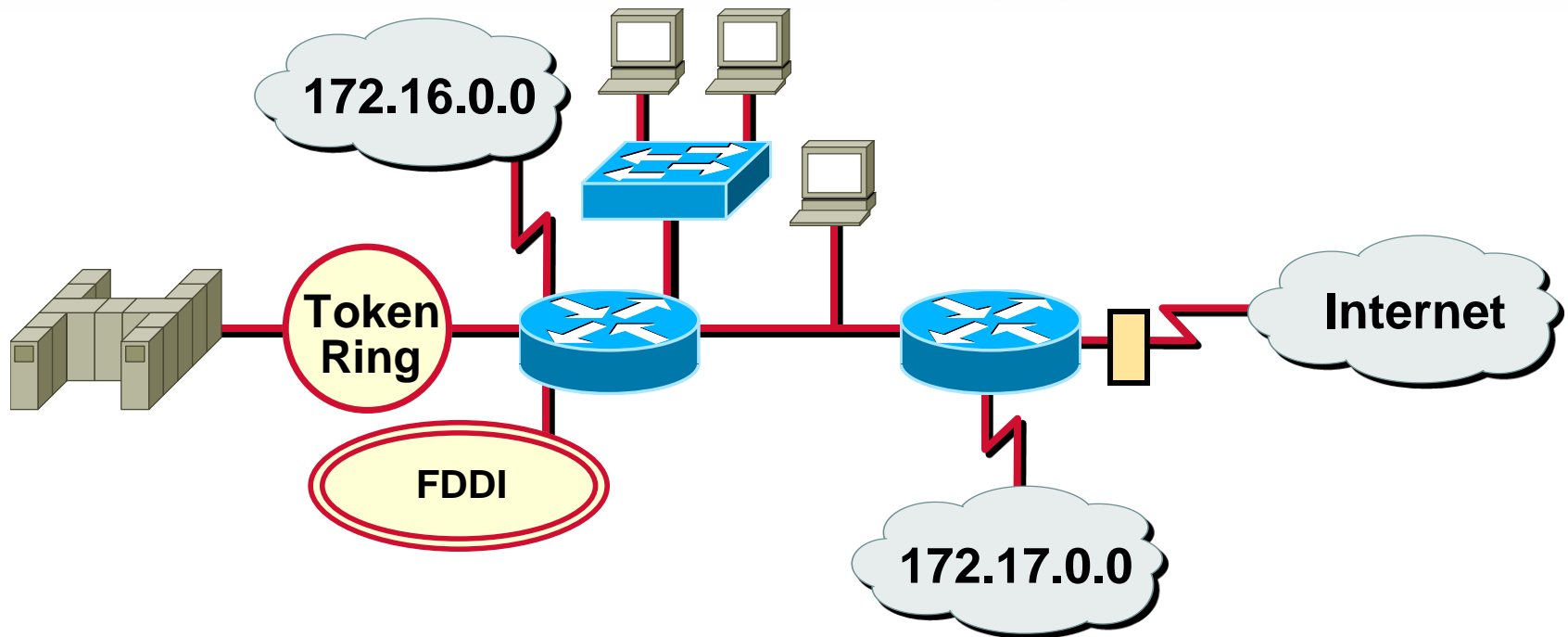
- **Identify the key functions and special processing of IP access lists**
- **Configure standard IP access lists**
- **Control virtual terminal access with access class**
- **Configure extended IP access lists**
- **Verify and monitor IP access lists**

Why Use Access Lists?



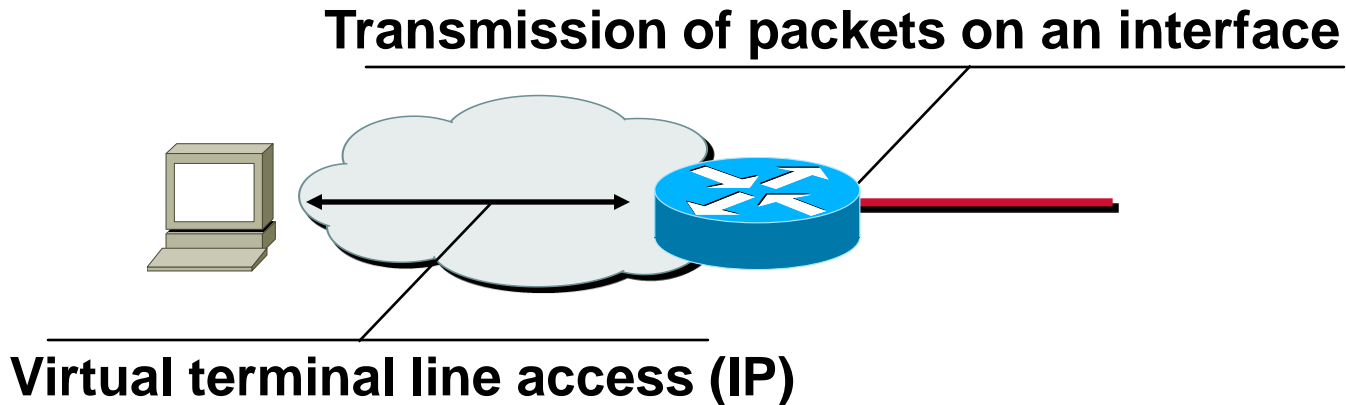
- **Manage IP Traffic as network access grows**

Why Use Access Lists?



- **Manage IP traffic as network access grows**
- **Filter packets as they pass through the router**

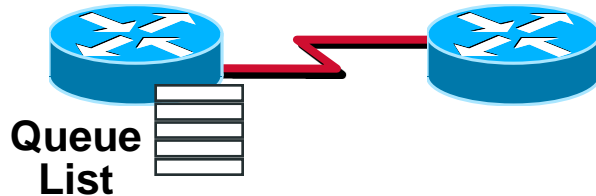
Access List Applications



- **Permit or deny packets moving through the router**
- **Permit or deny vty access to or from the router**
- **Without access lists all packets could be transmitted onto all parts of your network**

Other Access List Uses

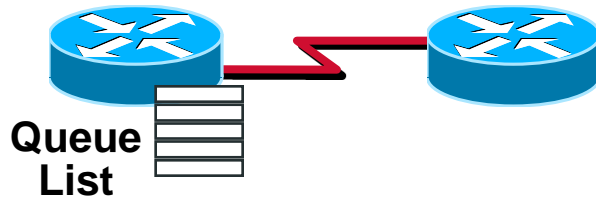
Priority and custom queuing



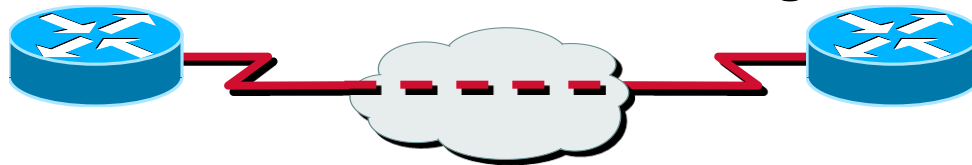
Special handling for traffic based on packet tests

Other Access List Uses

Priority and custom queuing



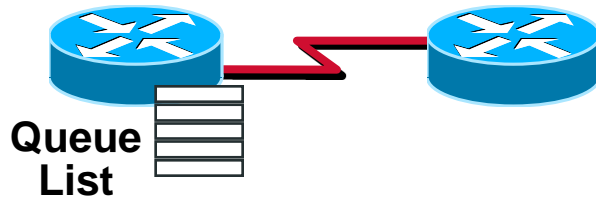
Dial-on-demand routing



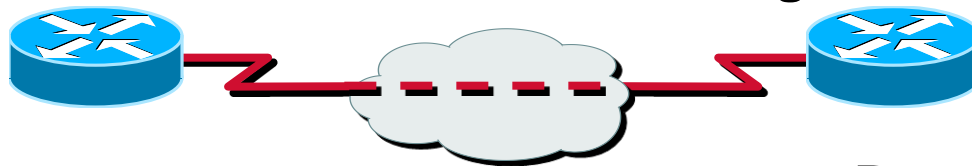
Special handling for traffic based on packet tests

Other Access List Uses

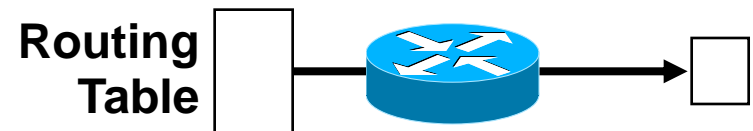
Priority and custom queuing



Dial-on-demand routing

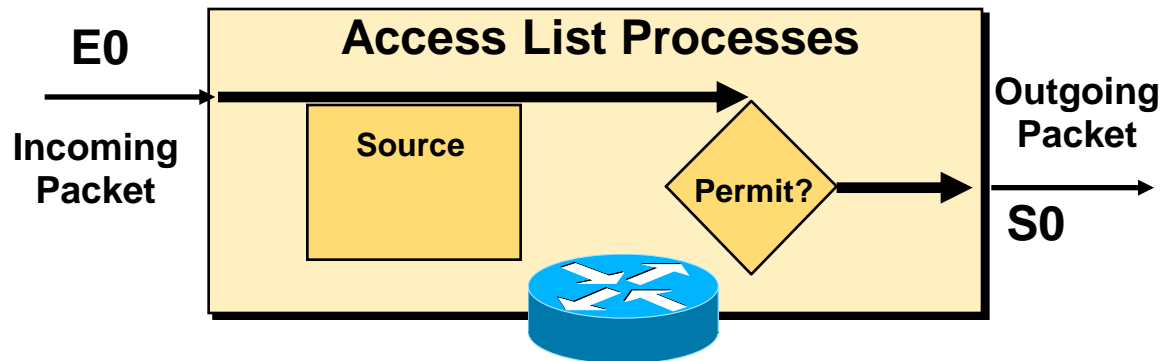


Route filtering



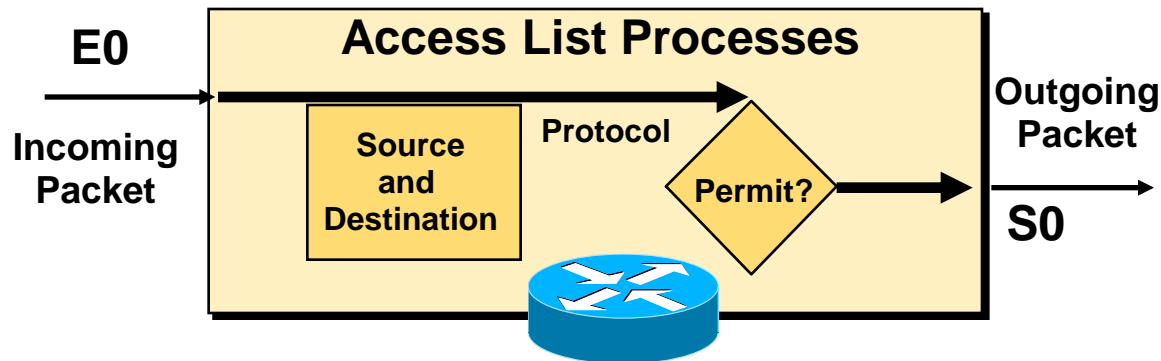
Special handling for traffic based on packet tests

What Are Access Lists?



