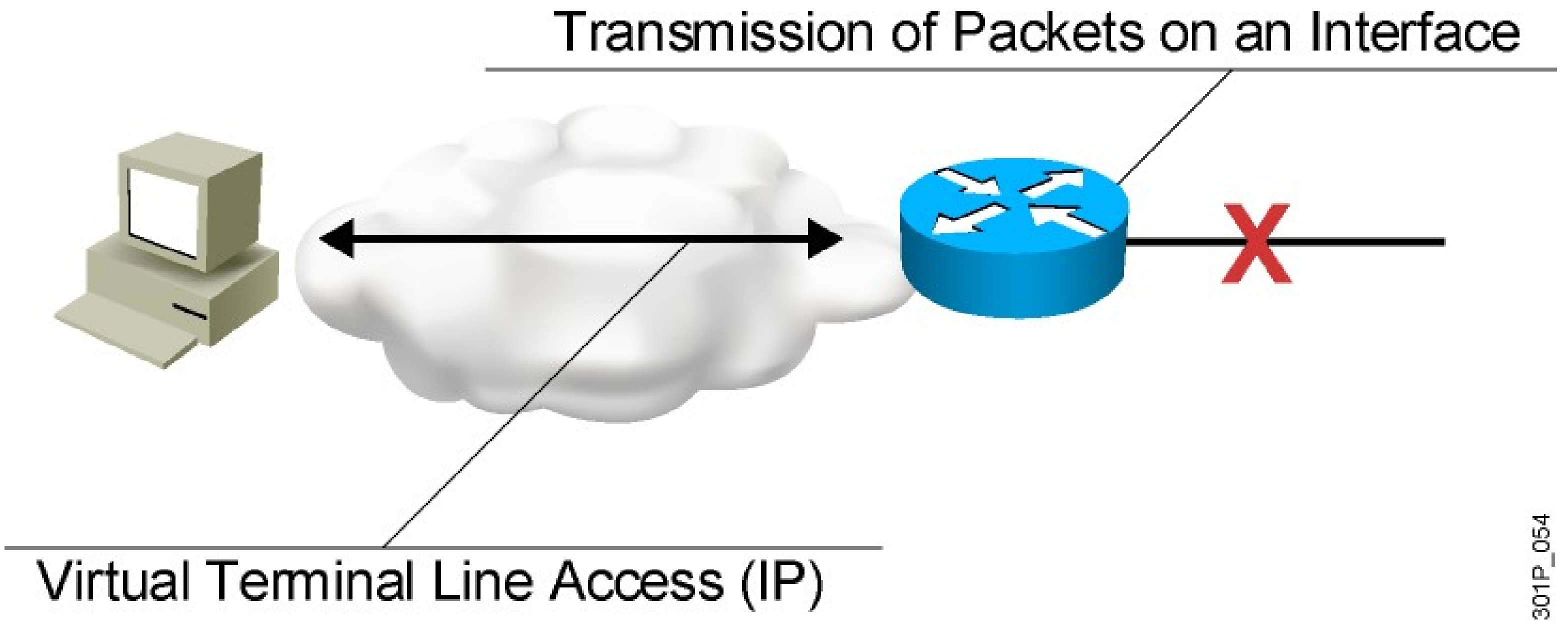
Cisco Networking Academy
Mind Wide Open

Chapter 4: Access Control Lists

Libor Polčák

Vysoké učení technické v Brně, Fakulta informačních technologií
Božetěchova 1/2, 612 66 Brno
ipolcak@fit.vutbr.cz

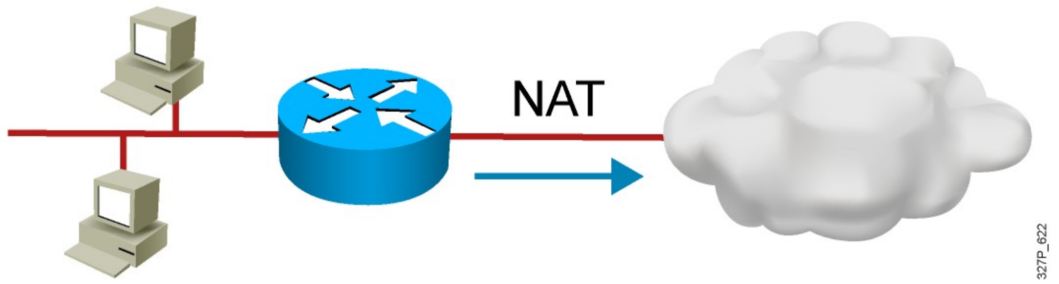
ACL Applications: Filtering


Transmission of Packets on an Interface
Virtual Terminal Line Access (IP)
301P_054

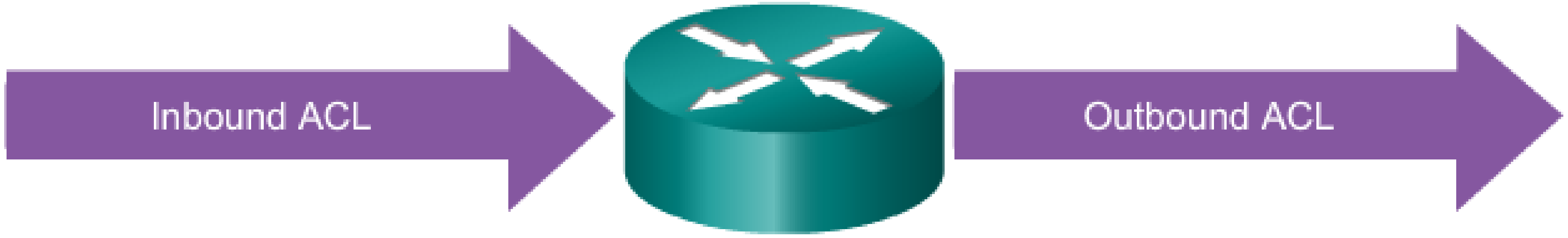
- Permit or deny packets moving through the router.
- Permit or deny vty access to or from the router.
- Without ACLs, all packets could be transmitted to all parts of your network.

ACL Applications: Classification

Special handling for traffic based on packet tests



Purpose of ACLs
ACL Operation

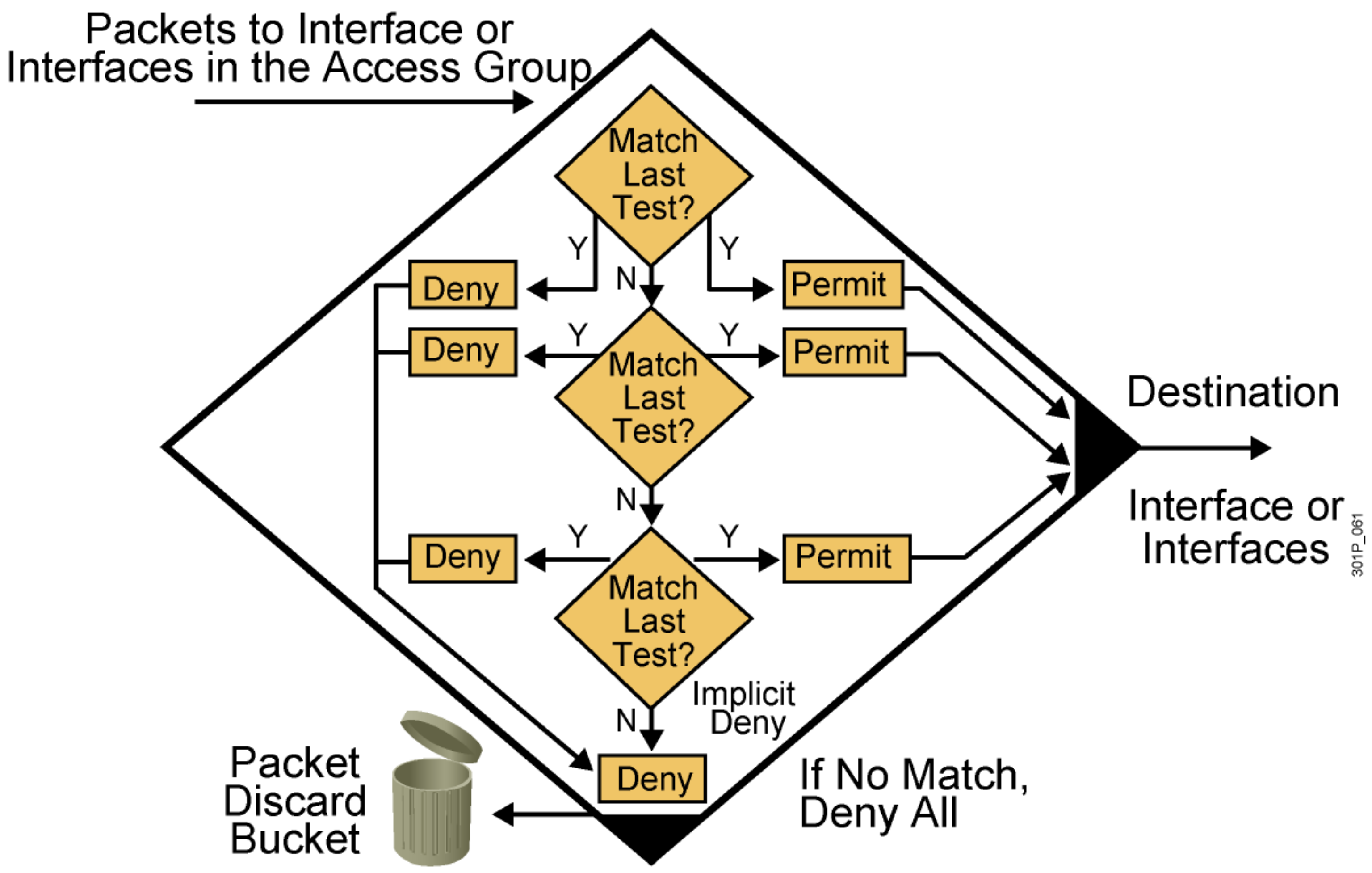


An inbound ACL filters packets coming into a specific interface and before they are routed to the outbound interface.

An outbound ACL filters packets after being routed, regardless of the inbound interface.

The last statement of an ACL is always an implicit deny. This statement is automatically inserted at the end of each ACL even though it is not physically present. The implicit deny blocks all traffic. Because of this *implicit deny, an ACL that does not have at least one permit statement will block all traffic.*

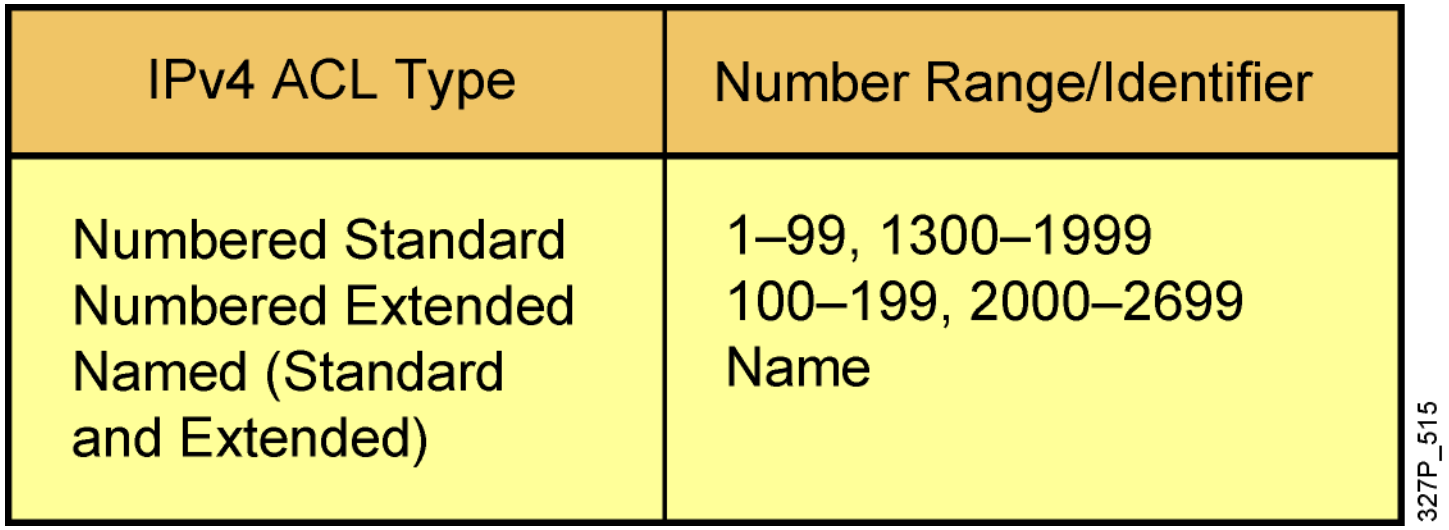
A List of Tests: Deny or Permit



Types of ACLs

- **Standard ACL**
  - Checks source address
  - Generally permits or denies entire protocol suite
- **Extended ACL**
  - Checks source and destination address
  - Generally permits or denies specific protocols and applications
- Two methods used to identify standard and extended ACLs:
  - Numbered ACLs use a number for identification
  - Named ACLs use a descriptive name or number for identification

How to Identify ACLs



IPv4 ACL Type	Number Range/Identifier
Numbered Standard	1–99, 1300–1999
Numbered Extended	100–199, 2000–2699
Named (Standard and Extended)	Name

- Numbered standard IPv4 lists (1–99) test conditions of all IP packets for source addresses. Expanded range (1300–1999).
- Numbered extended IPv4 lists (100–199) test conditions of source and destination addresses, specific TCP/IP protocols, and destination ports. Expanded range (2000–2699).
- Named ACLs identify IP standard and extended ACLs with an alphanumeric string (name).

Standard versus Extended IPv4 ACLs

Types of Cisco IPv4 ACLs

Standard ACLs

```
access-list 10 permit 192.168.30.0 0.0.0.255
```

Standard ACLs filter IP packets based on the source address only.

