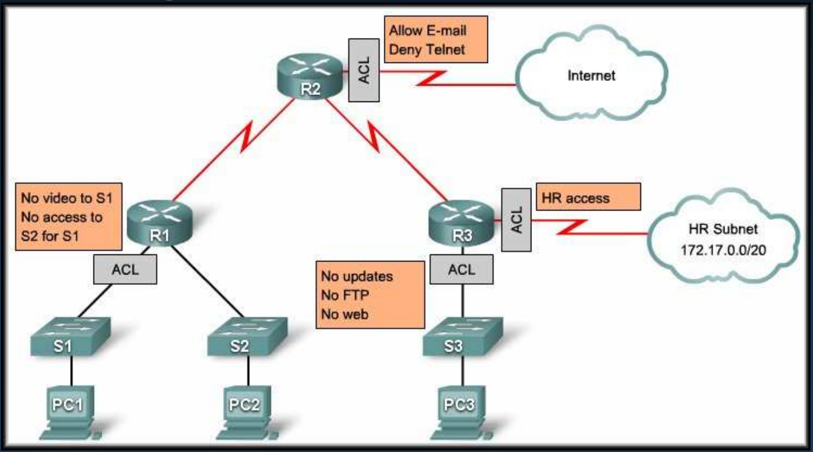
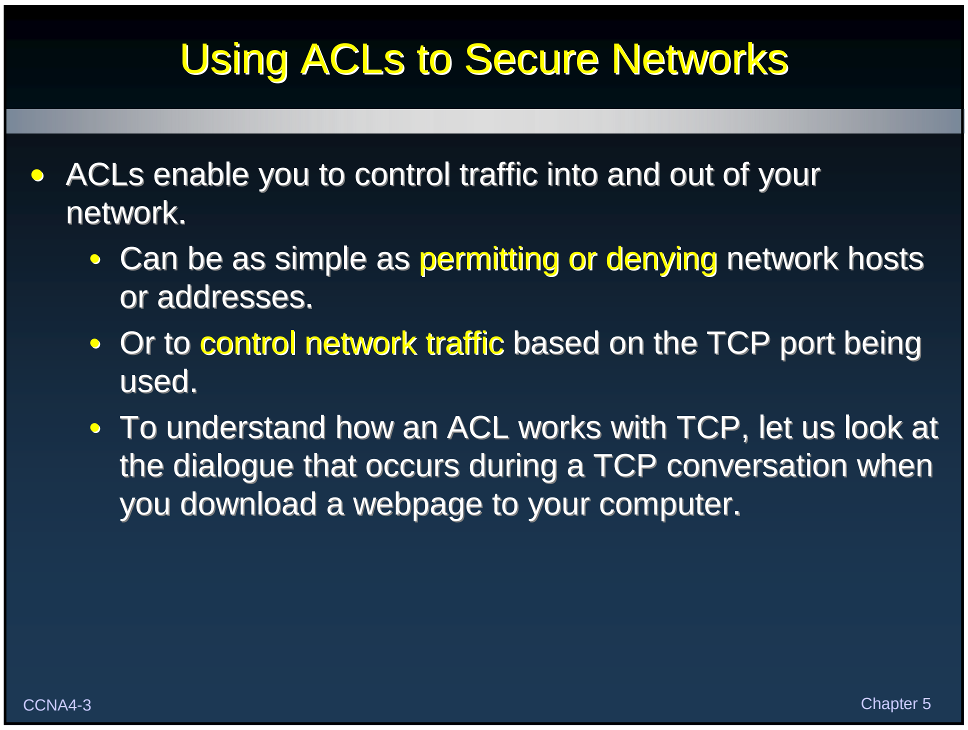
Chapter 5

Access Control Lists (ACLs)

Access Control Lists

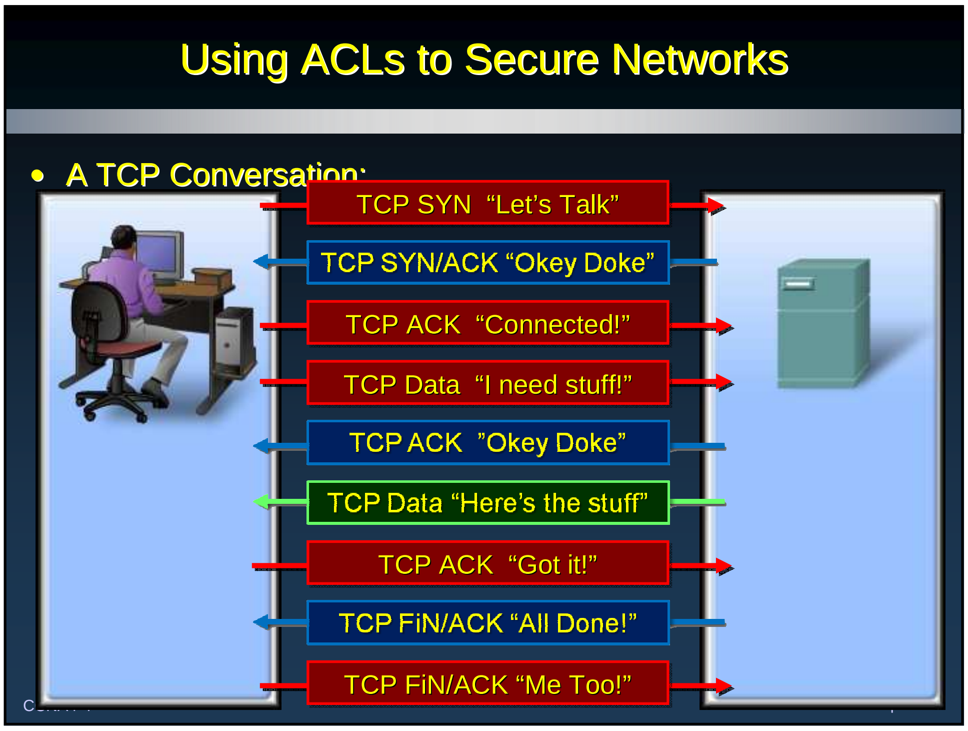
Using ACLs to Secure Networks





Using ACLs to Secure Networks

- ACLs enable you to control traffic into and out of your network.
  - Can be as simple as permitting or denying network hosts or addresses.
  - Or to control network traffic based on the TCP port being used.
  - To understand how an ACL works with TCP, let us look at the dialogue that occurs during a TCP conversation when you download a webpage to your computer.