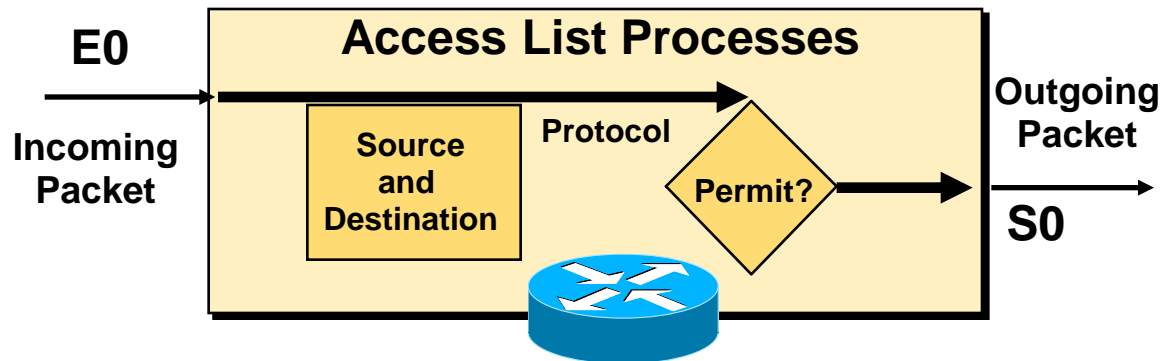
- **Standard**
 - Checks Source address
 - Generally permits or denies entire protocol suite

What Are Access Lists?



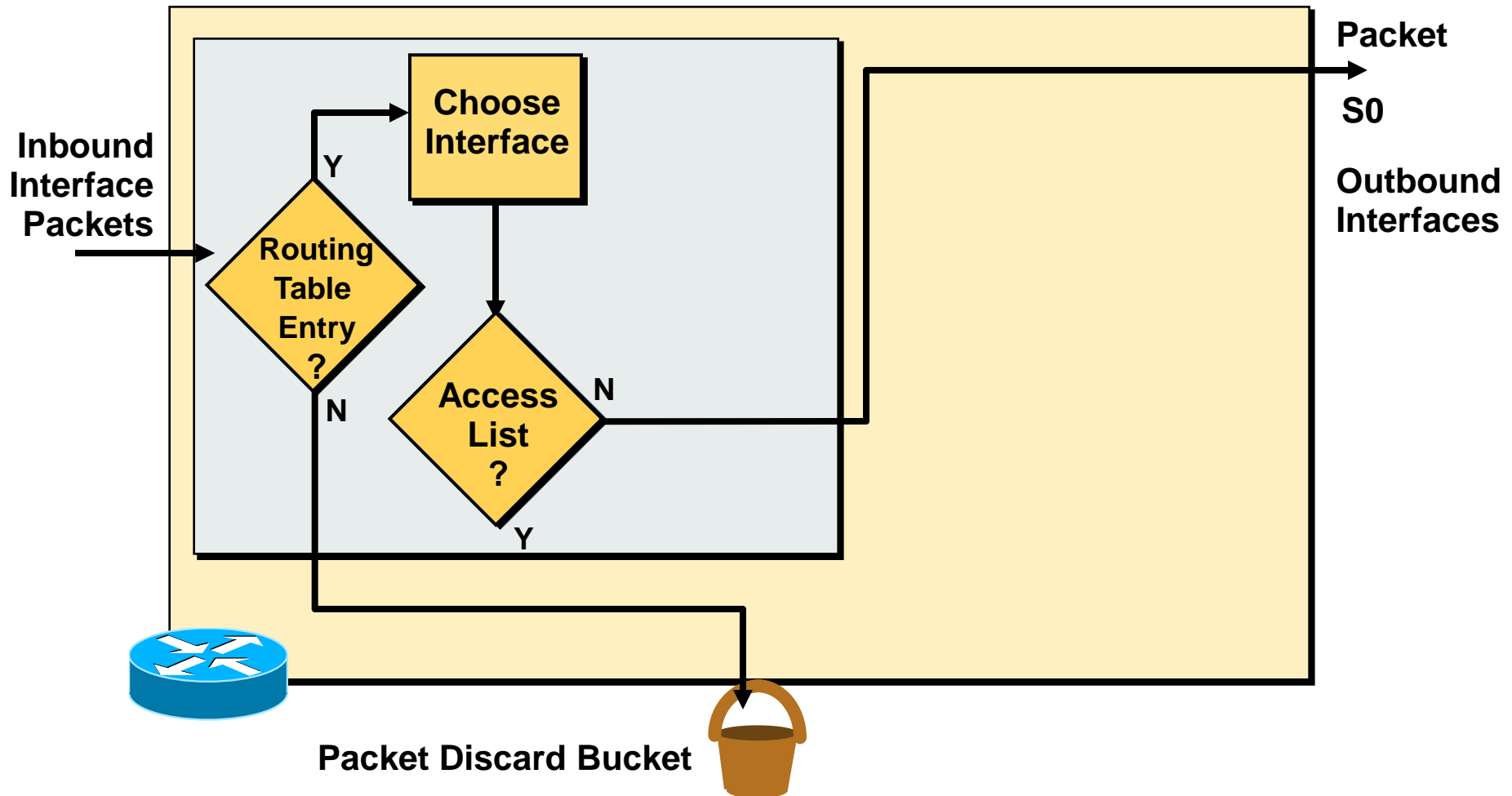
- **Standard**
 - Checks Source address
 - Generally permits or denies entire protocol suite
- **Extended**
 - Checks Source and Destination address
 - Generally permits or denies specific protocols

What Are Access Lists?

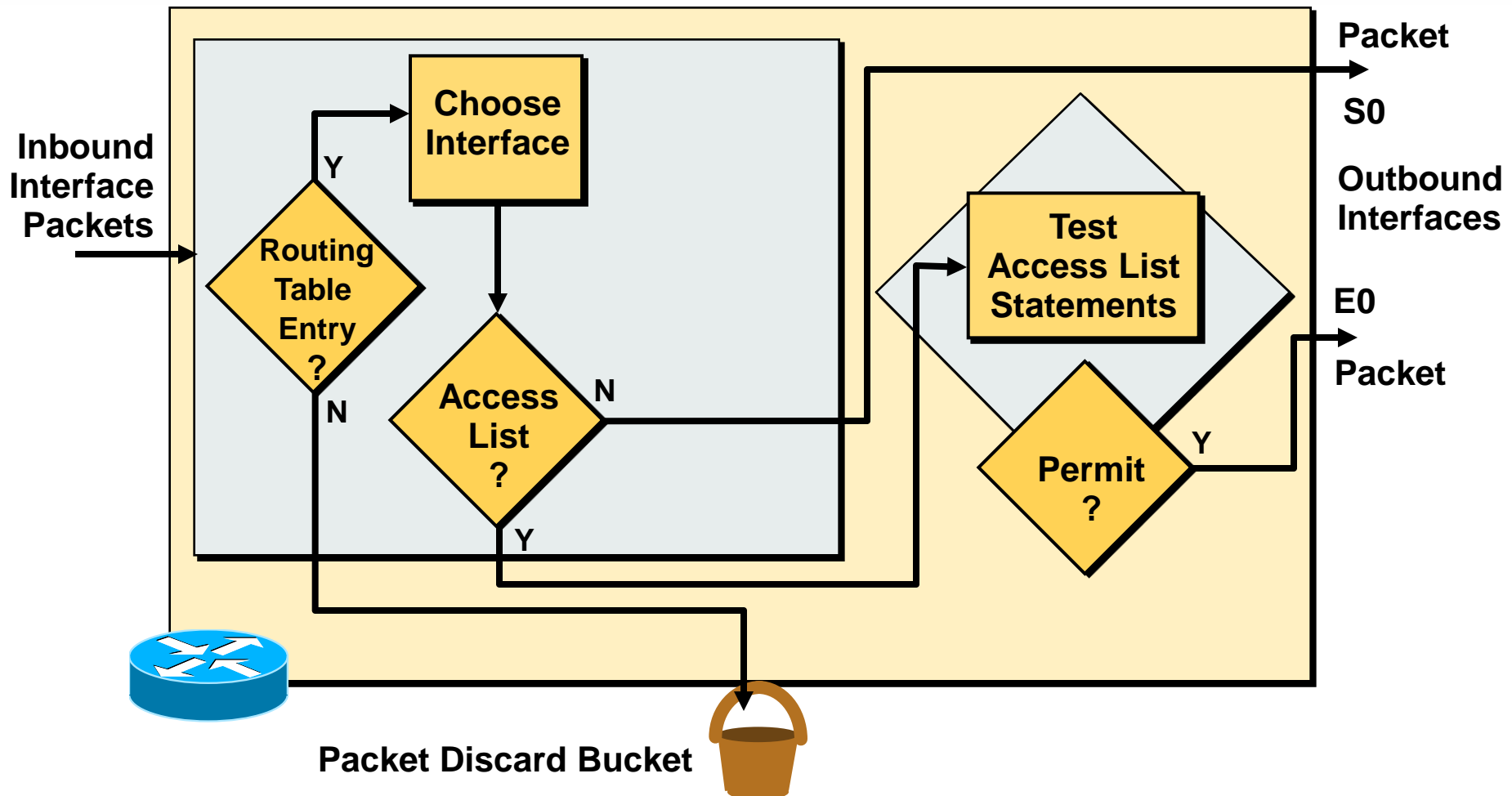


- **Standard**
 - Checks Source address
 - Generally permits or denies entire protocol suite
- **Extended**
 - Checks Source and Destination address
 - Generally permits or denies specific protocols
- **Inbound or Outbound**