Extended ACLs

```
access-list 103 permit tcp 192.168.30.0 0.0.0.255 any eq 80
```

Extended ACLs filter IP packets based on several attributes, including the following:

- Source and destination IP addresses
- Source and destination TCP and UDP ports
- Protocol type/ Protocol number (example: IP, ICP, UDP, TCP, etc.)

IP Access List Entry Sequence Numbering

- Requires Cisco IOS Release 12.3
- Allows you to edit the order of ACL statements using sequence numbers
  - In software earlier than Cisco IOS Release 12.3, a text editor is used to create ACL statements, then the statements are copied into the router in the correct order.
- Allows you to remove a single ACL statement from the list using a sequence number
  - With named ACLs in software earlier than Cisco IOS Release 12.3, you must use **no {deny | permit} protocol source source-wildcard destination destination-wildcard** to remove an individual statement.
  - With numbered ACLs in software earlier than Cisco IOS Release 12.3, you must remove the entire ACL to remove a single ACL statement.

ACL Configuration Guidelines

- Standard or extended indicates what can be filtered.
- Only one ACL per interface, per network protocol, and per direction is allowed.
- The order of ACL statements controls testing, therefore, the most specific statements go at the top of the list.
- The last ACL test is always an implicit deny everything else statement, so every list needs at least one permit statement.
- ACLs are created globally and then applied to interfaces for inbound or outbound traffic.
- An ACL can filter traffic going through the router, or traffic to and from the router, depending on how it is applied.

Guidelines for ACL Placement

Where to Place ACLs

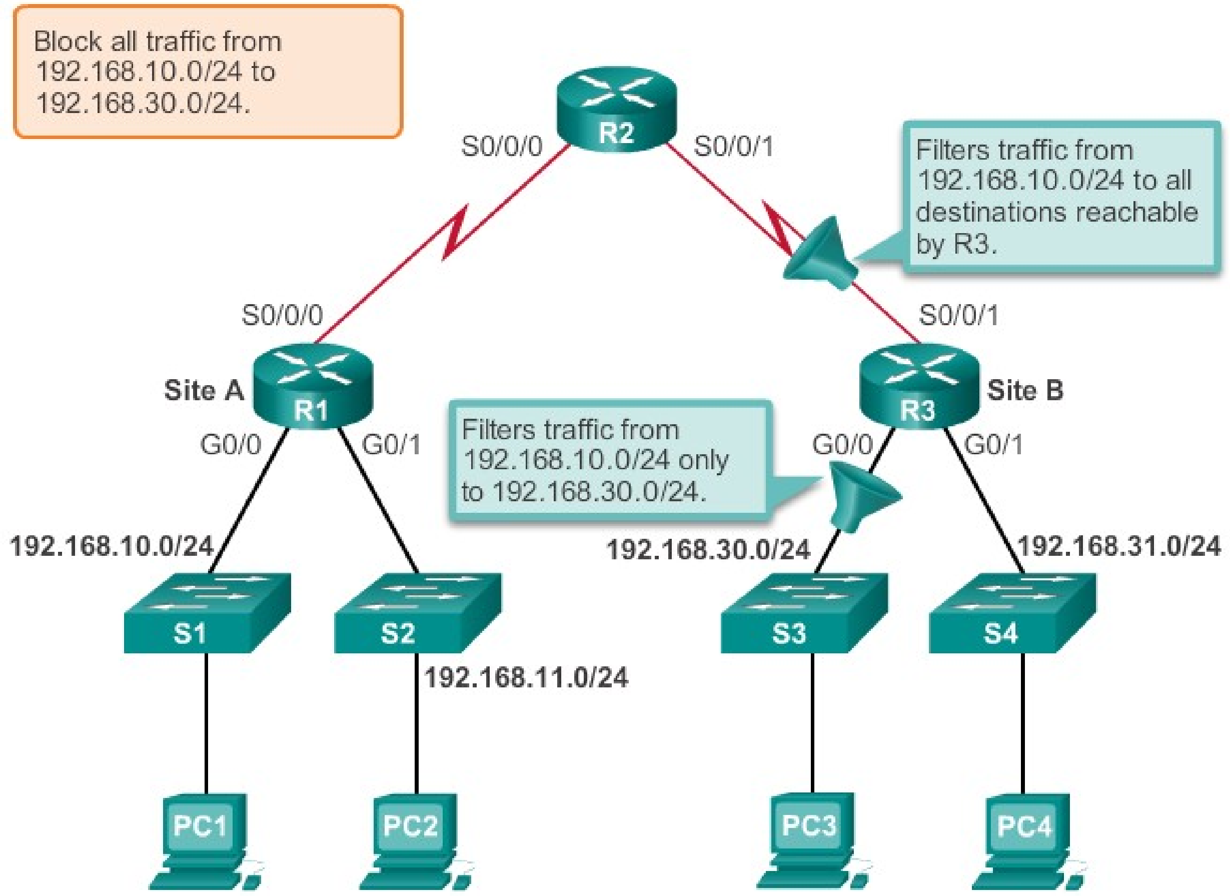
Every ACL should be placed where it has the greatest impact on efficiency. The basic rules are:

- Extended ACLs - Locate extended ACLs as close as possible to the source of the traffic to be filtered.
- Standard ACLs - Because standard ACLs do not specify destination addresses, place them as close to the destination as possible.

Placement of the ACL and therefore the type of ACL used may also depend on: the extent of the network administrator’s control, bandwidth of the networks involved, and ease of configuration.

Guidelines for ACL Placement

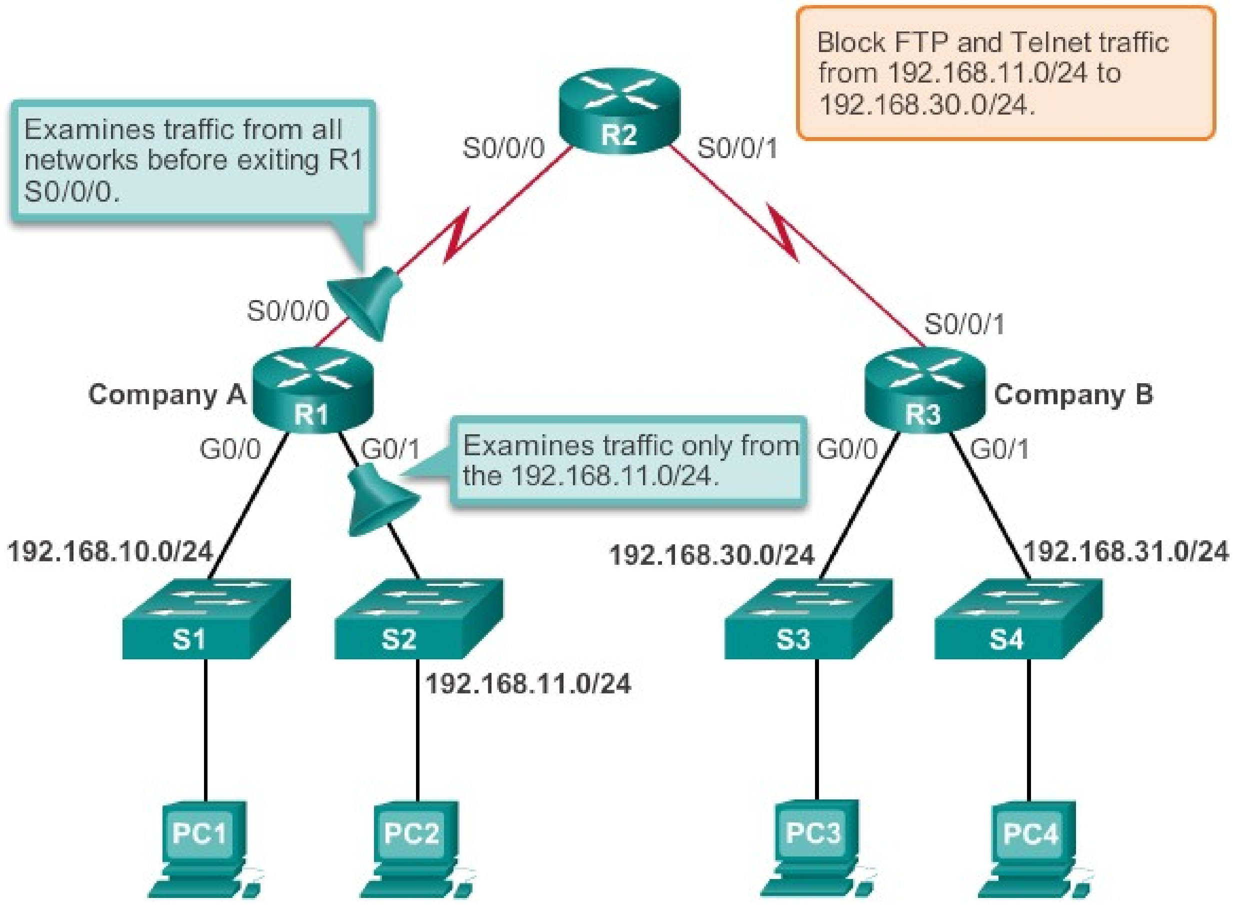
Standard ACL Placement



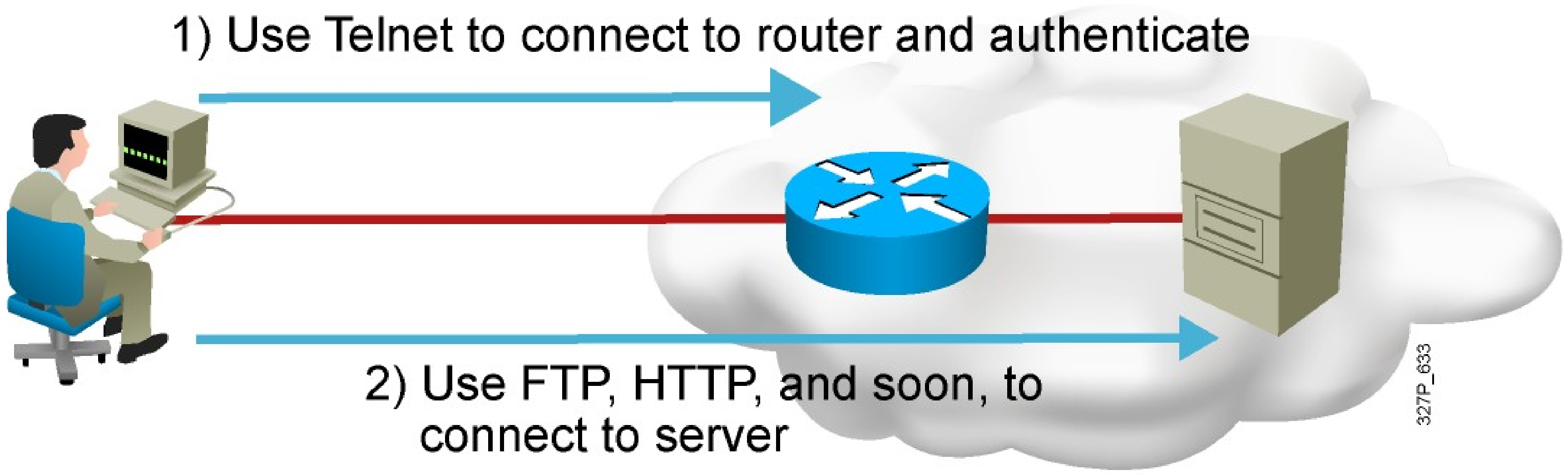
Block all traffic from 192.168.10.0/24 to 192.168.30.0/24.
R2
S0/0/0
S0/0/1
Filters traffic from 192.168.10.0/24 to all destinations reachable by R3.
S0/0/0
S0/0/1
Site A
R1
G0/0
G0/1
Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.
R3
Site B
G0/0
G0/1
192.168.10.0/24
192.168.30.0/24
192.168.31.0/24
S1
S2
S3
S4
192.168.11.0/24
PC1
PC2
PC3
PC4

Guidelines for ACL Placement

Extended ACL Placement

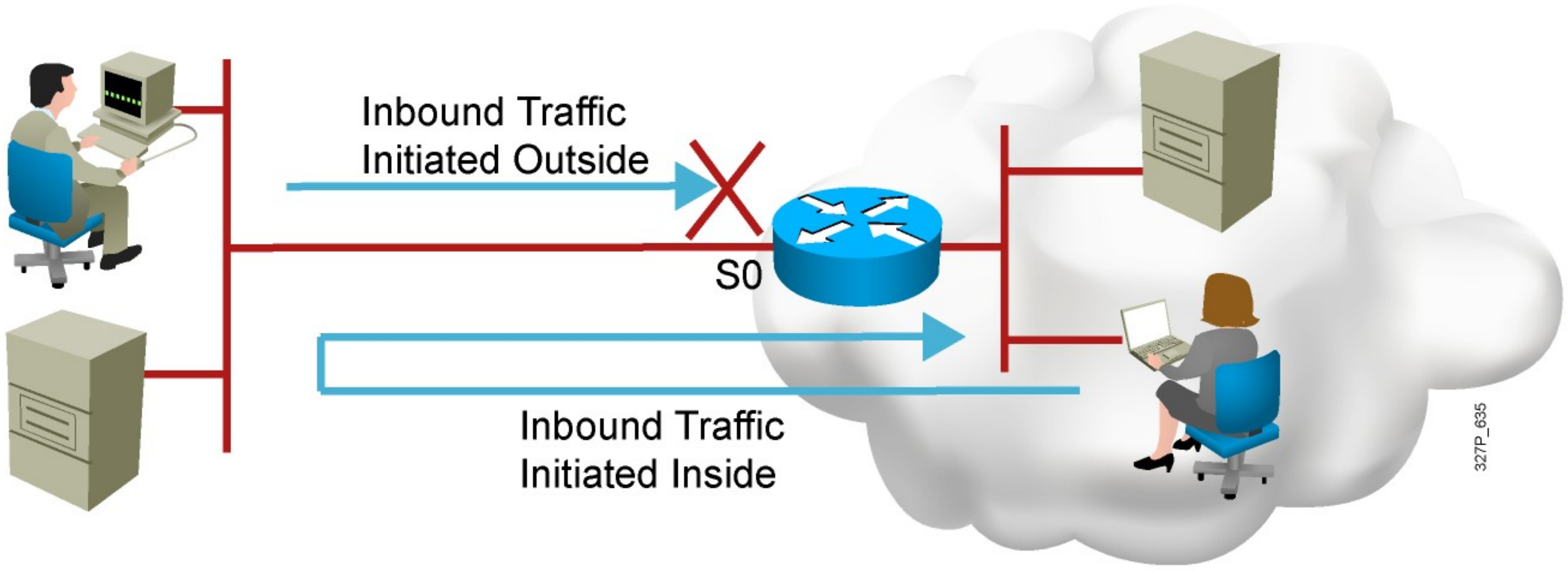

Block FTP and Telnet traffic from 192.168.11.0/24 to 192.168.30.0/24.
Examines traffic from all networks before exiting R1 S0/0/0.
Examines traffic only from the 192.168.11.0/24.
R2
S0/0/0
S0/0/1
S0/0/0
S0/0/1
Company A
R1
Company B
R3
G0/0
G0/1
G0/0
G0/1
192.168.10.0/24
192.168.30.0/24
192.168.31.0/24
S1
S2
S3
S4
192.168.11.0/24
PC1
PC2
PC3
PC4