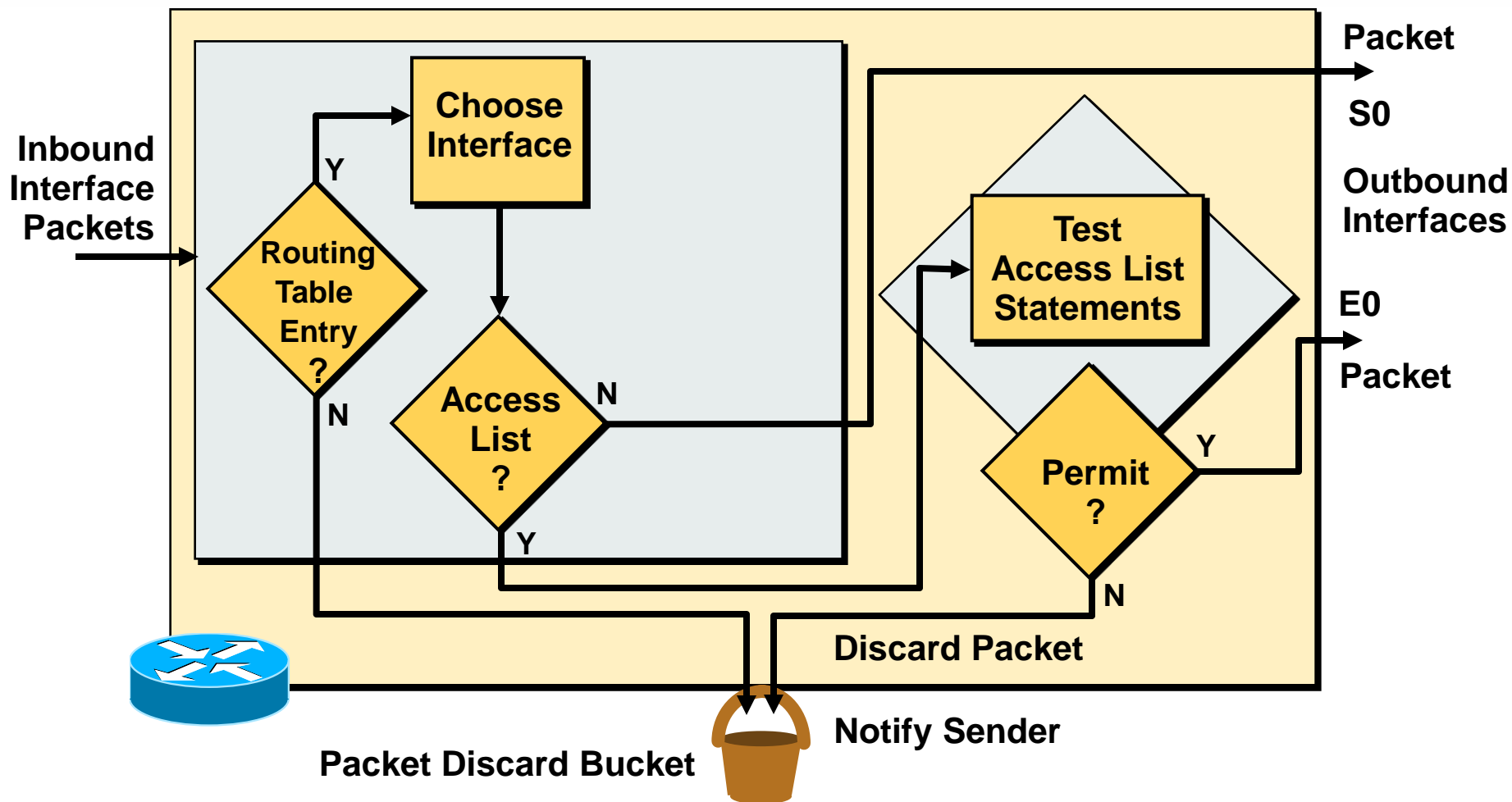
Outbound Access Lists



Outbound Access Lists

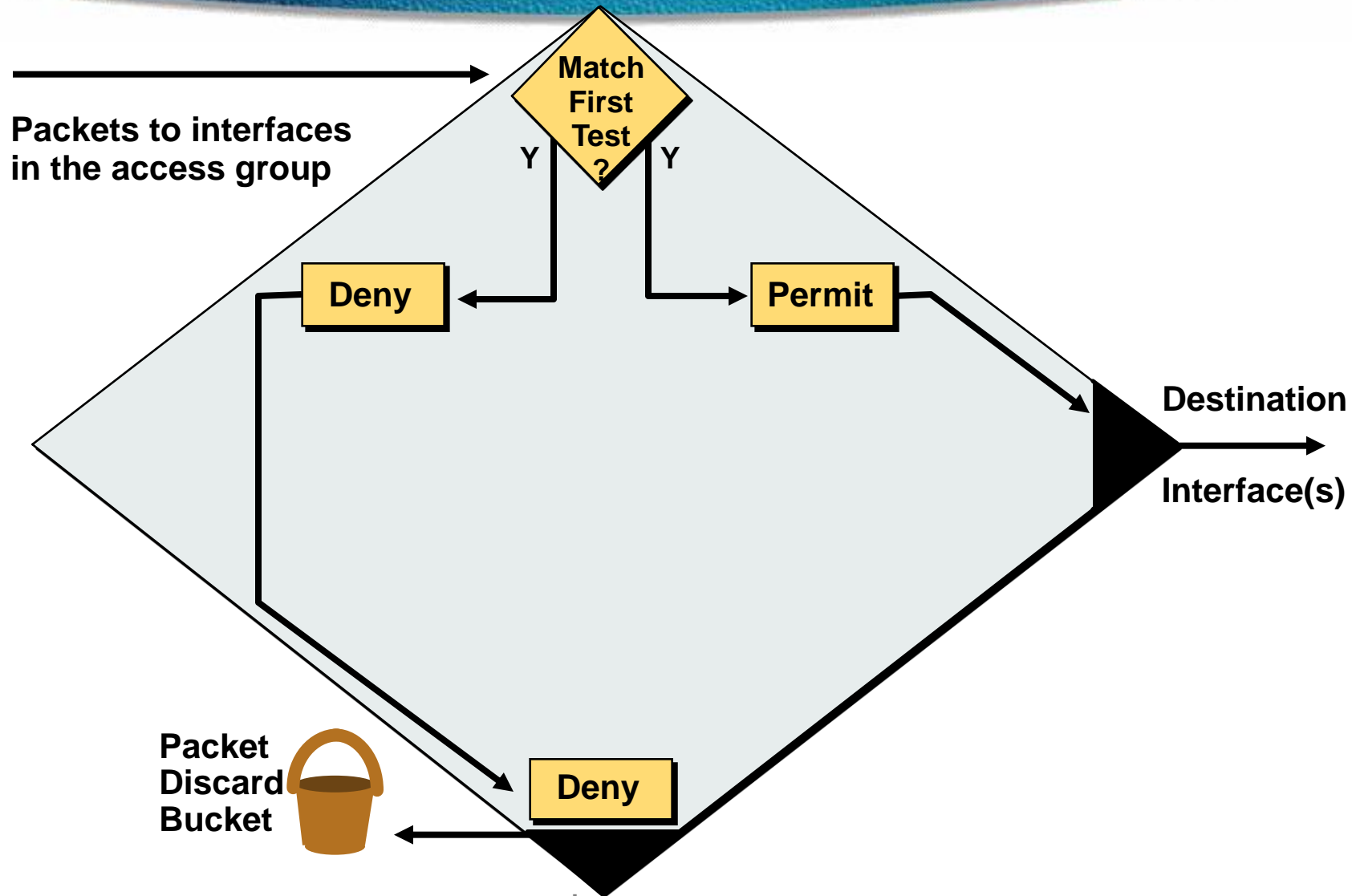


Outbound Access Lists

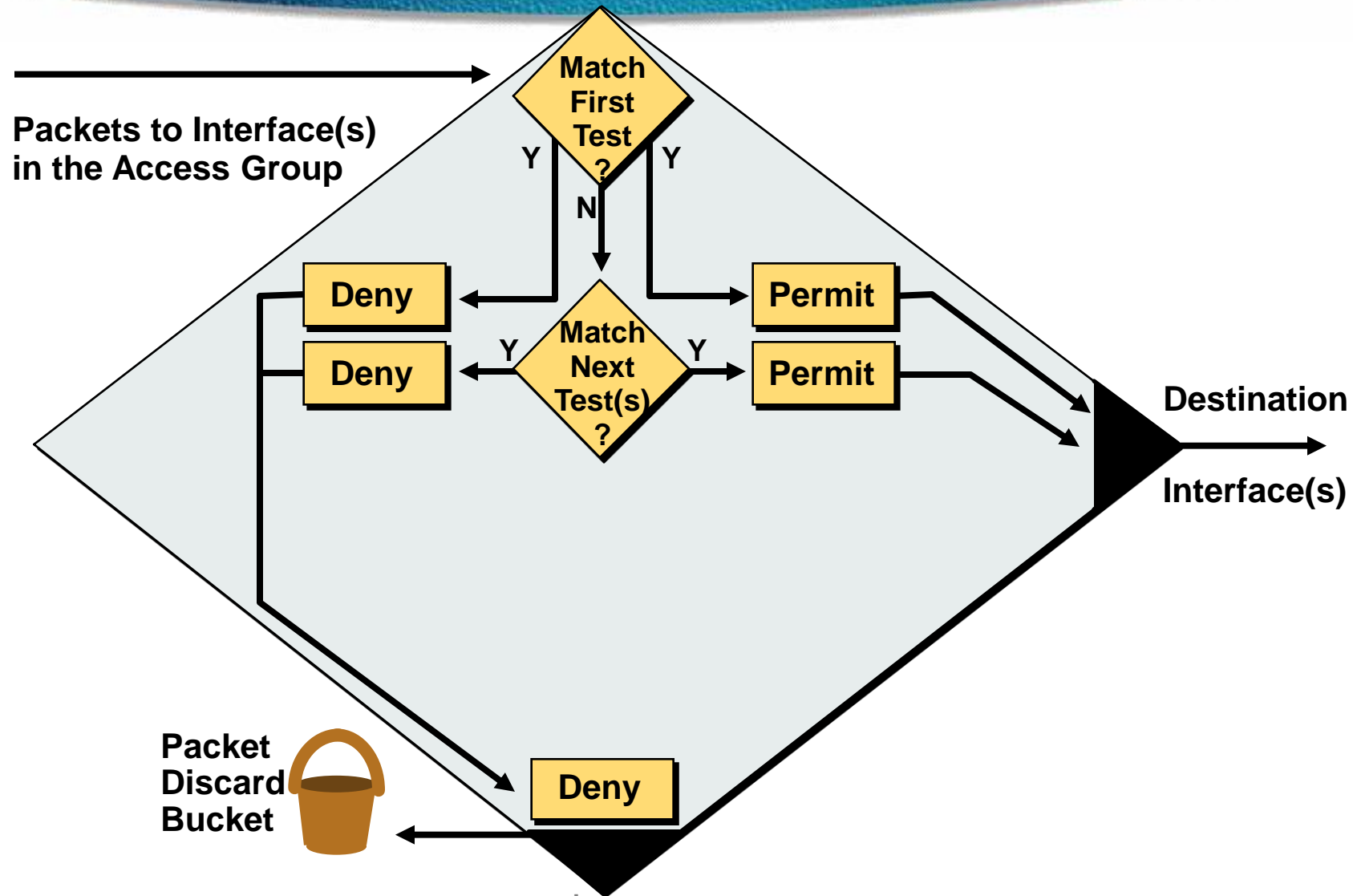


If no access list statement matches then discard the packet

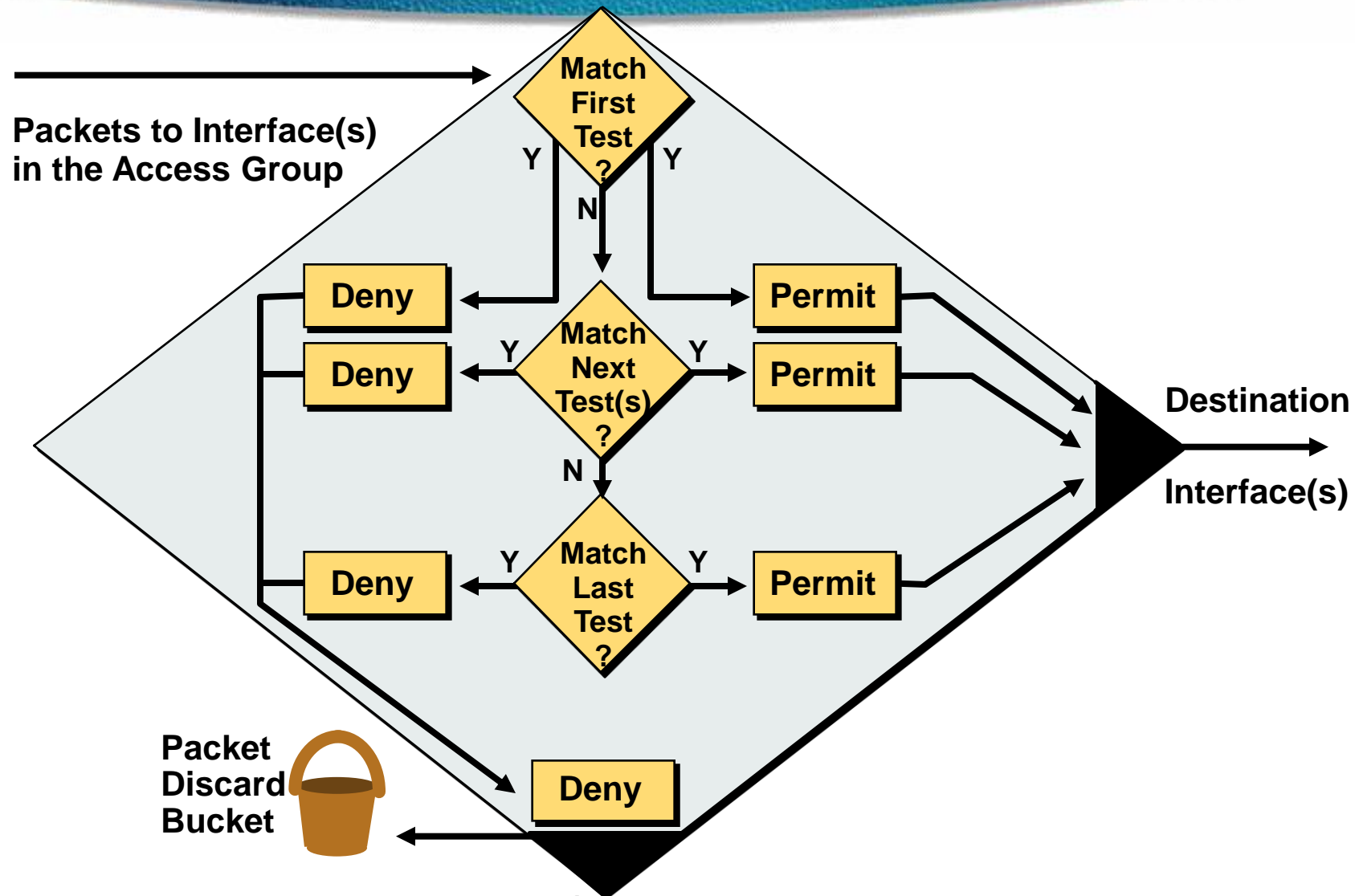
A List of Tests: Deny or Permit



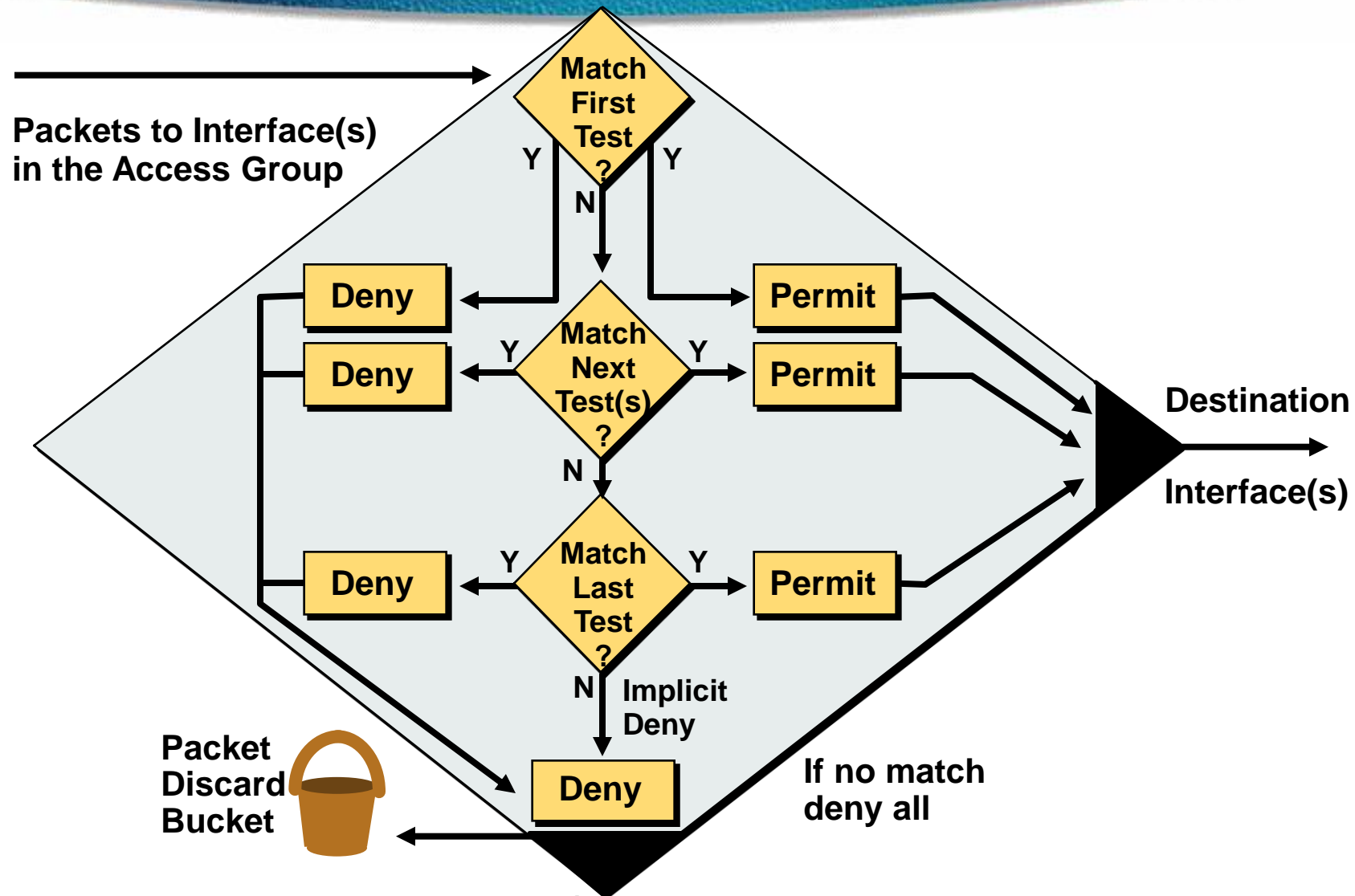
A List of Tests: Deny or Permit



A List of Tests: Deny or Permit



A List of Tests: Deny or Permit



Access List Configuration Guidelines

- Access list numbers indicate which protocol is filtered
- One access list per interface, per protocol, per direction
- The order of access list statements controls testing
- Most restrictive statements should be at the top of list
- There is an implicit deny any as the last access list test—every list should have at least one permit statement
- Create access lists before applying them to interfaces
- Access list, filter traffic going through the router; they do not apply to traffic originated from the router

Access List Command Overview

Step 1: Set parameters for this access list test statement (which can be one of several statements)

Router(config)#

```
access-list access-list-number { permit | deny } { test conditions }
```


Access List Command Overview

Step 1: Set parameters for this access list test statement (which can be one of several statements)

Router(config)#

```
access-list access-list-number { permit | deny } { test conditions }
```

Step 2: Enable an interface to use the specified access list

Router(config-if)#

```
{ protocol } access-group access-list-number {in / out}
```

IP Access lists are numbered 1-99 or 100-199

How to Identify Access Lists

Access List Type		Number Range/Identifier
IP	Standard	1-99

- **Standard IP lists (1 to 99) test conditions of all IP packets from source addresses**

How to Identify Access Lists

Access List Type		Number Range/Identifier
IP	Standard	1-99
	Extended	100-199

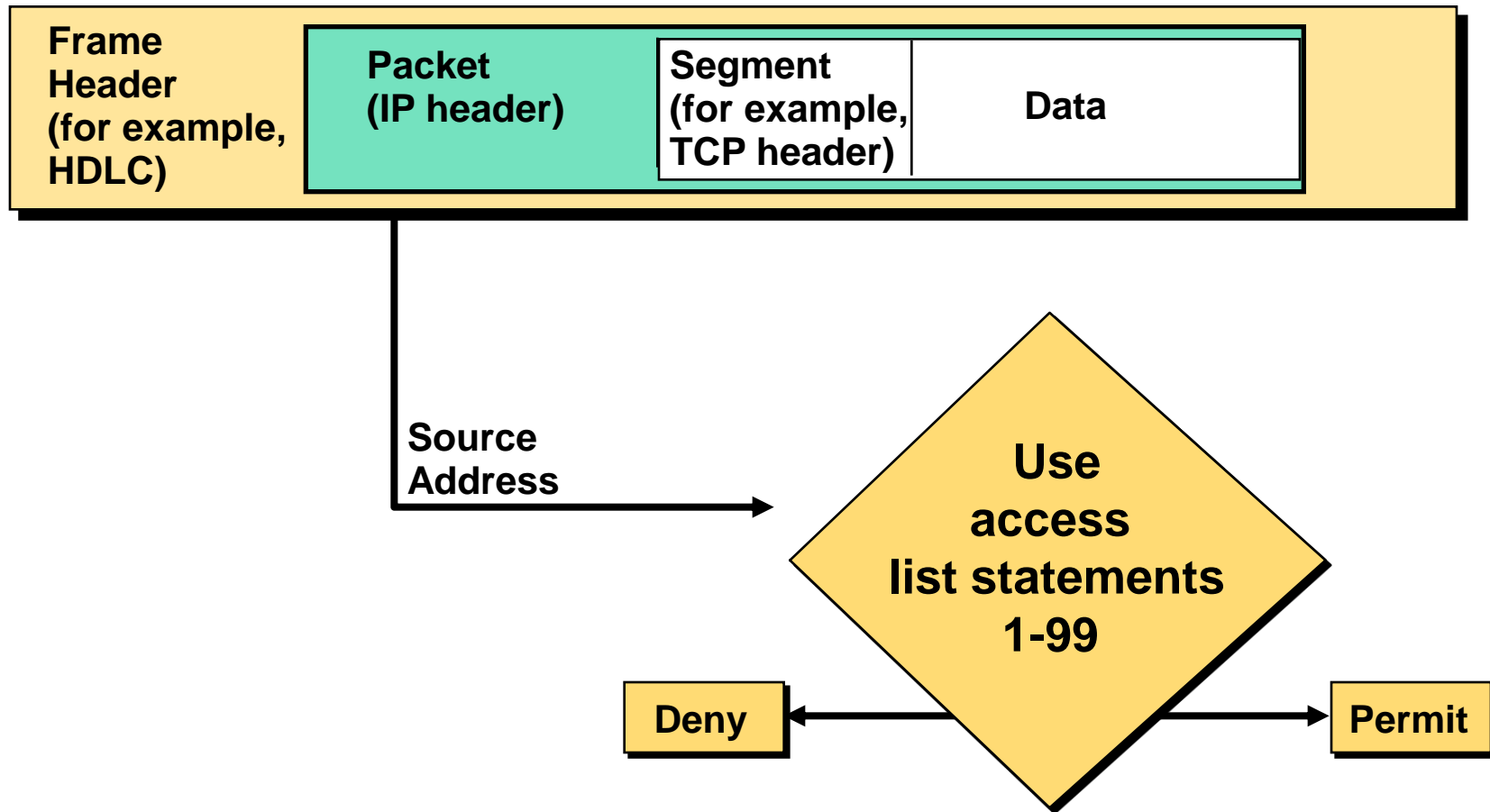
- **Standard IP lists (1 to 99) test conditions of all IP packets from source addresses**
- **Extended IP lists (100 to 199) can test conditions of source and destination addresses, specific TCP/IP protocols, and destination ports**

How to Identify Access Lists

Access List Type		Number Range/Identifier
IP	Standard	1-99
	Extended	100-199, 1300-1999, 2000-2699
	Named	Name (Cisco IOS 11.2 and later)
IPX	Standard	800-899
	Extended	900-999
	SAP filters	1000-1099
	Named	Name (Cisco IOS 11.2. F and later)