Dynamic ACLs


1) Use Telnet to connect to router and authenticate
2) Use FTP, HTTP, and soon, to connect to server
327P_633

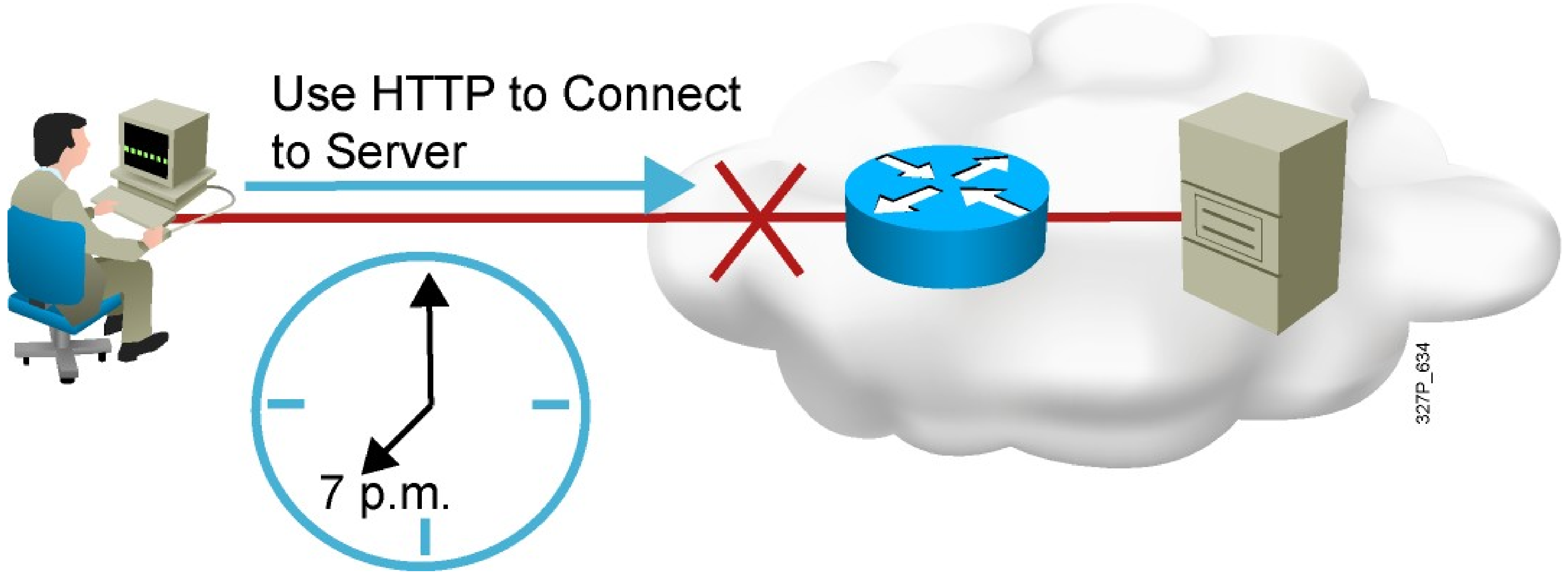
Dynamic ACLs (lock-and-key): Users that want to traverse the router are blocked until they use Telnet to connect to the router and are authenticated.

Reflexive ACLs



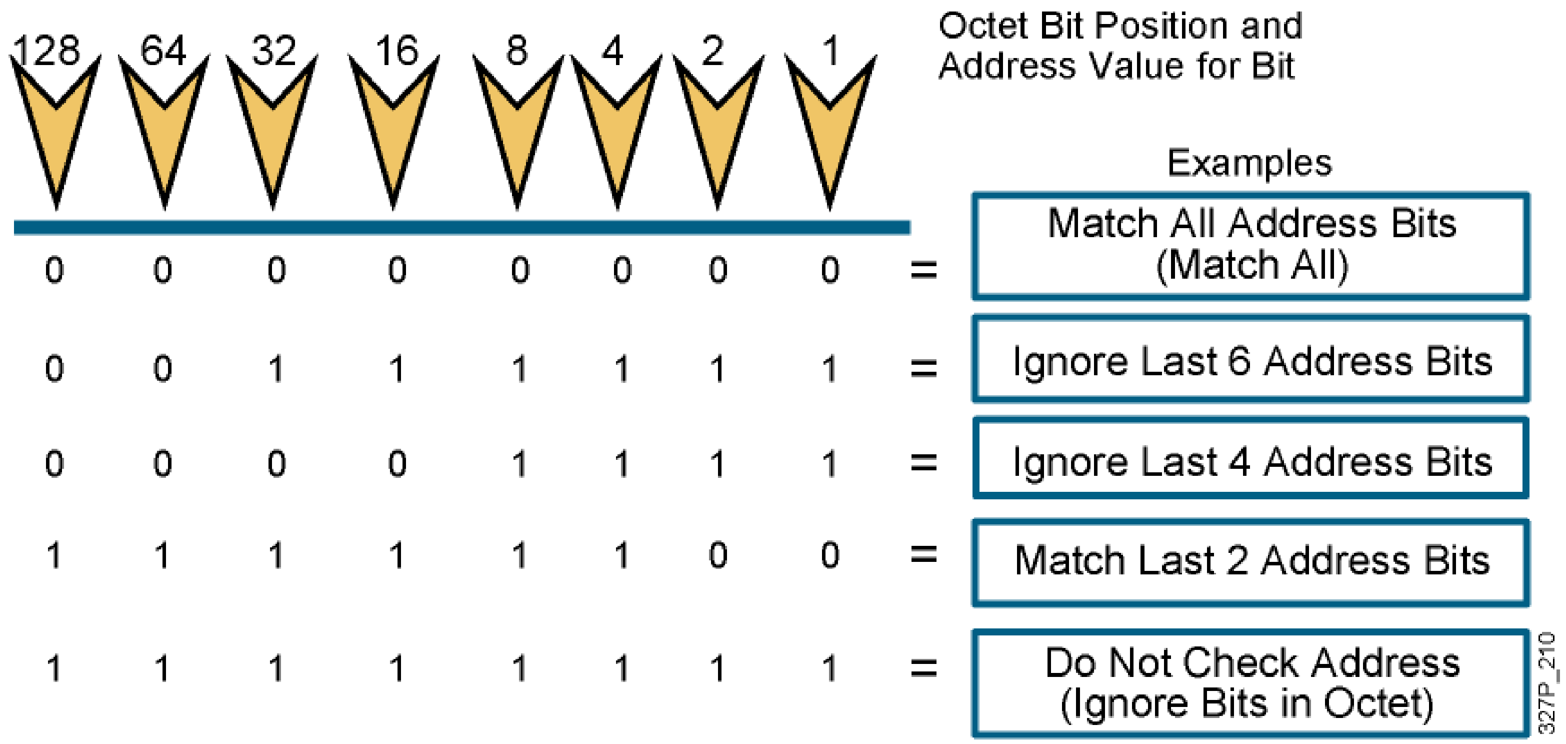
Reflexive ACLs: Used to allow outbound traffic and limit inbound traffic in response to sessions that originate inside

Time-Based ACLs



Time-based ACLs: Allow for access control based on the time of day and week

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
Examples
0 0 0 0 0 0 0 0 = Match 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 = Match Last 2 Address Bits
1 1 1 1 1 1 1 1 = Do Not Check Address (Ignore Bits in Octet)
327P_210

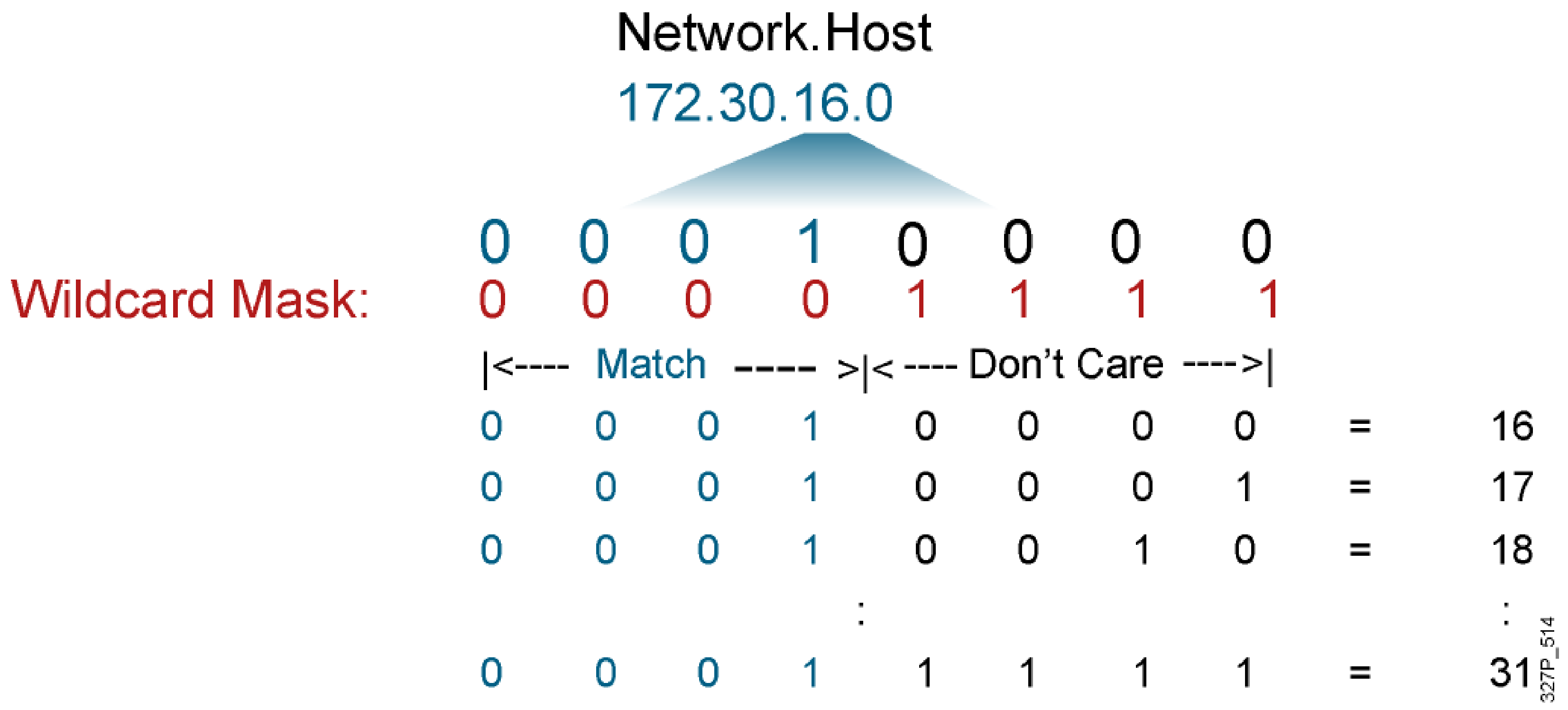
- 0 means to match the value of the corresponding address bit
- 1 means to ignore the value of the corresponding address bit

Wildcard Bits to Match IP Subnets

Match for IP subnets 172.30.16.0/24 to 172.30.31.0/24.

Address and wildcard mask:

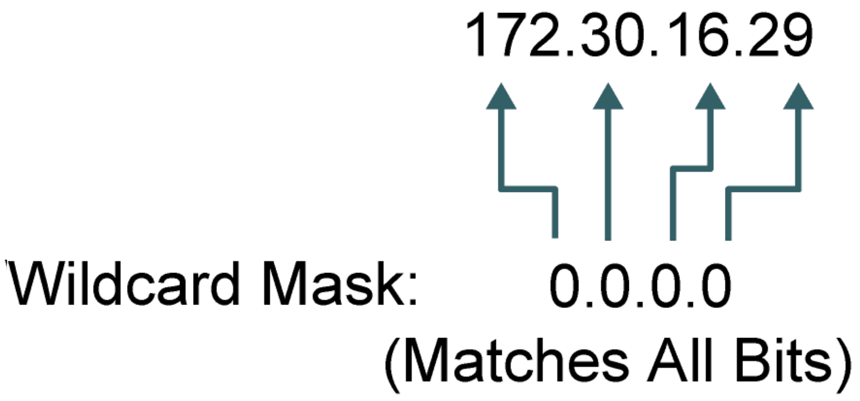
172.30.16.0 0.0.15.255



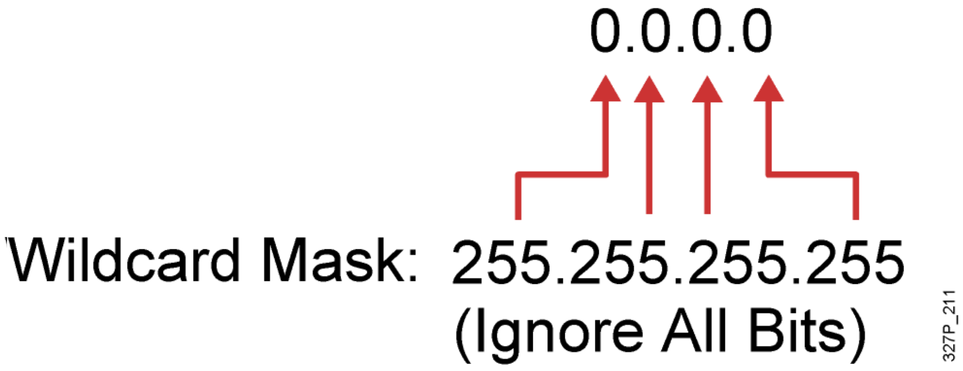
	0	0	0	1	0	0	0	0		
Wildcard Mask:	0	0	0	0	1	1	1	1		
	0	0	0	1	0	0	0	0	=	16
	0	0	0	1	0	0	0	1	=	17
	0	0	0	1	0	0	1	0	=	18
				:						:
	0	0	0	1	1	1	1	1	=	31

Wildcard Bit Mask Abbreviations

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



- 0.0.0.0 255.255.255.255 ignores all address bits
- Abbreviate expression with the keyword **any**

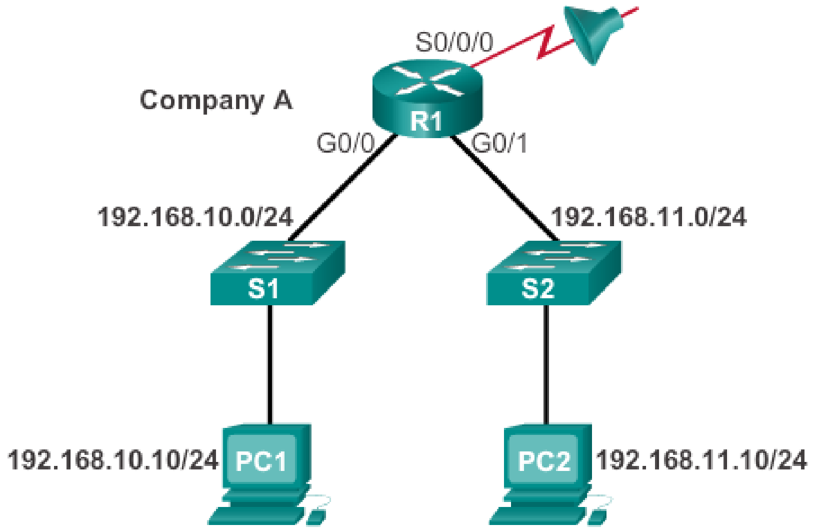


Summary (Theory)

- ACLs can be used for IP packet filtering or to identify traffic to assign it special handling.
- ACLs perform top-down processing and can be configured for incoming or outgoing traffic.
- You can create an ACL using a named or numbered ACL. Named or numbered ACLs can be configured as standard or extended ACLs, which determines what they can filter.
- Reflexive, dynamic, and time-based ACLs add more functionality to standard and extended ACLs.
- In a wildcard bit mask, a 0 bit means to match the corresponding address bit and a 1 bit means to ignore the corresponding address bit.

Configure Standard IPv4 ACLs

Entering Criteria Statements



ACL 1

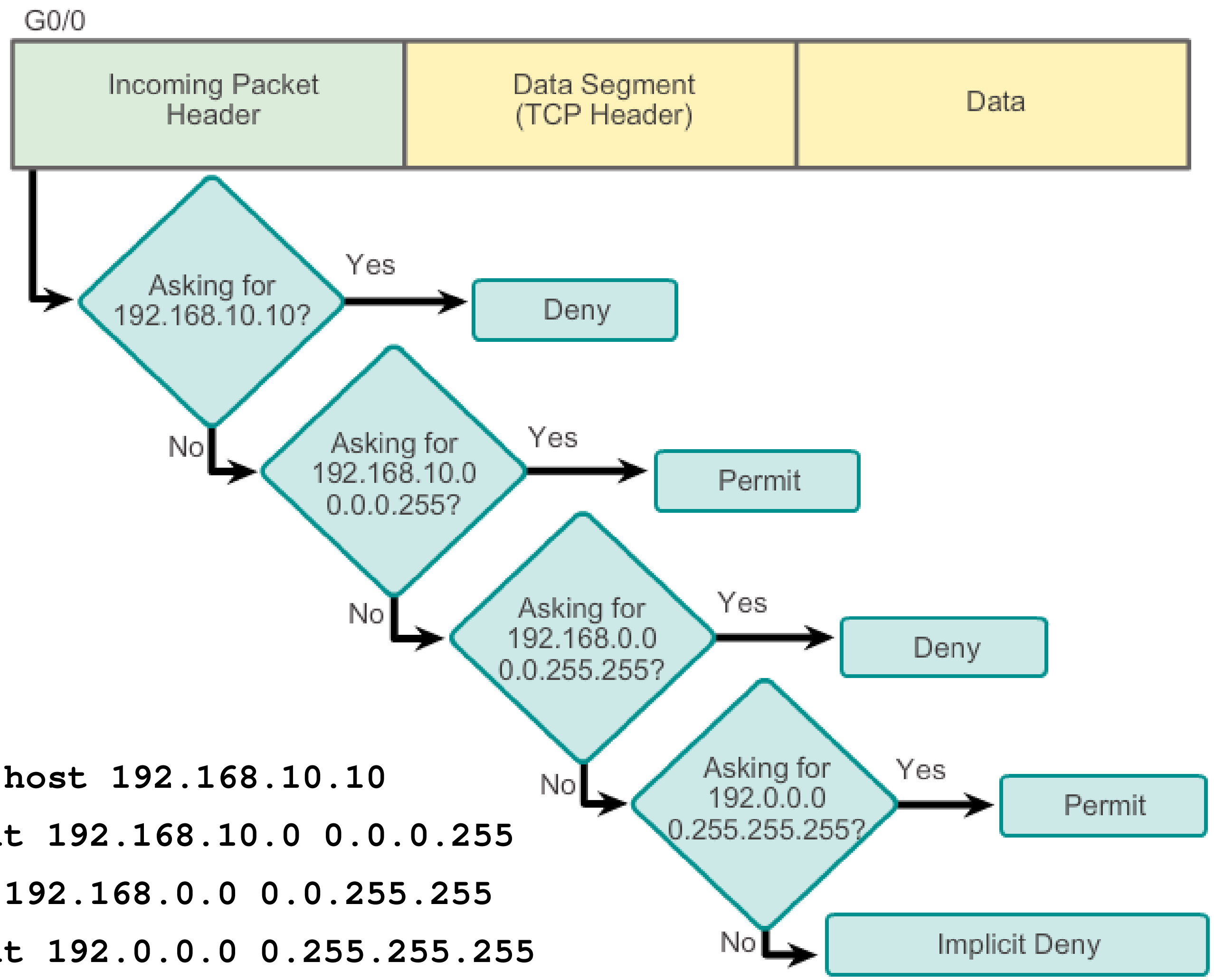
```
R1(config)#access-list 1 permit ip 192.168.10.0 0.0.0.255
```

ACL 2

```
R1(config)#access-list 2 permit ip 192.168.10.0 0.0.0.255
R1(config)#access-list 2 deny any
```

Configure Standard IPv4 ACLs

Configuring a Standard ACL


G0/0
Incoming Packet Header
Data Segment (TCP Header)
Data
Asking for 192.168.10.10?
Yes
Deny
No
Asking for 192.168.10.0 0.0.0.255?
Yes
Permit
No
Asking for 192.168.0.0 0.0.255.255?
Yes
Deny
No
Asking for 192.0.0.0 0.255.255.255?
Yes
Permit
No
Implicit Deny

Example ACL

- `access-list 2 deny host 192.168.10.10`
- `access-list 2 permit 192.168.10.0 0.0.0.255`
- `access-list 2 deny 192.168.0.0 0.0.255.255`
- `access-list 2 permit 192.0.0.0 0.255.255.255`

Numbered Standard IPv4 ACL Configuration

```
RouterX(config)#
```

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

- Uses 1 to 99 for the *access-list-number.*
- The first entry is assigned a sequence number of 10, and successive entries are incremented by 10.
- Default wildcard mask is 0.0.0.0 (only standard ACL).
- **no access-list *access-list-number*** removes the entire ACL.
- **remark** lets you add a description to the ACL.

```
RouterX(config-if)#
```

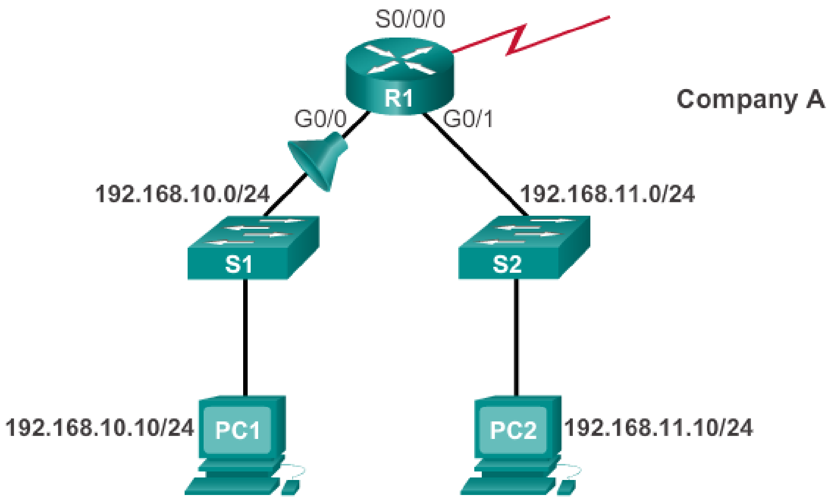
```
ip access-group access-list-number  {in | out}
```

- Activates the list on an interface.
- Sets inbound or outbound testing.
- **no ip access-group *access-list-number* {in | out}** removes the ACL from the interface.

Configure Standard IPv4 ACLs

Applying Standard ACLs to Interfaces (Cont.)

Deny a Specific Host



```
R1(config)#access-list 1 deny host 192.168.10.10
R1(config)#access-list 1 permit any
R1(config)#interface g0/0
R1(config-if)#ip access-group 1 in
```

Standard ACLs to Control vty Access

```
RouterX(config-line)#
```

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

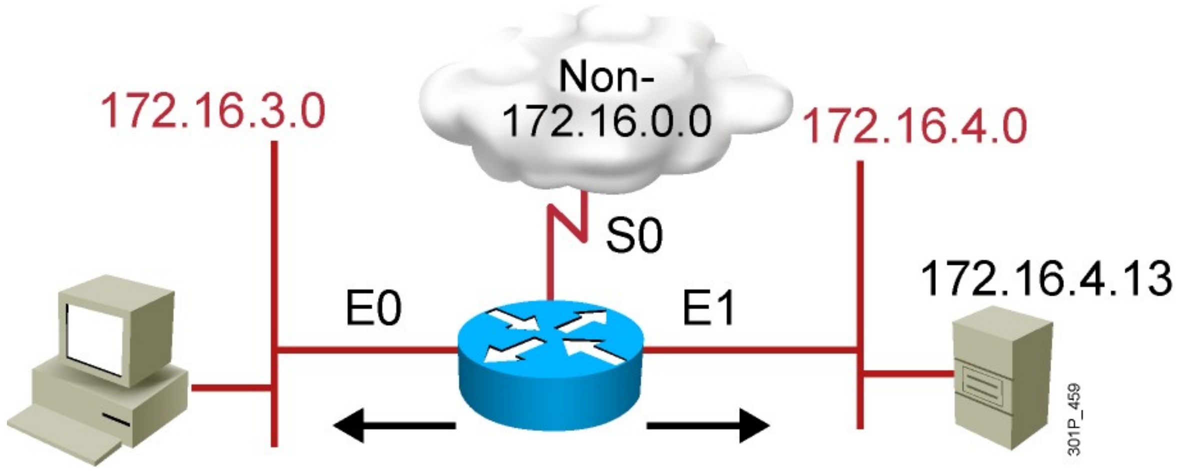
- Restricts incoming or outgoing connections between a particular vty and the addresses in an ACL

Example:

```
access-list 12 permit 192.168.1.0 0.0.0.255
!(implicit deny any)
!
line vty 0 4
 access-class 12 in
```

- Permits only hosts in network 192.168.1.0 0.0.0.255 to connect to the router vty lines