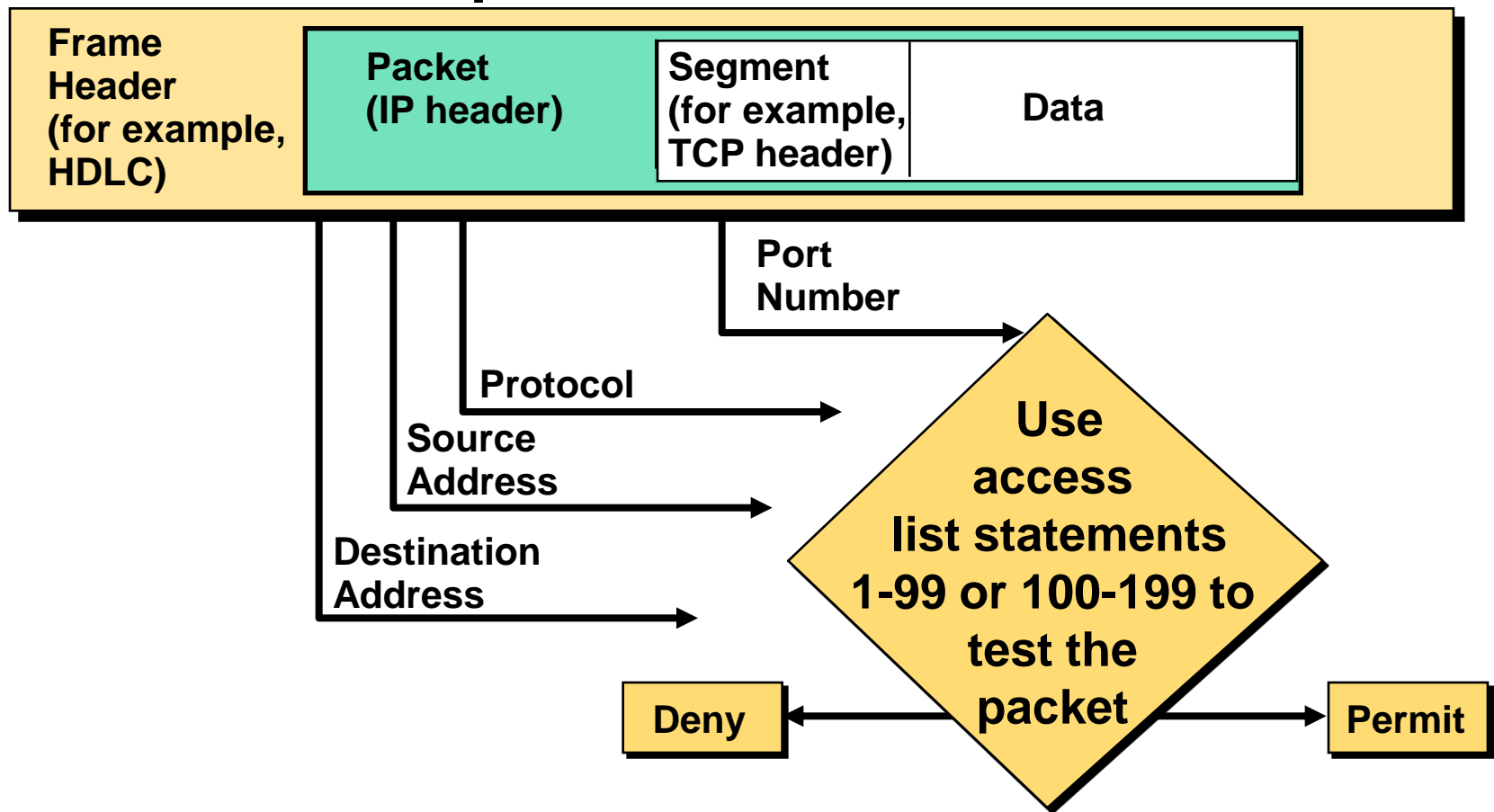
- **Standard IP lists (1 to 99) test conditions of all IP packets from source addresses**
- **Extended IP lists (100 to 199) can test conditions of source and destination addresses, specific TCP/IP protocols, and destination ports**
- **Other access list number ranges test conditions for other networking protocols**

Testing Packets with Standard Access Lists











Testing Packets with Extended Access Lists

An Example from a TCP/IP Packet



Wildcard Bits: How to Check the Corresponding Address Bits

128	64	32	16	8	4	2	1	Octet bit position and address value for bit	
									
0	0	0	0	0	0	0	0	=	check all address bits (match all)
0	0	1	1	1	1	1	1	=	ignore last 6 address bits
0	0	0	0	1	1	1	1	=	ignore last 4 address bits
1	1	1	1	1	1	0	0	=	check last 2 address bits
1	1	1	1	1	1	1	1	=	do not check address (ignore bits in octet)

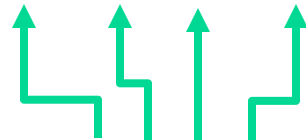
- 0 means check corresponding address bit value
- 1 means ignore value of corresponding address bit

Wildcard Bits to Match a Specific IP Host Address

Test conditions: Check all the address bits (match all)

An IP host address, for example:

172.30.16.29



Wildcard mask: 0.0.0.0
(checks all bits)

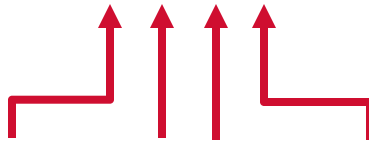
- **Example 172.30.16.29 0.0.0.0 checks all the address bits**
- **Abbreviate this wildcard mask using the IP address preceded by the keyword *host* (*host 172.30.16.29*)**

Wildcard Bits to Match Any IP Address

Test conditions: Ignore all the address bits (match any)

Any IP address

0.0.0.0



Wildcard mask: 255.255.255.255
(ignore all)

- Accept any address: **0.0.0.0 255.255.255.255**
- Abbreviate the expression using the keyword *any*

Wildcard Bits to Match IP Subnets

Check for IP subnets 172.30.16.0/24 to 172.30.31.0/24

Address and wildcard mask:

172.30.16.0 0.0.15.255

Network .host

172.30.16.0

Wildcard mask:

0	0	0	1	0	0	0	0		
0	0	0	0	1	1	1	1		
<---- match ---->				<---- don't care ---->					
0	0	0	1	0	0	0	0	=	16
0	0	0	1	0	0	0	1	=	17
0	0	0	1	0	0	1	0	=	18
			:						:
0	0	0	1	1	1	1	1	=	31



Configuring Standard IP Access Lists

Standard IP Access List Configuration

Router(config)#

```
access-list access-list-number {permit|deny} source [mask]
```

- Sets parameters for this list entry
- IP standard access lists use 1 to 99
- Default wildcard mask = 0.0.0.0
- “no access-list *access-list-number*” removes entire access-list

Standard IP Access List Configuration

Router(config)#

```
access-list access-list-number {permit|deny} source [mask]
```

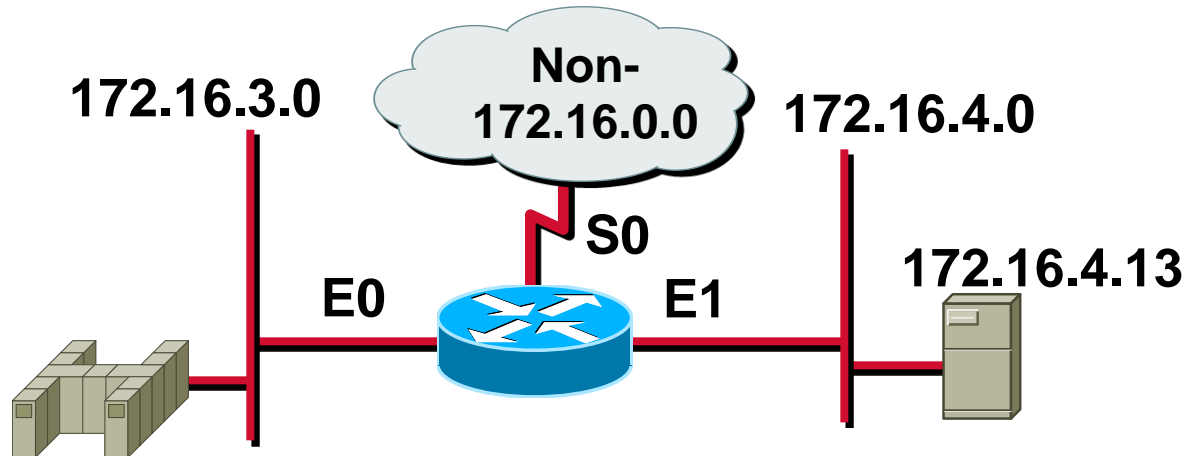
- Sets parameters for this list entry
- IP standard access lists use 1 to 99
- Default wildcard mask = 0.0.0.0
- “no access-list *access-list-number*” removes entire access-list

Router(config-if)#

```
ip access-group access-list-number { in | out }
```

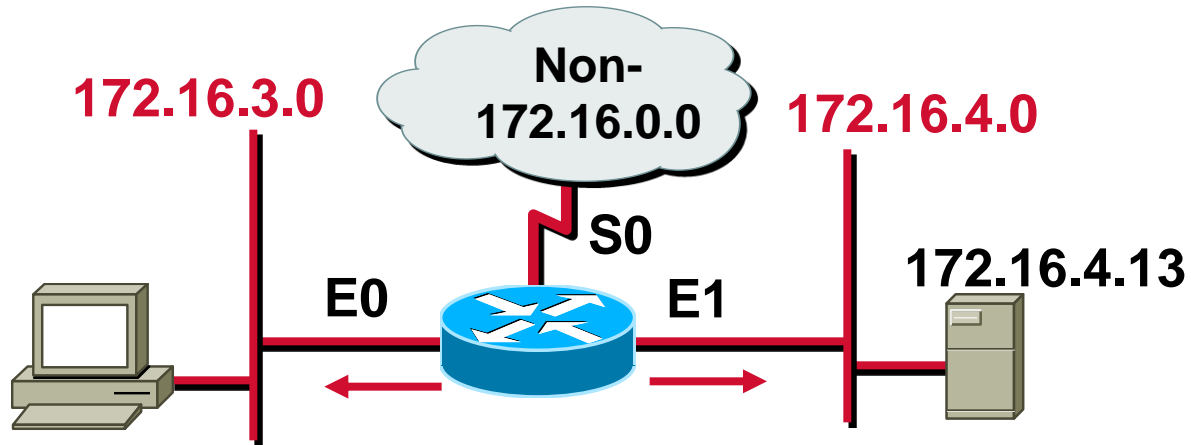
- Activates the list on an interface
- Sets inbound or outbound testing
- Default = Outbound
- “no ip access-group *access-list-number*” removes access-list from the interface

Standard IP Access List Example 1



```
access-list 1 permit 172.16.0.0 0.0.255.255  
(implicit deny all - not visible in the list)  
(access-list 1 deny 0.0.0.0 255.255.255.255)
```


Standard IP Access List Example 1

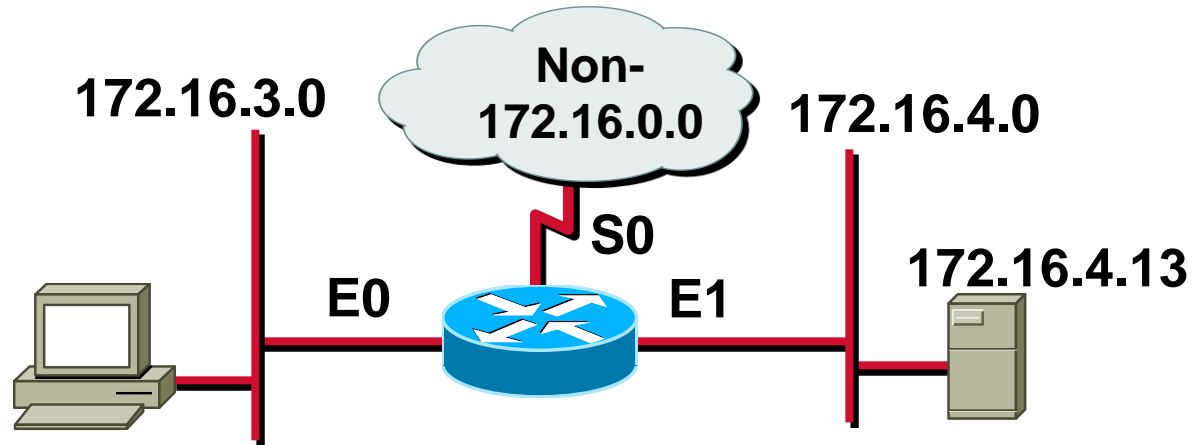


```
access-list 1 permit 172.16.0.0 0.0.255.255  
(implicit deny all - not visible in the list)  
(access-list 1 deny 0.0.0.0 255.255.255.255)
```

```
interface ethernet 0  
ip access-group 1 out  
interface ethernet 1  
ip access-group 1 out
```