Numbered Standard IPv4 ACL Example 1

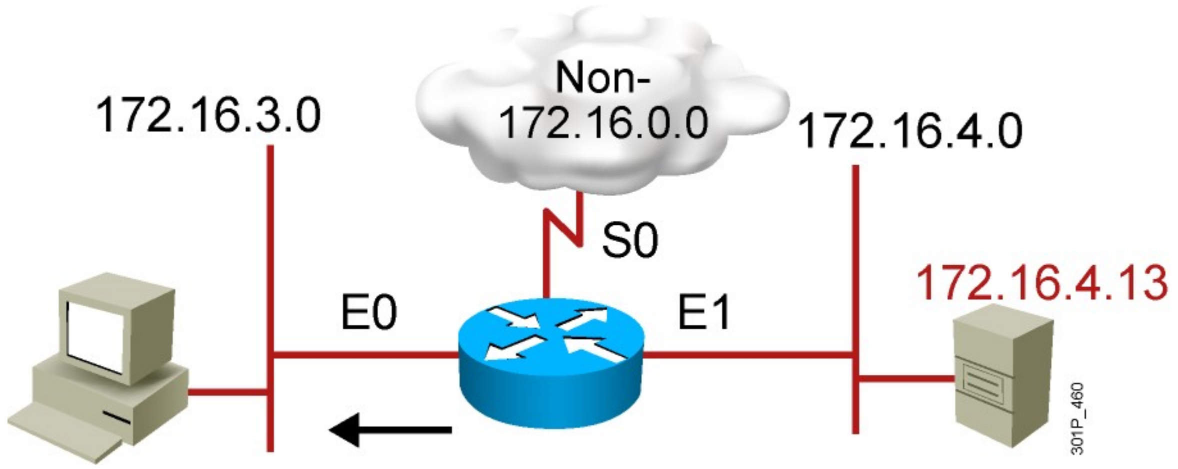


```
RouterX(config)# access-list 1 permit 172.16.0.0  0.0.255.255
!(implicit deny all - not visible in the list)
!(access-list 1 deny 0.0.0.0   255.255.255.255)

RouterX(config)# interface ethernet 0
RouterX(config-if)# ip access-group 1 out
RouterX(config)# interface ethernet 1
RouterX(config-if)# ip access-group 1 out
```

Permit my network only

Numbered Standard IPv4 ACL Example 2

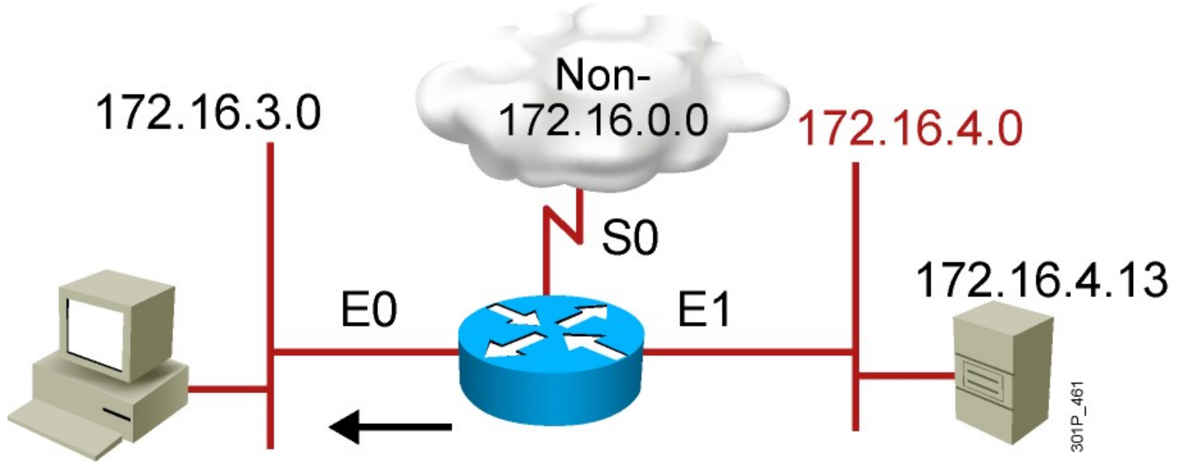


```
RouterX(config)# access-list 1 deny 172.16.4.13 0.0.0.0
RouterX(config)# access-list 1 permit 0.0.0.0  255.255.255.255
!(implicit deny all)
!(access-list 1 deny 0.0.0.0   255.255.255.255)

RouterX(config)# interface ethernet 0
RouterX(config-if)# ip access-group 1 out
```

Deny a specific host

Numbered Standard IPv4 ACL Example 3

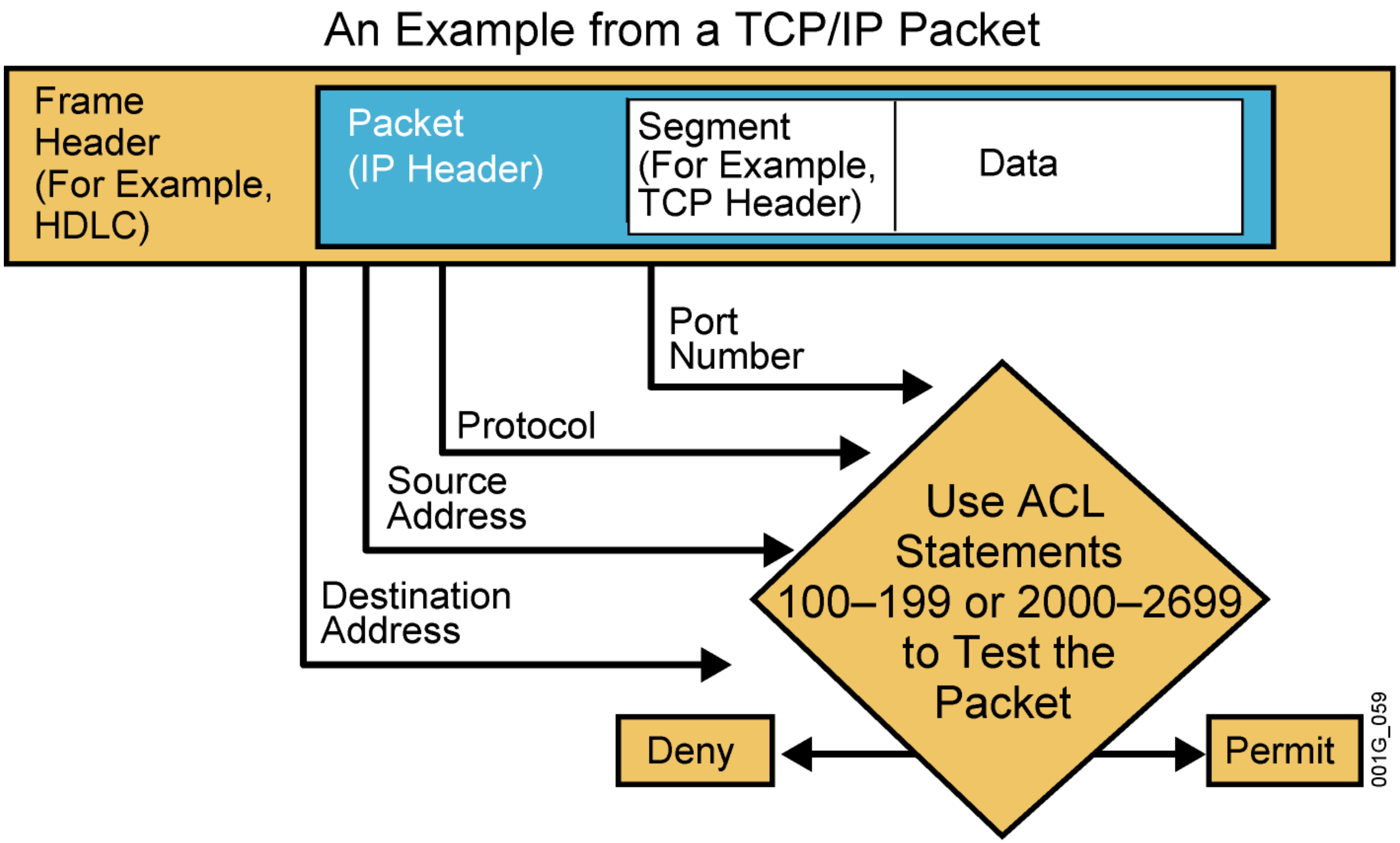


```
RouterX(config)# access-list 1 deny 172.16.4.0  0.0.0.255
RouterX(config)# access-list 1 permit any
!(implicit deny all)
!(access-list 1 deny 0.0.0.0   255.255.255.255)

RouterX(config)# interface ethernet 0
RouterX(config-if)# ip access-group 1 out
```

Deny a specific subnet

Testing Packets with Numbered Extended IPv4 ACLs


An Example from a TCP/IP Packet
Frame Header (For Example, HDLC)
Packet (IP Header)
Segment (For Example, TCP Header)
Data
Port Number
Protocol
Source Address
Destination Address
Use ACL Statements 100–199 or 2000–2699 to Test the Packet
Deny
Permit
001G_059

Numbered Extended IPv4 ACL Configuration

```
RouterX(config)#
```

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

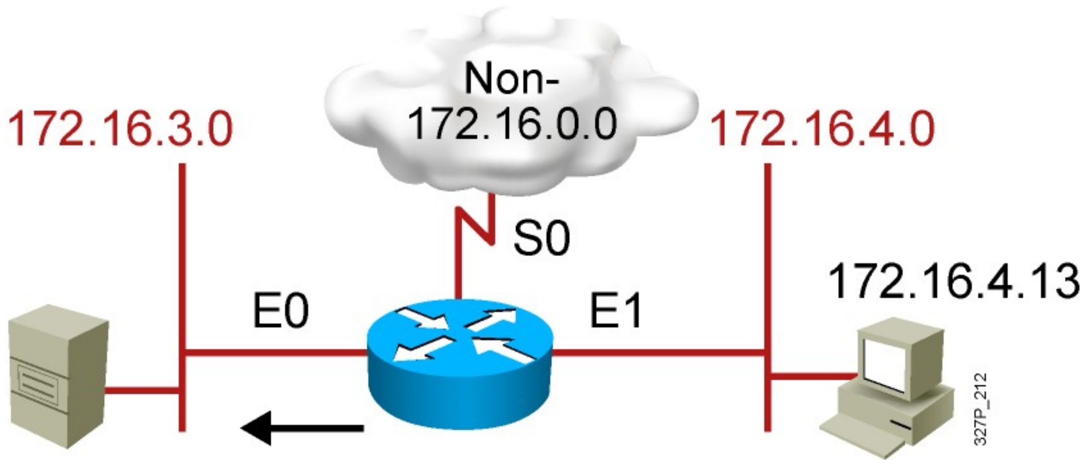
- Sets parameters for this list entry

```
RouterX(config-if)#
```

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

- Activates the extended list on an interface

Numbered Extended IPv4 ACL Example 1

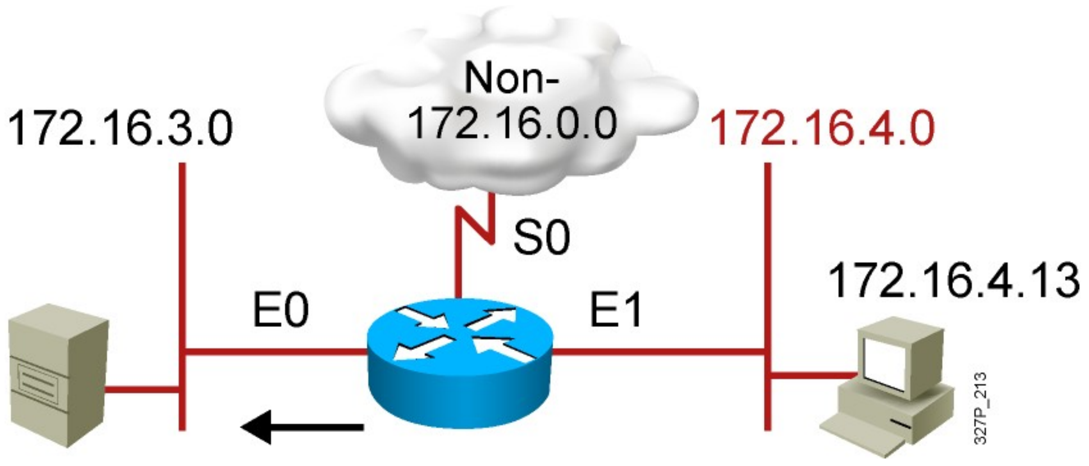


```
RouterX(config)# access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
RouterX(config)# access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
RouterX(config)# access-list 101 permit ip any any
!(implicit deny all)
!(access-list 101 deny ip 0.0.0.0 255.255.255.255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

RouterX(config)# interface ethernet 0
RouterX(config-if)# ip access-group 101 out
```

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

Numbered Extended IPv4 ACL Example 2



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

RouterX(config)# interface ethernet 0
RouterX(config-if)# ip access-group 101 out
```

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

Named IP ACL Configuration

```
RouterX(config)#
```

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

- Alphanumeric name string must be unique

```
RouterX(config {std- | ext-}nacl)#
```

```
[sequence-number] {permit | deny} {ip access list test conditions}
{permit | deny} {ip access list test conditions}
```

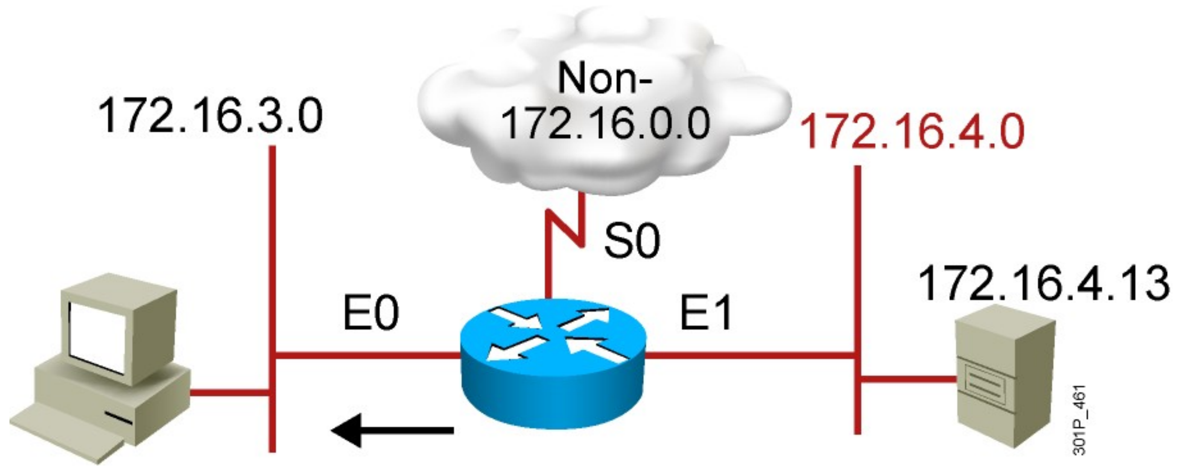
- If not configured, sequence numbers are generated automatically starting at 10 and incrementing by 10
- **no** *sequence number* removes the specific test from the named ACL

```
RouterX(config-if)#
```

```
ip access-group name {in | out}
```

- Activates the named IP ACL on an interface

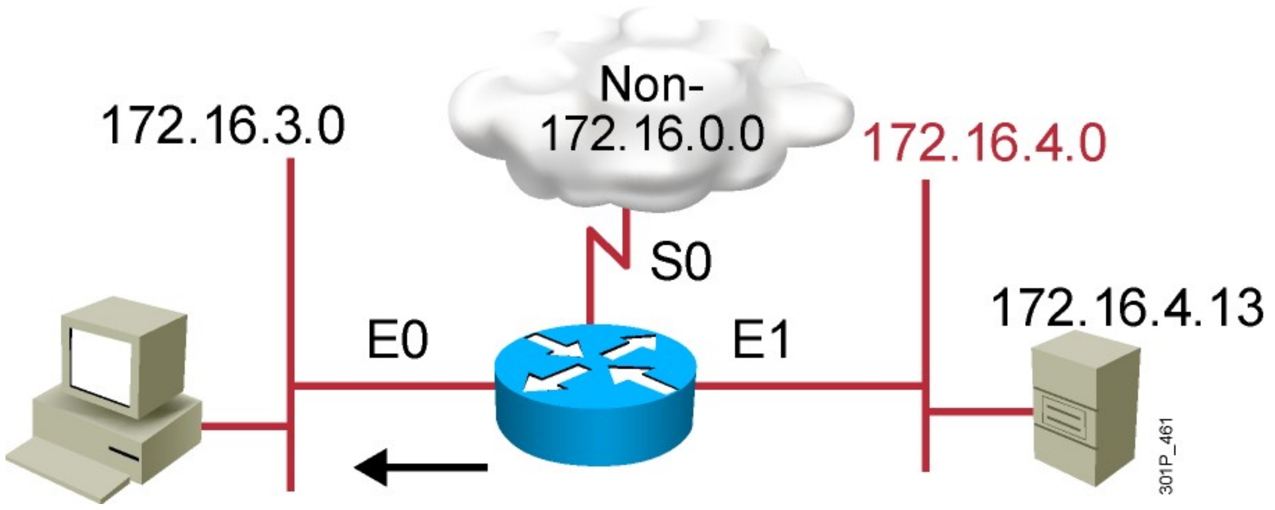
Named Standard IPv4 ACL Example



```
RouterX(config)#ip access-list standard troublemaker
RouterX(config-std-nacl)#deny host 172.16.4.13
RouterX(config-std-nacl)#permit 172.16.4.0 0.0.0.255
RouterX(config-std-nacl)#interface e0
RouterX(config-if)#ip access-group troublemaker out
```

Deny a specific host

Named Extended IPv4 ACL Example



```
RouterX(config)#ip access-list extended badgroup
RouterX(config-ext-nacl)#deny tcp 172.16.4.0 0.0.0.255 any eq 23
RouterX(config-ext-nacl)#permit ip any any
RouterX(config-ext-nacl)#interface e0
RouterX(config-if)#ip access-group badgroup out
```

Deny Telnet from a specific subnet

Commenting ACL Statements

```
RouterX(config)#
```

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

- Creates a named ACL

```
RouterX(config {std- | ext-}nacl)#
```

```
remark remark
```

- Creates a named ACL comment

Or

```
RouterX(config)#
```

```
access-list access-list-number remark remark
```

- Creates a numbered ACL comment

Monitoring ACL Statements

```
RouterX# show access-lists {access-list-number | name}
```

```
RouterX# show access-lists
Standard IP access list SALES
    10 deny   10.1.1.0, wildcard bits 0.0.0.255
    20 permit 10.3.3.1
    30 permit 10.4.4.1
    40 permit 10.5.5.1
Extended IP access list ENG
    10 permit tcp host 10.22.22.1 any eq telnet (25 matches)
    20 permit tcp host 10.33.33.1 any eq ftp
    30 permit tcp host 10.44.44.1 any eq ftp-data
```

Displays all access lists

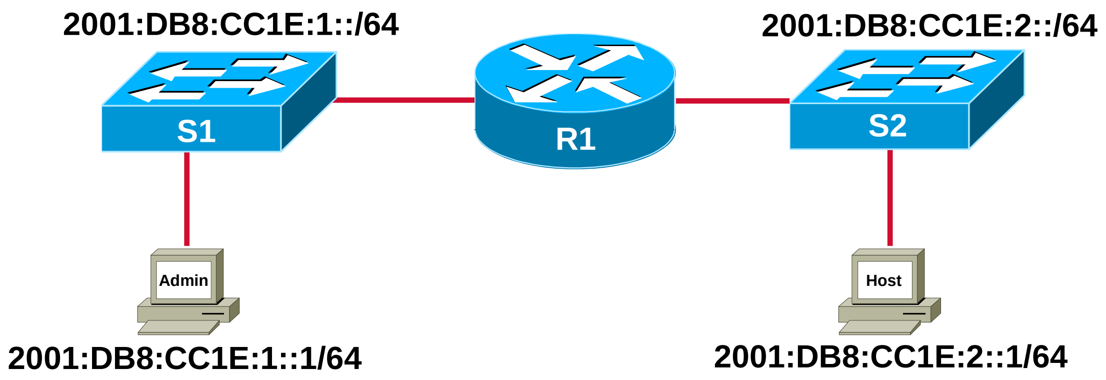
Verifying ACLs

```
RouterX# show ip interfaces 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 unreachables 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>
```

IPv6 ACL Operation

- IPv6 ACLs are very similar to IPv4 ACLs in both operation and configuration. Being familiar with IPv4 access lists makes IPv6 ACLs easy to understand and configure.
- **IPv6 has only one type of ACL, which is equivalent to an IPv4 extended named ACL.**
- There are no numbered ACLs in IPv6, only named ACL.
- IPv4 uses the command **ip access-group** to apply an IPv4 ACL to an IPv4 interface. IPv6 uses the **ipv6 traffic-filter** command to perform the same function for IPv6 ACLs.
- IPv6 ACLs do not use wildcard masks. Instead, the prefix-length is used to indicate how much of an IPv6 source or destination address should be matched.

Restrict Access to VTY Lines



- Allow the Admin PC to telnet into R1 while denying all others.
- Use the **ipv6 access-list** command to create a named IPv6 ACL. Like IPv4 named ACLs, IPv6 names are alphanumeric, case sensitive and must be unique.
- Use the **permit** or **deny** statements to specify one or more conditions to determine if a packet is forwarded or dropped.
- Use the **ipv6 access-class** command to apply the ACL to the VTY lines.

ACL Configuration Example

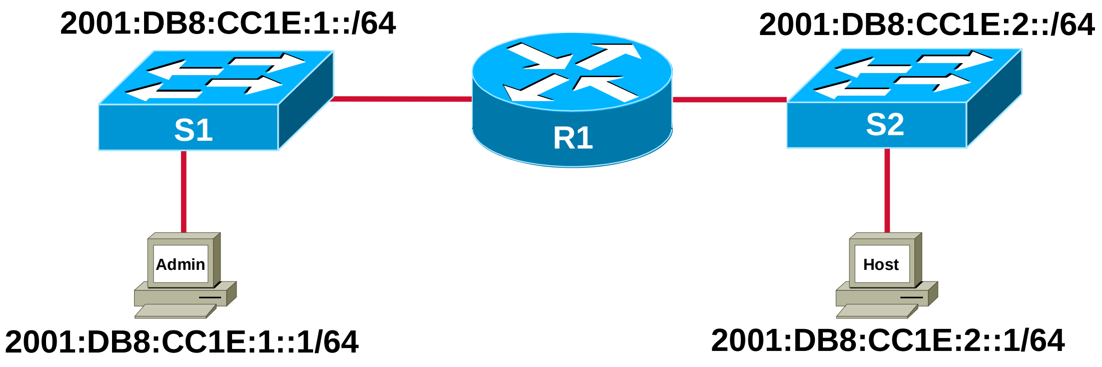
- The **permit** statement only allows the Admin PC to telnet into R1.
- The implicit deny statement (not configured) will deny all others from establishing a telnet session into R1.
- Apply the ACL to the VTY lines, using the **ipv6 access-class** command and with **in** as the direction.

```
R1(config)#ipv6 access-list NO_TELNET
R1(config-ipv6-acl)#permit tcp host 2001:db8:cc1e:1::1 any eq 23
R1(config-ipv6-acl)#exit
R1(config)#line vty 0 15
R1(config-line)#ipv6 access-class NO_TELNET in
R1(config-line)#exit
R1(config)#
```

ACL Configuration Example (Cont'd)

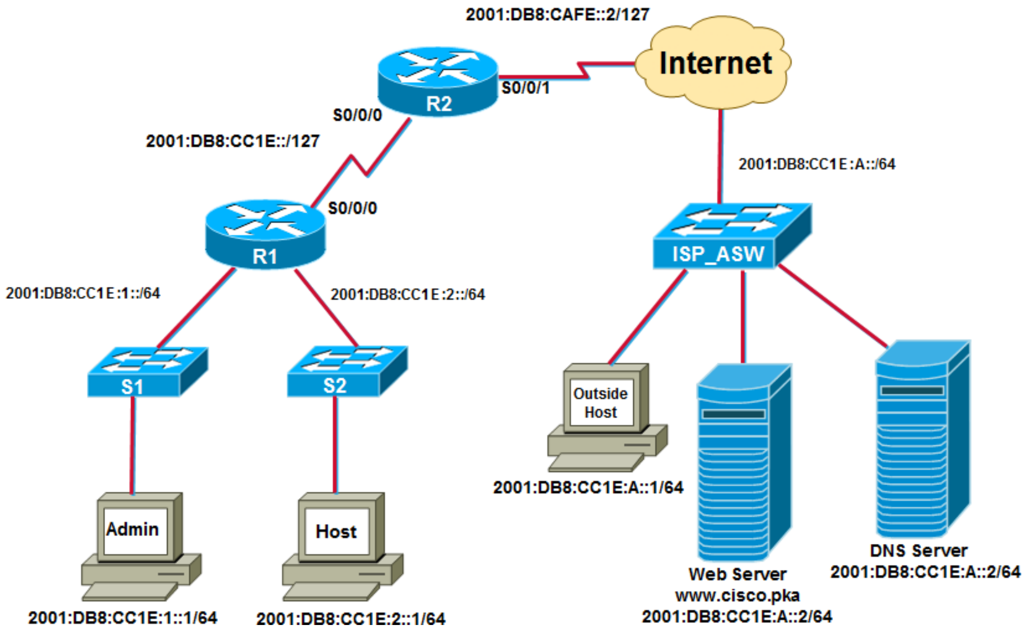
- The **show access-lists** command displays all configured IPv4 and IPv6 ACLs configured on the router.
- The **show ipv6 access-list** command will display all configured IPv6 access lists specified by name. (No numbered IPv6 ACLs)

```
R1#show ipv6 access-list
IPv6 access list NO_TELNET
    permit tcp host 2001:DB8:CC1E:1::1 any eq telnet
```



Restrict Web Server Access

Configure an extended ACL to block TCP applications HTTP & FTP traffic sourcing from the Admin PC and Host PC specific IPv6 address when destined for the Internet LAN. Permit all other types of traffic.