- **Permit my network only**

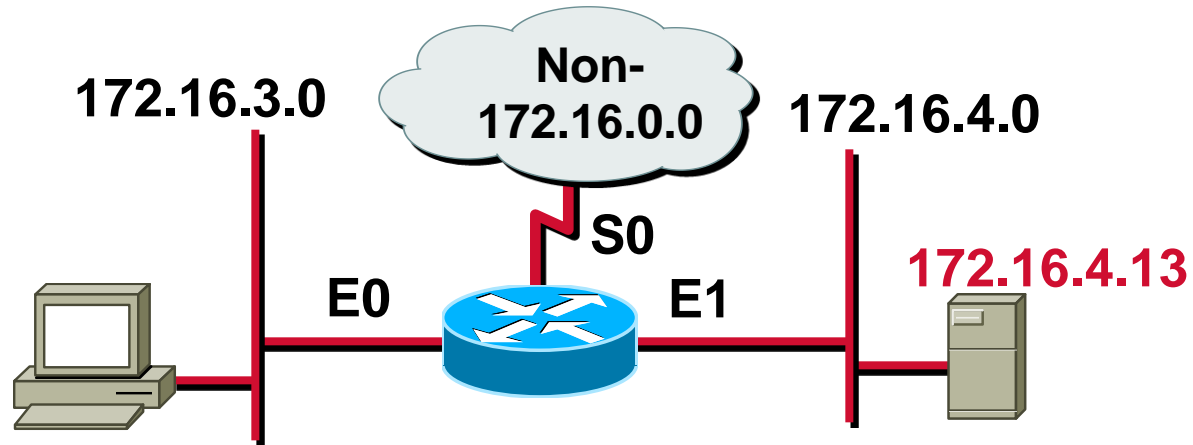
Standard IP Access List Example 2



```
access-list 1 deny 172.16.4.13 0.0.0.0
```

Deny a specific host

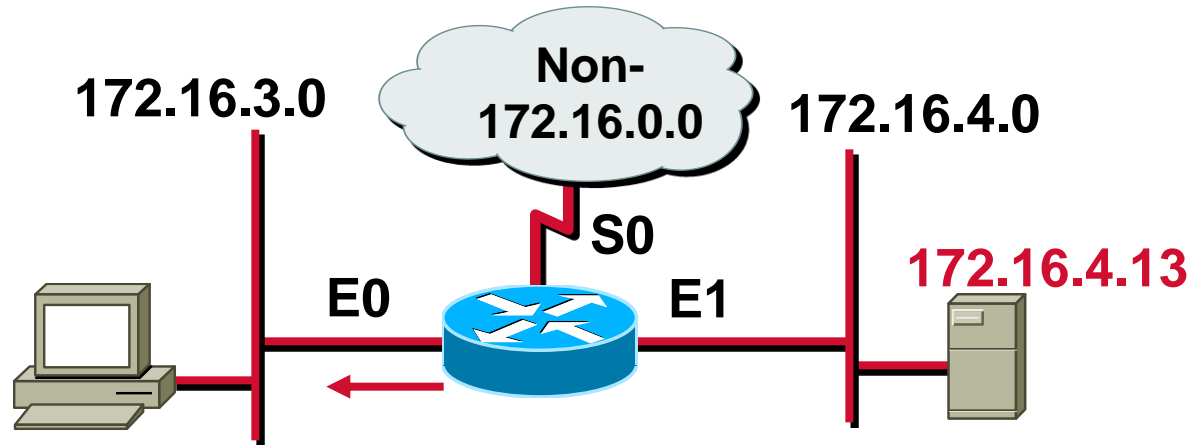
Standard IP Access List Example 2



```
access-list 1 deny 172.16.4.13 0.0.0.0
access-list 1 permit 0.0.0.0 255.255.255.255
(implicit deny all)
(access-list 1 deny 0.0.0.0 255.255.255.255)
```

Deny a specific host

Standard IP Access List Example 2

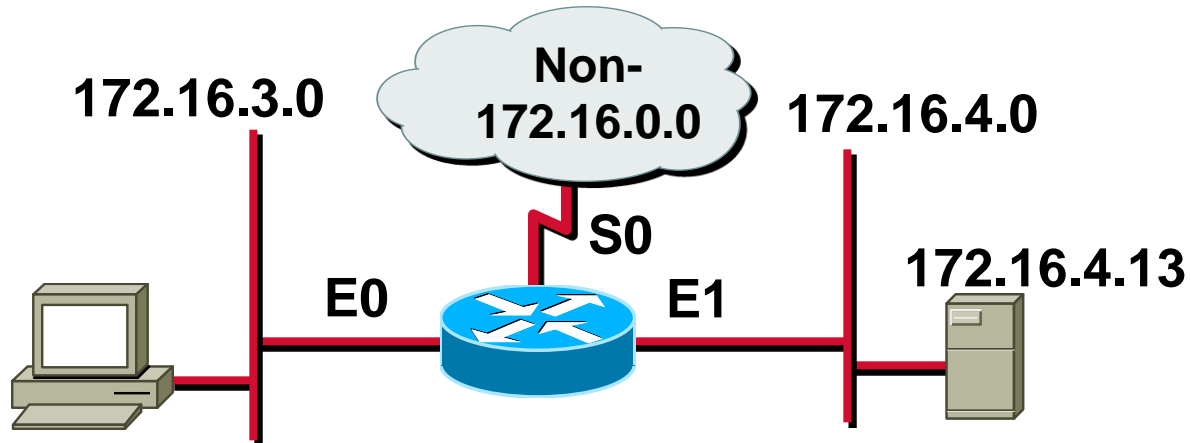


```
access-list 1 deny 172.16.4.13 0.0.0.0
access-list 1 permit 0.0.0.0 255.255.255.255
(implicit deny all)
(access-list 1 deny 0.0.0.0 255.255.255.255)

interface ethernet 0
ip access-group 1 out
```

- Deny a specific host

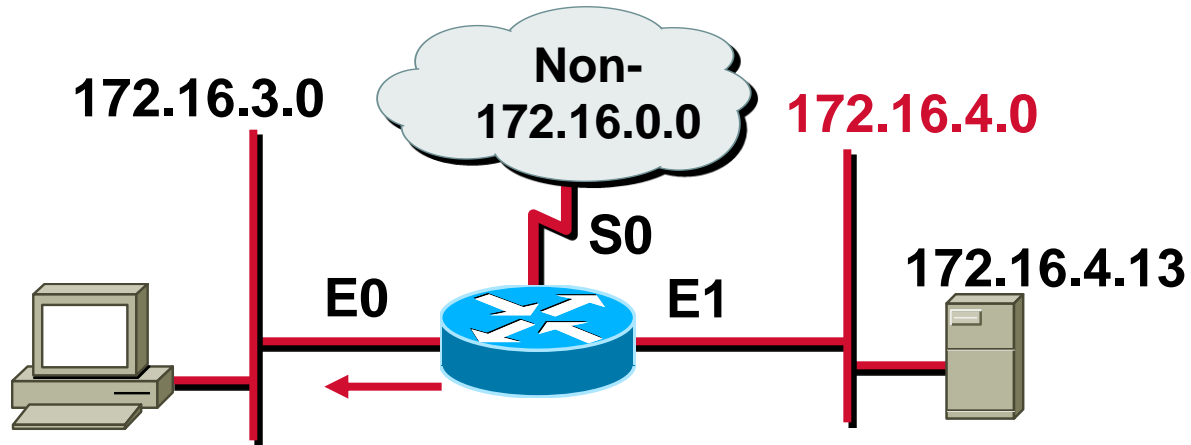
Standard IP Access List Example 3



```
access-list 1 deny 172.16.4.0 0.0.0.255
access-list 1 permit any
(implicit deny all)
(access-list 1 deny 0.0.0.0 255.255.255.255)
```

Deny a specific subnet

Standard IP Access List Example 3



```
access-list 1 deny 172.16.4.0 0.0.0.255
access-list 1 permit any
(implicit deny all)
(access-list 1 deny 0.0.0.0 255.255.255.255)

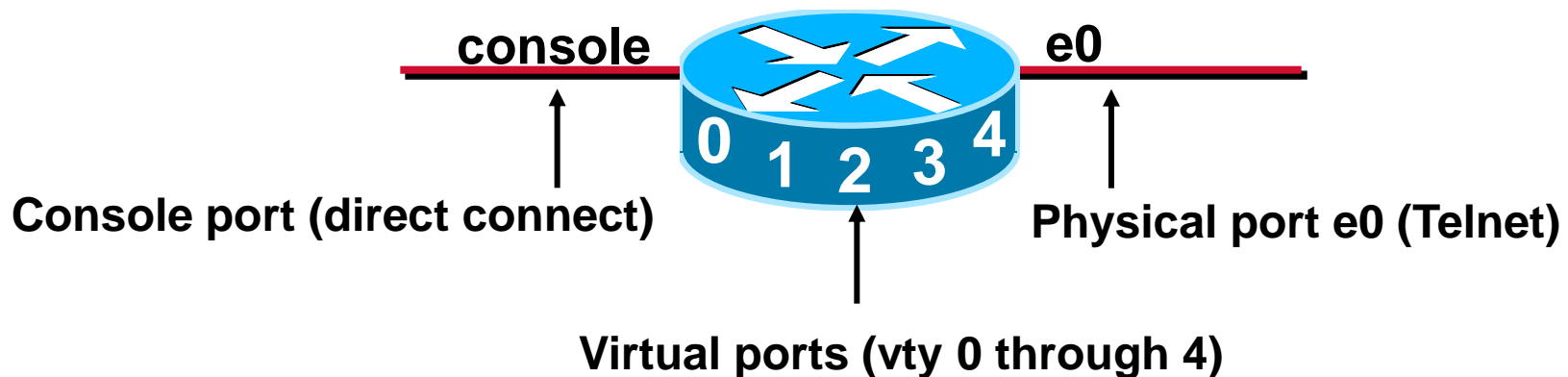
interface ethernet 0
ip access-group 1 out
```

- **Deny a specific subnet**



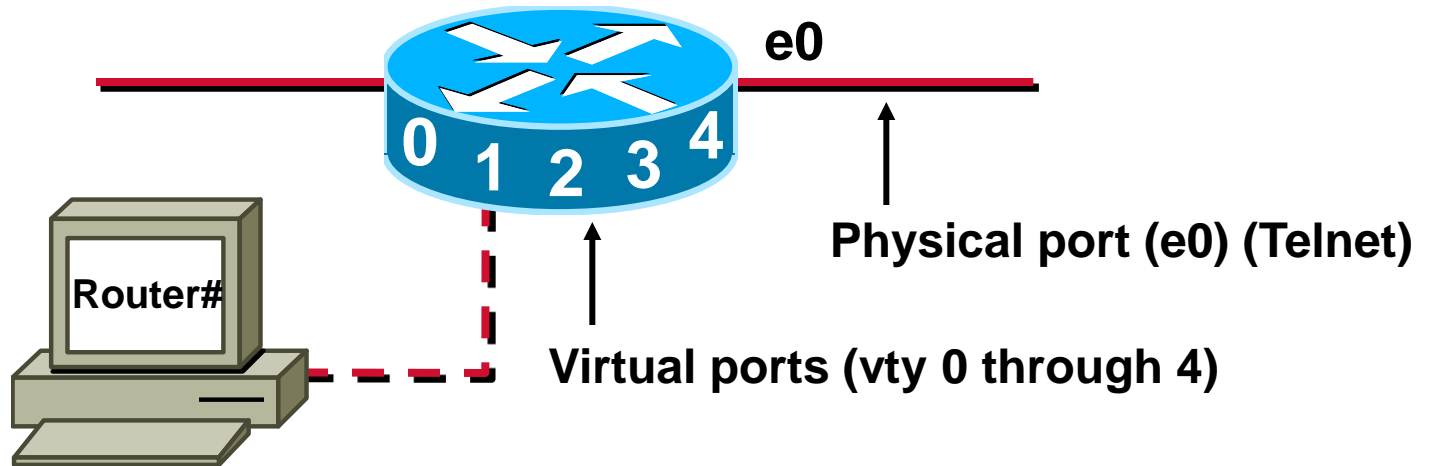
Control vty Access With Access Class

Filter Virtual Terminal (vty) Access to a Router



- Five virtual terminal lines (0 through 4)
- Filter addresses that can access into the router's vty ports
- Filter vty access out from the router

How to Control vty Access



- Setup IP address filter with standard access list statement
- Use line configuration mode to filter access with the *access-class* command
- Set identical restrictions on all vtys

Virtual Terminal Line Commands

Router(config)#

```
line vty{vty# | vty-range}
```

- Enters configuration mode for a vty or vty range

Router(config-line)#

```
access-class access-list-number {in|out}
```

- Restricts incoming or outgoing vty connections for address in the access list

Virtual Terminal Access Example

Controlling Inbound Access

```
access-list 12 permit 192.89.55.0 0.0.0.255
!  
line vty 0 4  
  access-class 12 in
```

- **Permits only hosts in network 192.89.55.0 to connect to the router's vtys**



Configuring Extended IP Access Lists

Standard versus External Access List

Standard	Extended
Filters Based on Source.	Filters Based on Source and destination.
Permit or deny entire TCP/IP protocol suite.	Specifies a specific IP protocol and port number.
Range is 1 through 99	Range is 100 through 199.