```
R1(config)#ipv6 access-list DENY_WWW_FTP
R1(config-ipv6-acl)#remark Deny WWW and FTP access from R1 LANs to Web Server
R1(config-ipv6-acl)#deny tcp 2001:db8:cc1e:1::/64 2001:db8:cc1e:a::/64 eq www
R1(config-ipv6-acl)#deny tcp 2001:db8:cc1e:1::/64 2001:db8:cc1e:a::/64 eq ftp
R1(config-ipv6-acl)#deny tcp 2001:db8:cc1e:2::/64 2001:db8:cc1e:a::/64 eq www
R1(config-ipv6-acl)#deny tcp 2001:db8:cc1e:2::/64 2001:db8:cc1e:a::/64 eq ftp
R1(config-ipv6-acl)#permit ipv6 any any
R1(config-ipv6-acl)#exit
R1(config)# int s0/0/0
R1(config-if)# ipv6 traffic-filter DENY_WWW_FTP out
```

IPv6 ACL Verification Commands

```
R1#show ipv6 access-list DENY_WWW_FTP
IPv6 access list DENY_WWW_FTP

deny tcp 2001:DB8:CC1E:1::/64
2001:DB8:CC1E:A::/64 eq www
(28 match(es))

deny tcp 2001:DB8:CC1E:1::/64
2001:DB8:CC1E:A::/64 eq ftp

deny tcp 2001:DB8:CC1E:2::/64
2001:DB8:CC1E:A::/64 eq ftp

deny tcp 2001:DB8:CC1E:2::/64
2001:DB8:CC1E:A::/64 eq www

permit ipv6 any any (3 match(es))
```

The ACL matched 28 denies based on the ACL statement.

The **deny** and **permit** command is used to specify one or more conditions to determine if a packet is forwarded or dropped.

Editing IPv6 ACLs

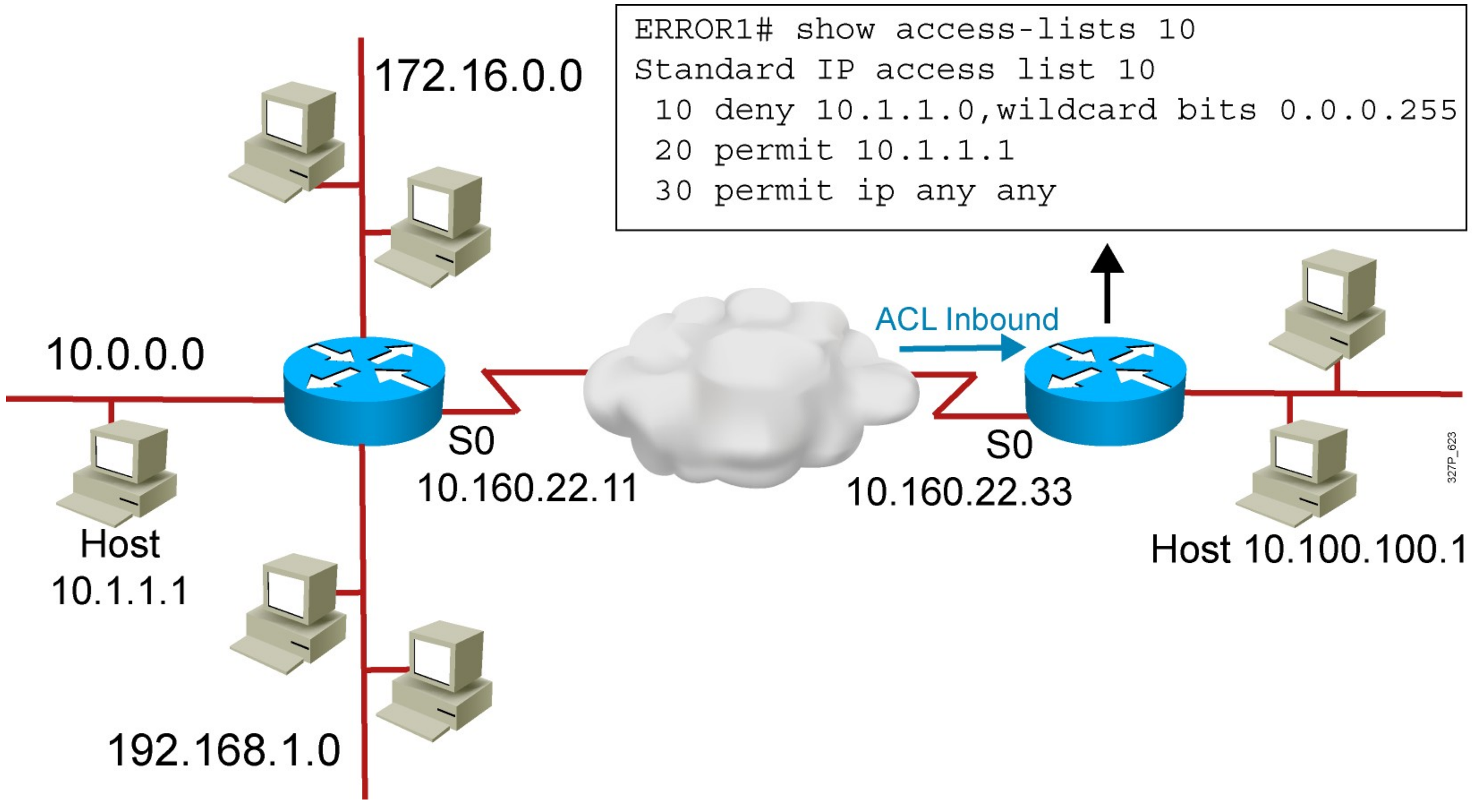
- In order to edit an IPv6 ACL, you can insert an ACL statement based on the sequence number. By default, sequence numbers are in increments of 10.

```
R1#show access-lists
IPv6 access list NO_TELNET
    permit tcp host 2001:DB8:CC1E:1::1 any eq telnet (2 matches) sequence 10
IPv6 access list DENY_WWW_FTP
    deny tcp 2001:DB8:CC1E:1::/64 2001:DB8:CC1E:A::/64 eq www sequence 20
    deny tcp 2001:DB8:CC1E:1::/64 2001:DB8:CC1E:A::/64 eq ftp sequence 30
    deny tcp 2001:DB8:CC1E:2::/64 2001:DB8:CC1E:A::/64 eq www sequence 40
    deny tcp 2001:DB8:CC1E:2::/64 2001:DB8:CC1E:A::/64 eq ftp sequence 50
    permit ipv6 any any sequence 60
```

```
R1(config)#ipv6 access-list DENY_WWW_FTP
R1(config-ipv6-acl)#permit tcp host 2001:db8:cc1e:1::12 host 2001:db8:cc1e:a:: eq www sequence 25
R1(config-ipv6-acl)#permit tcp host 2001:db8:cc1e:1::12 host 2001:db8:cc1e:a:: eq ftp sequence 25
```

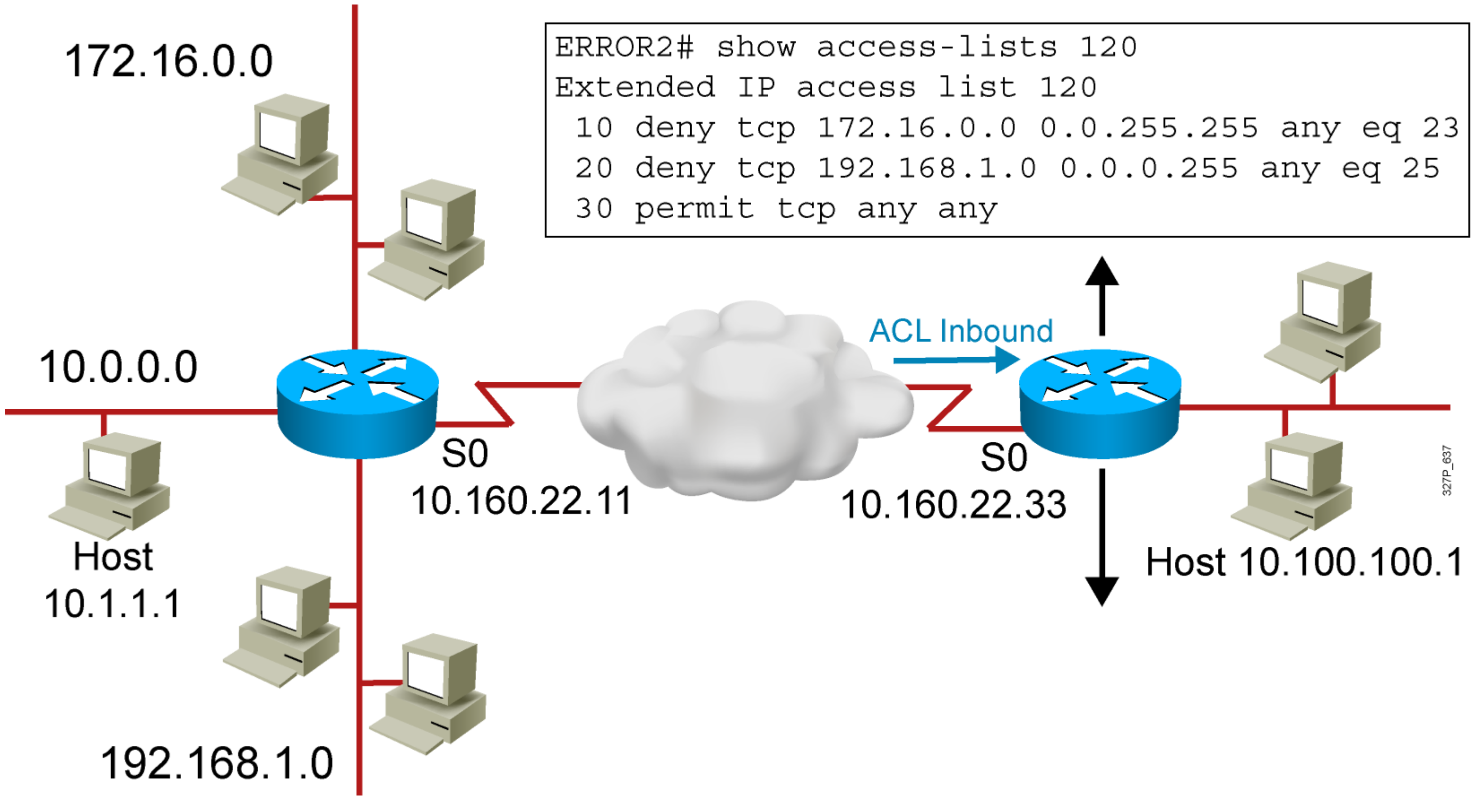
```
R1#show ipv6 access-list
IPv6 access list NO_TELNET
    permit tcp host 2001:DB8:CC1E:1::1 any eq telnet (2 matches) sequence 10
IPv6 access list DENY_WWW_FTP
    deny tcp 2001:DB8:CC1E:1::/64 2001:DB8:CC1E:A::/64 eq www sequence 20
    permit tcp host 2001:DB8:CC1E:1::12 host 2001:DB8:CC1E:A:: eq www sequence 25
    permit tcp host 2001:DB8:CC1E:1::12 host 2001:DB8:CC1E:A:: eq ftp sequence 25
    deny tcp 2001:DB8:CC1E:1::/64 2001:DB8:CC1E:A::/64 eq ftp sequence 30
    deny tcp 2001:DB8:CC1E:2::/64 2001:DB8:CC1E:A::/64 eq ftp sequence 40
    deny tcp 2001:DB8:CC1E:2::/64 2001:DB8:CC1E:A::/64 eq www sequence 50
    permit ipv6 any any sequence 60
```

Troubleshooting Common ACL Errors


```
ERROR1# show access-lists 10
Standard IP access list 10
 10 deny 10.1.1.0,wildcard bits 0.0.0.255
 20 permit 10.1.1.1
 30 permit ip any any
```

Error 1: Host 10.1.1.1 has no connectivity with 10.100.100.1.

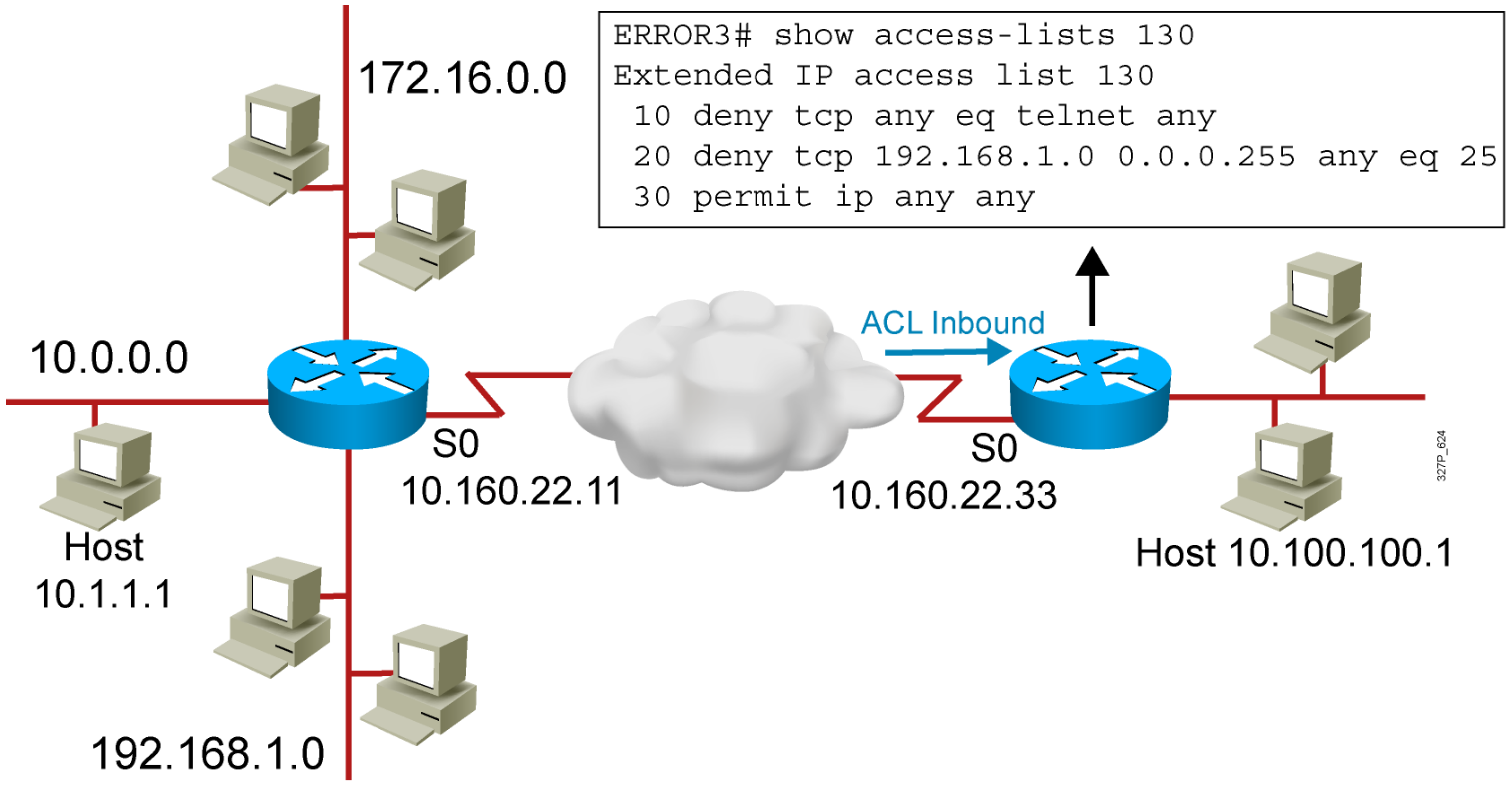
Troubleshooting Common ACL Errors (Cont.)