Extended IP Access List Configuration

Router(config)#

```
access-list access-list-number { permit | deny } protocol source  
source-wildcard [operator port] destination destination-wildcard  
[ operator port ] [ established ] [log]
```

- Sets parameters for this list entry

Extended IP Access List Configuration

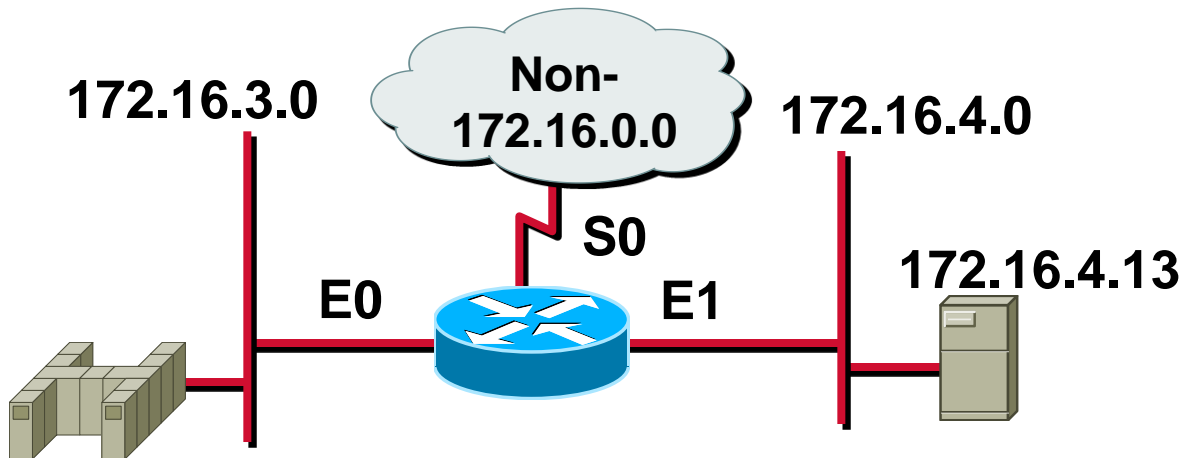
```
Router(config)# access-list access-list-number  
{ permit | deny } protocol source source-wildcard [operator  
port] destination destination-wildcard [ operator port ] [ established ] [log]
```

- **Sets parameters for this list entry**

```
Router(config-if)# ip access-group access-list-number { in  
| out }
```

- **Activates the extended list on an interface**

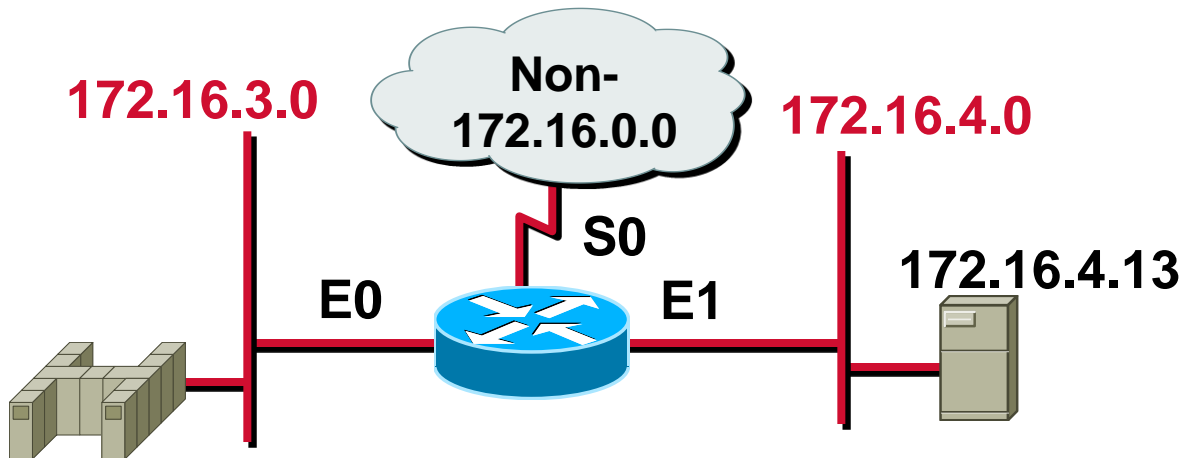
Extended Access List Example 1



```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
```

- Deny FTP from subnet 172.16.4.0 to subnet 172.16.3.0 out of E0
- Permit all other traffic

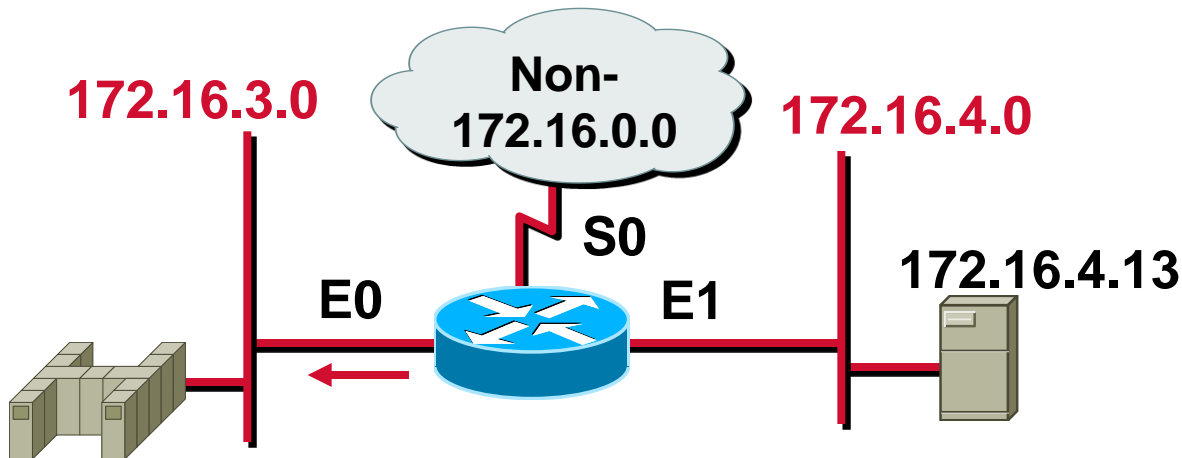
Extended Access List Example 1



```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
access-list 101 permit ip any any
(implicit deny all)
(access-list 101 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255)
```

- Deny FTP from subnet 172.16.4.0 to subnet 172.16.3.0 out of E0
- Permit all other traffic

Extended Access List Example 1

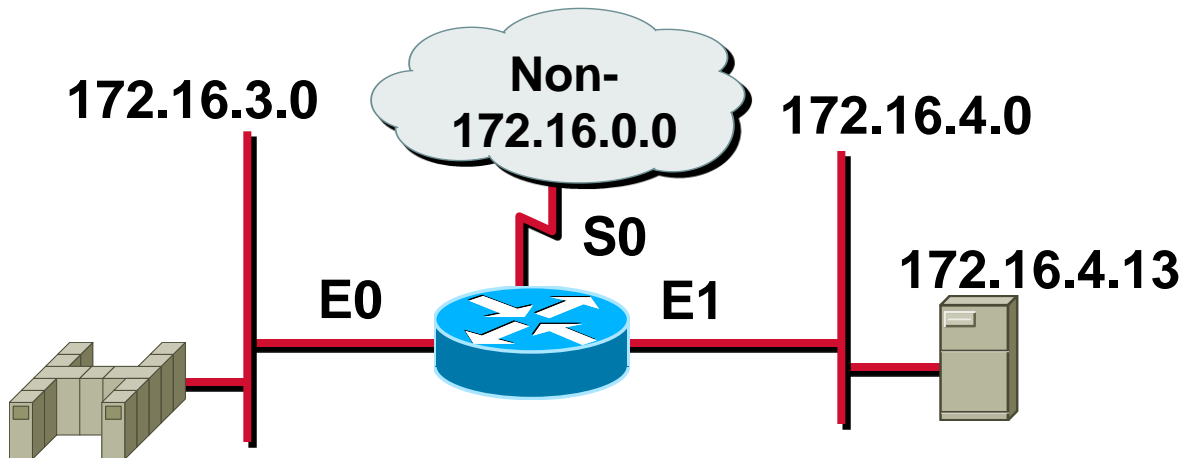


```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
access-list 101 permit ip any any
(implicit deny all)
(access-list 101 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255)

interface ethernet 0
ip access-group 101 out
```

- Deny FTP from subnet 172.16.4.0 to subnet 172.16.3.0 out of E0
- Permit all other traffic

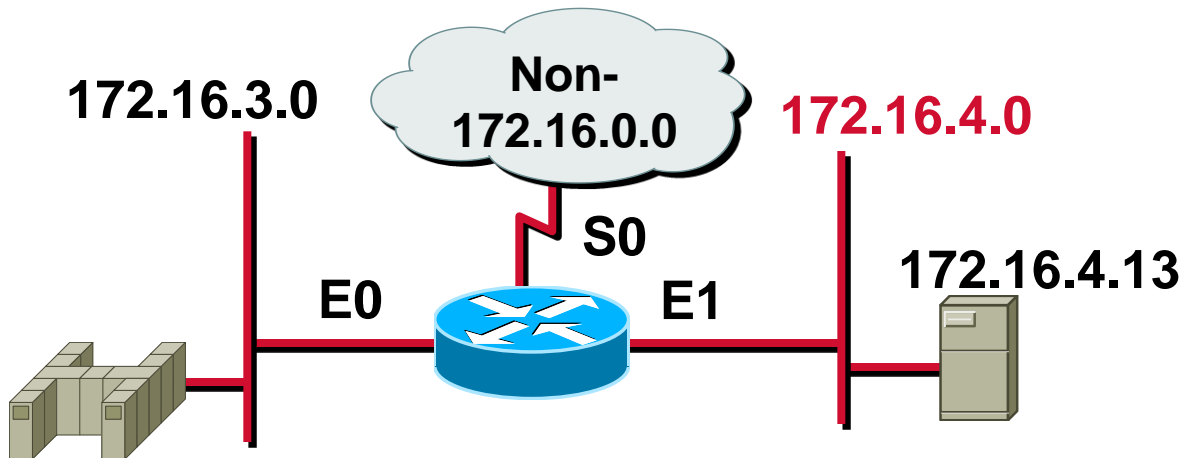
Extended Access List Example 2



```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 any eq 23
```

- Deny only Telnet from subnet 172.16.4.0 out of E0
- Permit all other traffic

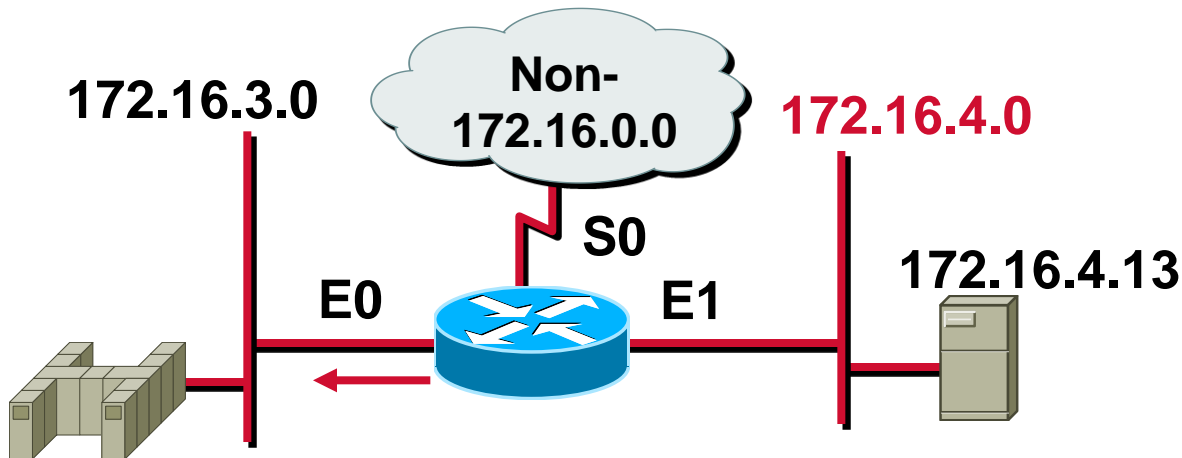
Extended Access List Example 2



```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 any eq 23
access-list 101 permit ip any any
(implicit deny all)
```

- Deny only Telnet from subnet 172.16.4.0 out of E0
- Permit all other traffic

Extended Access List Example 2



```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 any eq 23
access-list 101 permit ip any any
(implicit deny all)
```

```
interface ethernet 0
ip access-group 101 out
```

- Deny only Telnet from subnet 172.16.4.0 out of E0
- Permit all other traffic

Using Named IP Access Lists

- **Feature for Cisco IOS Release 11.2 or later**

Router(config)#

```
ip access-list { standard | extended } name
```

- **Alphanumeric name string must be unique**

Using Named IP Access Lists

- **Feature for Cisco IOS Release 11.2 or later**

Router(config)#

```
ip access-list { standard | extended } name
```

- **Alphanumeric name string must be unique**

Router(config {std- | ext-}nacl)#

```
{ permit | deny } { ip access list test conditions }  
{ permit | deny } { ip access list test conditions }  
no { permit | deny } { ip access list test conditions }
```

- **Permit or deny statements have no prepended number**
- **"no" removes the specific test from the named access list**

Using Named IP Access Lists

- **Feature for Cisco IOS Release 11.2 or later**

```
Router(config)# ip access-list { standard | extended } name
```

- **Alphanumeric name string must be unique**

```
Router(config {std- | ext-}nacl)# { permit | deny }  
{ ip access list test conditions }  
{ permit | deny } { ip access list test conditions }  
no { permit | deny } { ip access list test conditions }
```

- **Permit or deny statements have no prepended number**
- **"no" removes the specific test from the named access list**

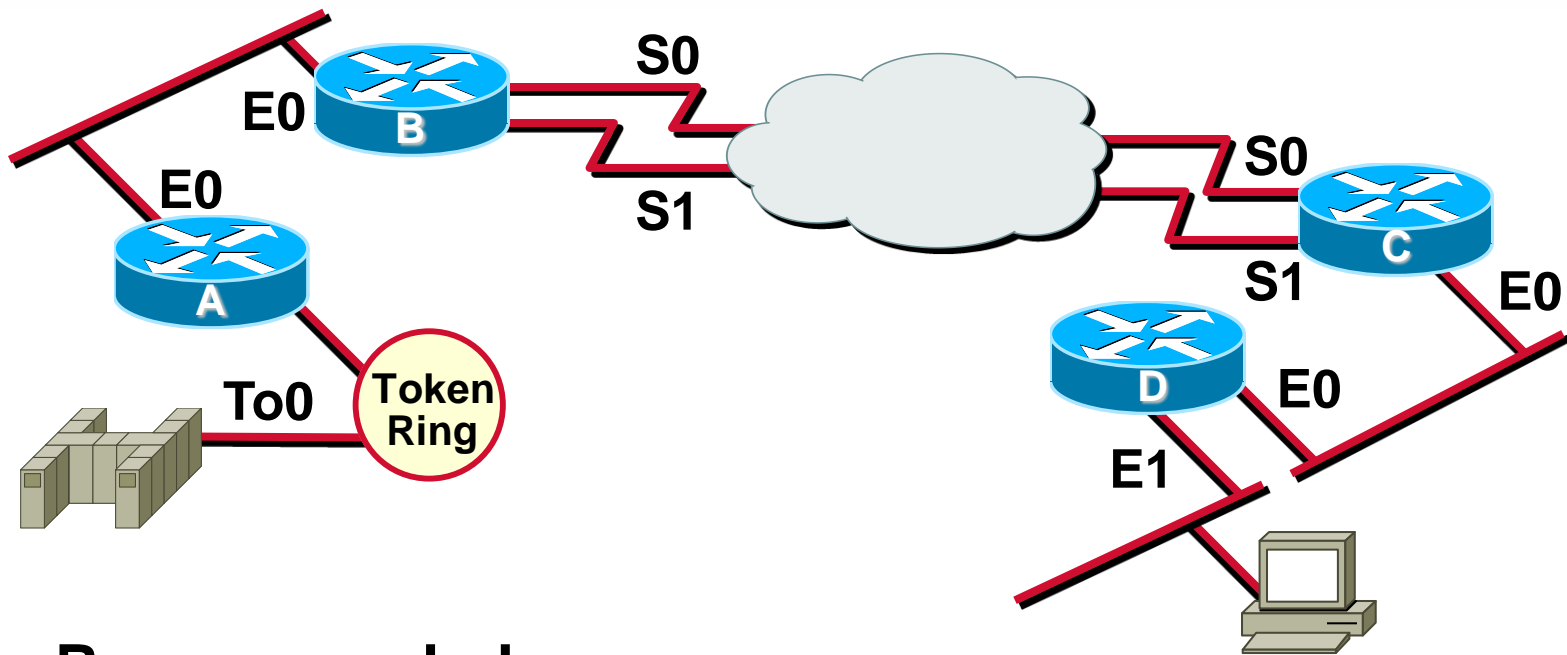
```
Router(config-if)# ip access-group name { in | out }
```

- **Activates the IP named access list on an interface**

Access List Configuration Principles

- **Order of access list statements is crucial**
Recommended: use a text editor on a TFTP server or use PC to cut and paste
- **Top-down processing**
Place more specific test statements first
- **No reordering or removal of statements**
Use no access-list *number* command to remove entire access list
Exception: Named access lists permit removal of individual statements
- **Implicit deny all**
Unless access list ends with explicit permit any

Where to Place IP Access Lists



Recommended:

- Place extended access lists close to the source
- Place standard access lists close to the destination

Verifying Access Lists

```
wg_ro_a#show ip int e0
Ethernet0 is up, line protocol is up
  Internet address is 10.1.1.11/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound access list is 1
  Proxy ARP is enabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachable are always sent
  ICMP mask replies are never sent
  IP fast switching is enabled
  IP fast switching on the same interface is disabled
  IP Feature Fast switching turbo vector
  IP multicast fast switching is enabled
  IP multicast distributed fast switching is disabled
<text ommitted>
```

Monitoring Access List Statements

```
wg_ro_a#show {protocol} access-list {access-list number}
```

```
wg_ro_a#show access-lists {access-list number}
```

```
wg_ro_a#show access-lists
```

```
Standard IP access list 1
```

```
    permit 10.2.2.1
```

```
    permit 10.3.3.1
```

```
    permit 10.4.4.1
```

```
    permit 10.5.5.1
```

```
Extended IP access list 101
```

```
    permit tcp host 10.22.22.1 any eq telnet
```

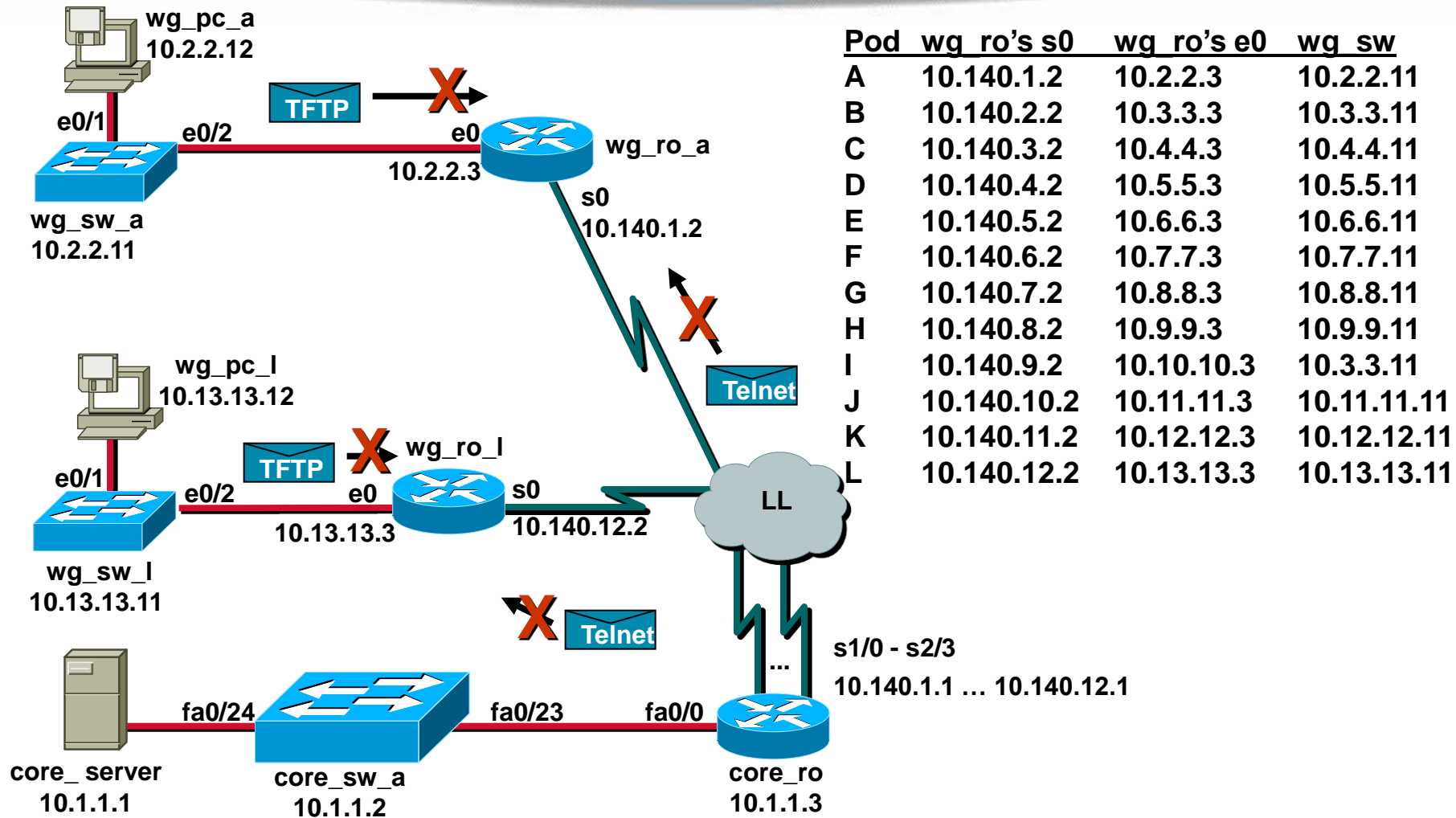
```
    permit tcp host 10.33.33.1 any eq ftp
```

```
    permit tcp host 10.44.44.1 any eq ftp-data
```



Laboratory Exercise

Visual Objective



Summary

After completing this chapter, you should be able to perform the following tasks:

- **Identify the key functions and processing of IP access lists**
- **Configure standard IP access lists**
- **Control vty access with an access class**
- **Configure extended IP access lists**
- **Verify and monitor IP access lists**

Review Questions

- 1. What are the two types of IP access lists?**
- 2. What is the last statement in all access lists?**
- 3. What command do you use to apply an access list to a vty port?**