```
ERROR2# show access-lists 120
Extended IP access list 120
 10 deny tcp 172.16.0.0 0.0.255.255 any eq 23
 20 deny tcp 192.168.1.0 0.0.0.255 any eq 25
 30 permit tcp any any
```

Error 2: The 192.168.1.0 network cannot use TFTP to connect to 10.100.100.1.

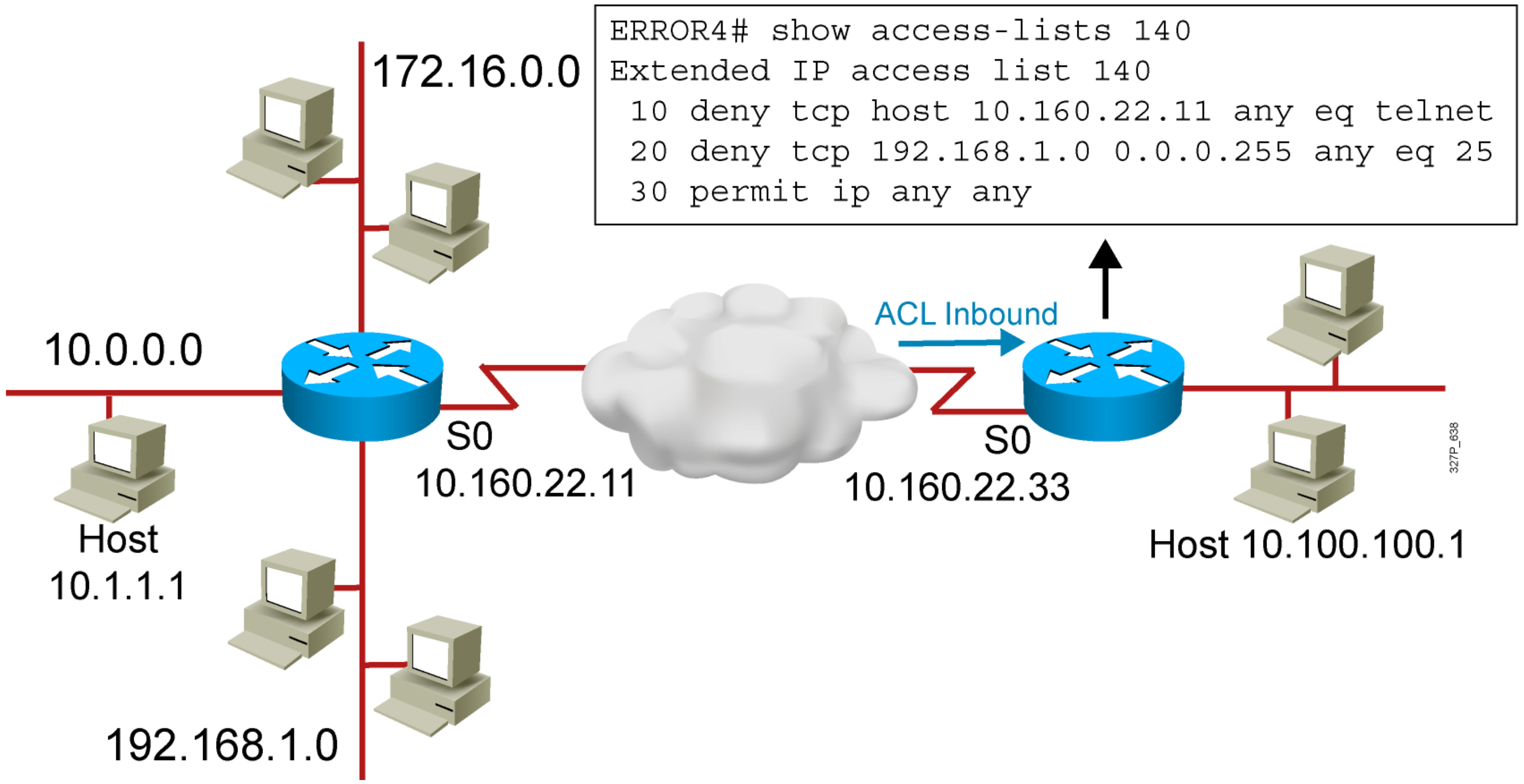
Troubleshooting Common ACL Errors (Cont.)



```
ERROR3# show access-lists 130
Extended IP access list 130
 10 deny tcp any eq telnet any
 20 deny tcp 192.168.1.0 0.0.0.255 any eq 25
 30 permit ip any any
```

Error 3: 172.16.0.0 network can use Telnet to connect to 10.100.100.1, but this connection should not be allowed.

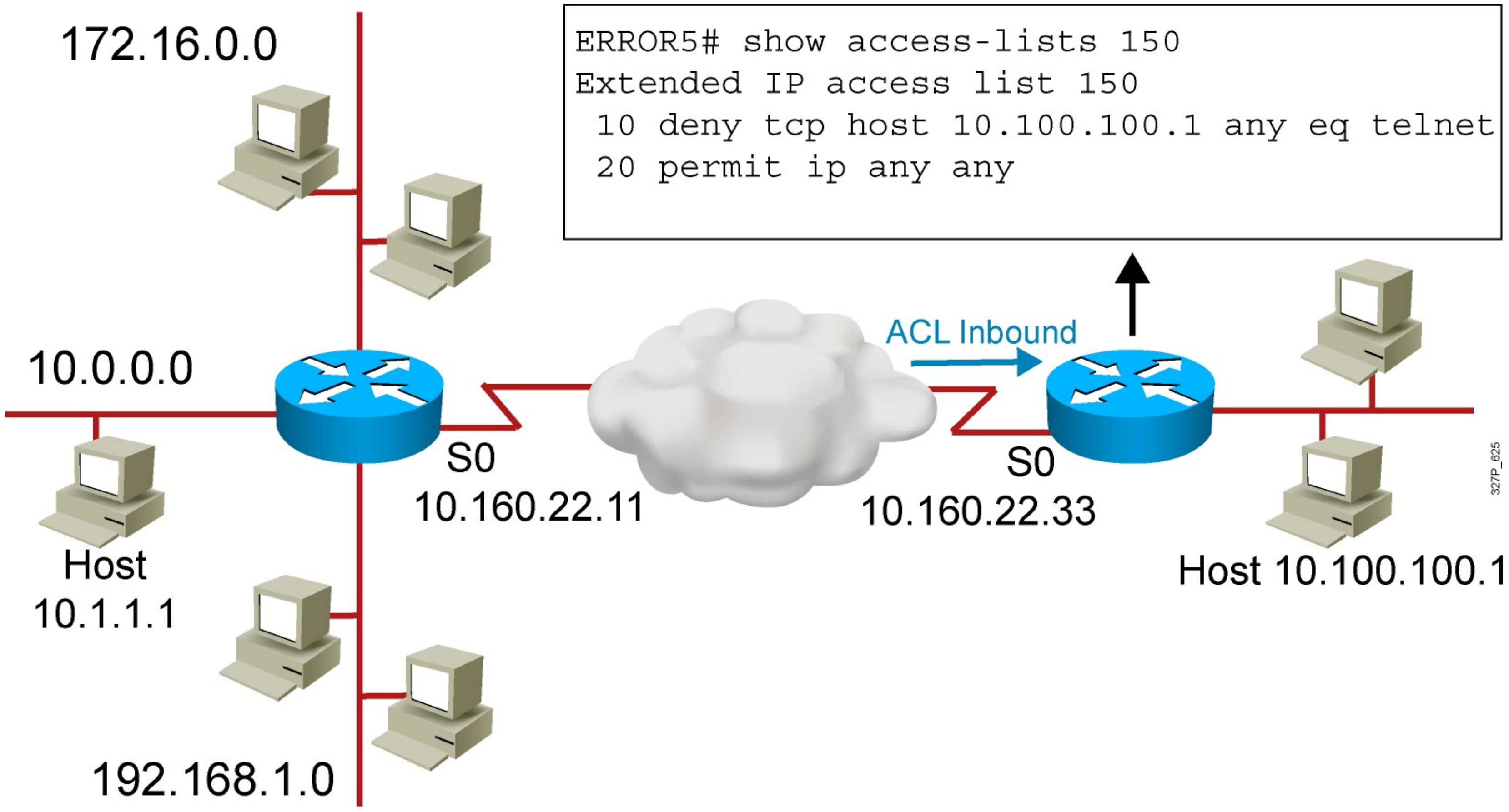
Troubleshooting Common ACL Errors (Cont.)



```
ERROR4# show access-lists 140
Extended IP access list 140
 10 deny tcp host 10.160.22.11 any eq telnet
 20 deny tcp 192.168.1.0 0.0.0.255 any eq 25
 30 permit ip any any
```

Error 4: Host 10.1.1.1 can use Telnet to connect to 10.100.100.1, but this connection should not be allowed.

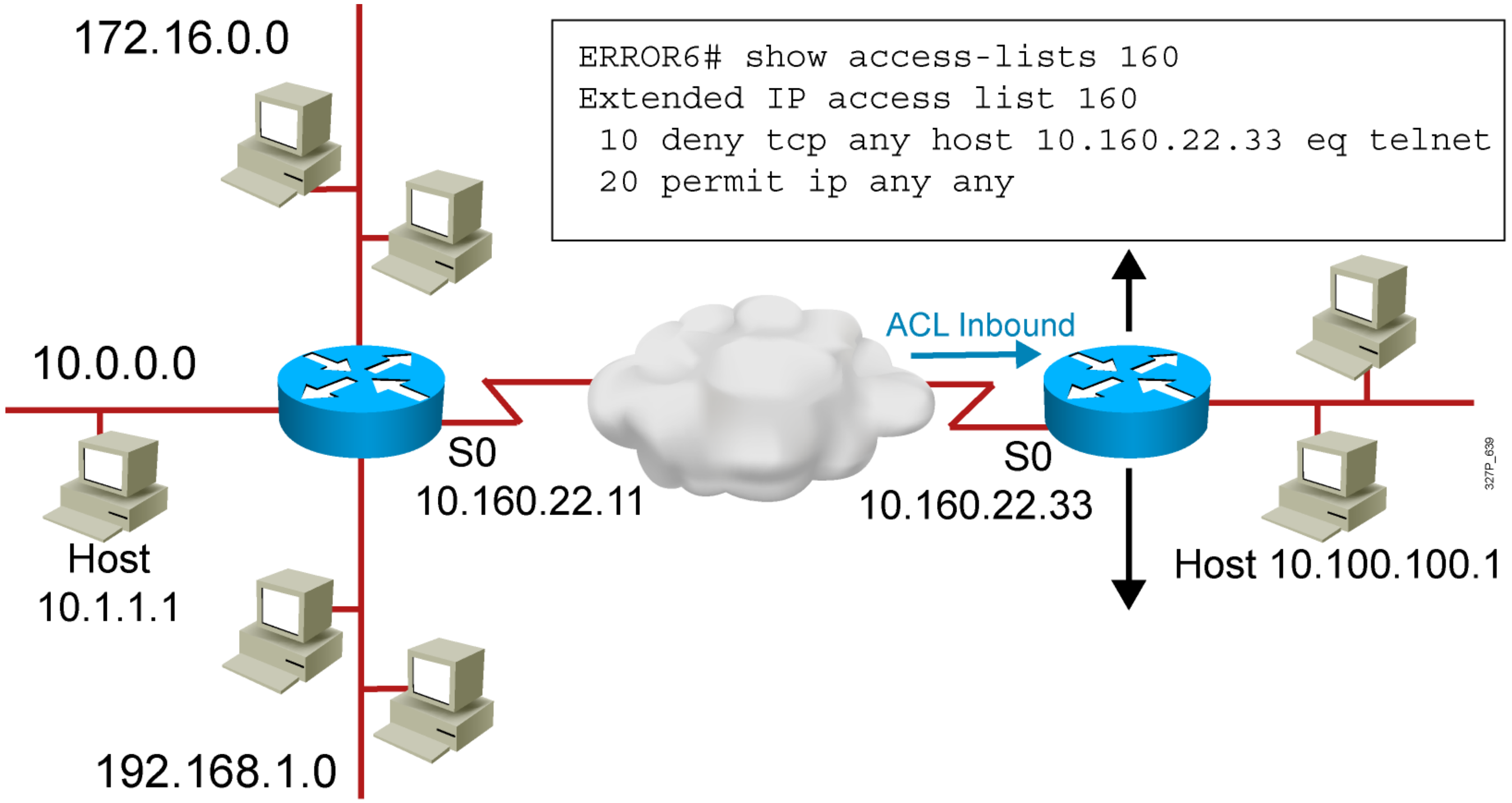
Troubleshooting Common ACL Errors (Cont.)



```
ERROR5# show access-lists 150
Extended IP access list 150
 10 deny tcp host 10.100.100.1 any eq telnet
 20 permit ip any any
```

Error 5: Host 10.100.100.1 can use Telnet to connect to 10.1.1.1, but this connection should not be allowed.

Troubleshooting Common ACL Errors (Cont.)



```
ERROR6# show access-lists 160
Extended IP access list 160
 10 deny tcp any host 10.160.22.33 eq telnet
 20 permit ip any any
```

Error 6: Host 10.1.1.1 can use Telnet to connect into router B, but this connection should not be allowed.

Summary

- ACLs can be used for IP packet filtering or to identify traffic to assign it special handling.
- ACLs perform top-down processing and can be configured for incoming or outgoing traffic.
- You can create an ACL using a named or numbered ACL. Named or numbered ACLs can be configured as standard or extended ACLs, which determines what they can filter.
- Reflexive, dynamic, and time-based ACLs add more functionality to standard and extended ACLs.
- In a wildcard bit mask, a 0 bit means to match the corresponding address bit and a 1 bit means to ignore the corresponding address bit.

Thank you for your attention