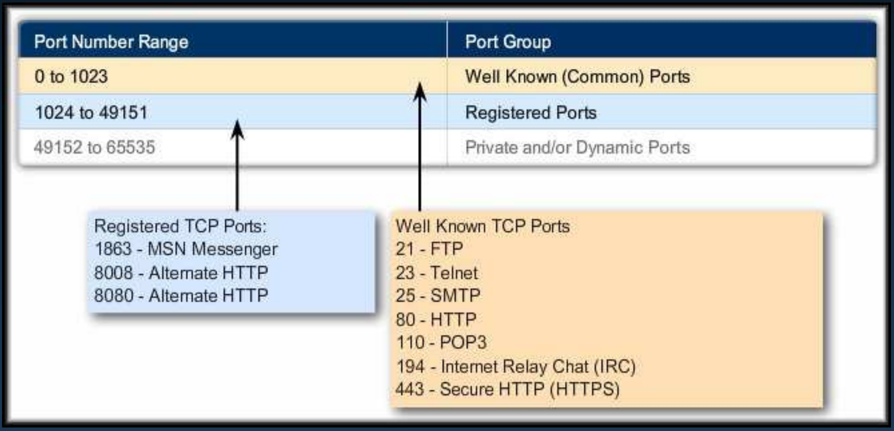
Using ACLs to Secure Networks

- A TCP Conversation:

TCP SYN “Let’s Talk”
TCP SYN/ACK “Okey Doke”
TCP ACK “Connected!”
TCP Data “I need stuff!”
TCP ACK ”Okey Doke”
TCP Data “Here’s the stuff”
TCP ACK “Got it!”
TCP FiN/ACK “All Done!”
TCP FiN/ACK “Me Too!”

Using ACLs to Secure Networks

- The TCP data segment also identifies the port matching the requested service…..TCP



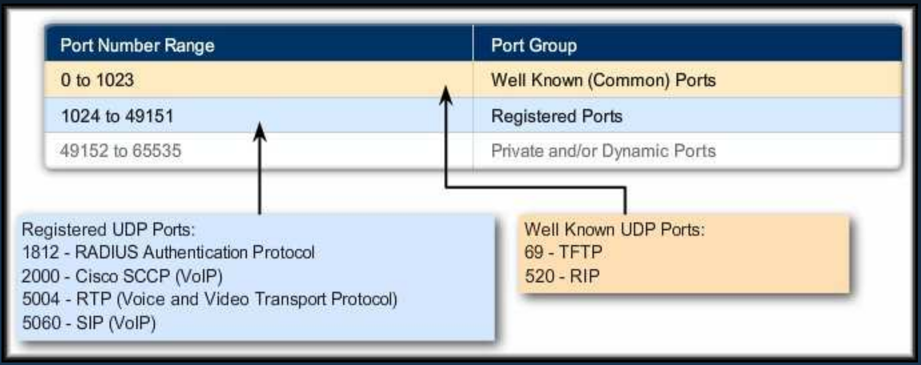
Port Number Range	Port Group
0 to 1023	Well Known (Common) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Registered TCP Ports:
1863 - MSN Messenger
8008 - Alternate HTTP
8080 - Alternate HTTP

Well Known TCP Ports
21 - FTP
23 - Telnet
25 - SMTP
80 - HTTP
110 - POP3
194 - Internet Relay Chat (IRC)
443 - Secure HTTP (HTTPS)

Using ACLs to Secure Networks

- The TCP data segment also identifies the port matching the requested service…..UDP



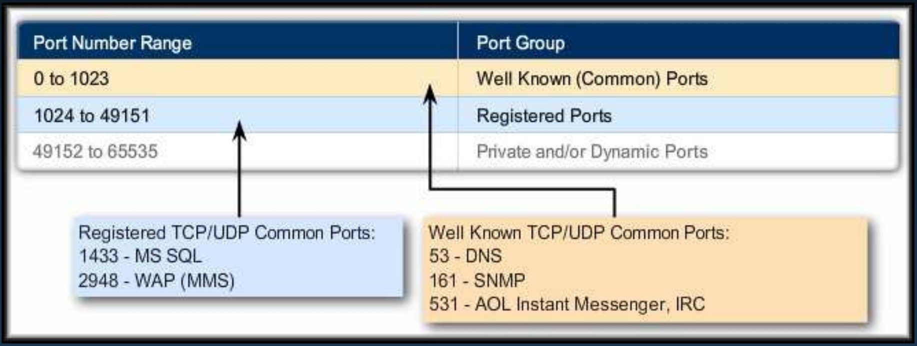
Port Number Range	Port Group
0 to 1023	Well Known (Common) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Registered UDP Ports:
1812 - RADIUS Authentication Protocol
2000 - Cisco SCCP (VoIP)
5004 - RTP (Voice and Video Transport Protocol)
5060 - SIP (VoIP)

Well Known UDP Ports:
69 - TFTP
520 - RIP

Using ACLs to Secure Networks

- The TCP data segment also identifies the port matching the requested service.....Common



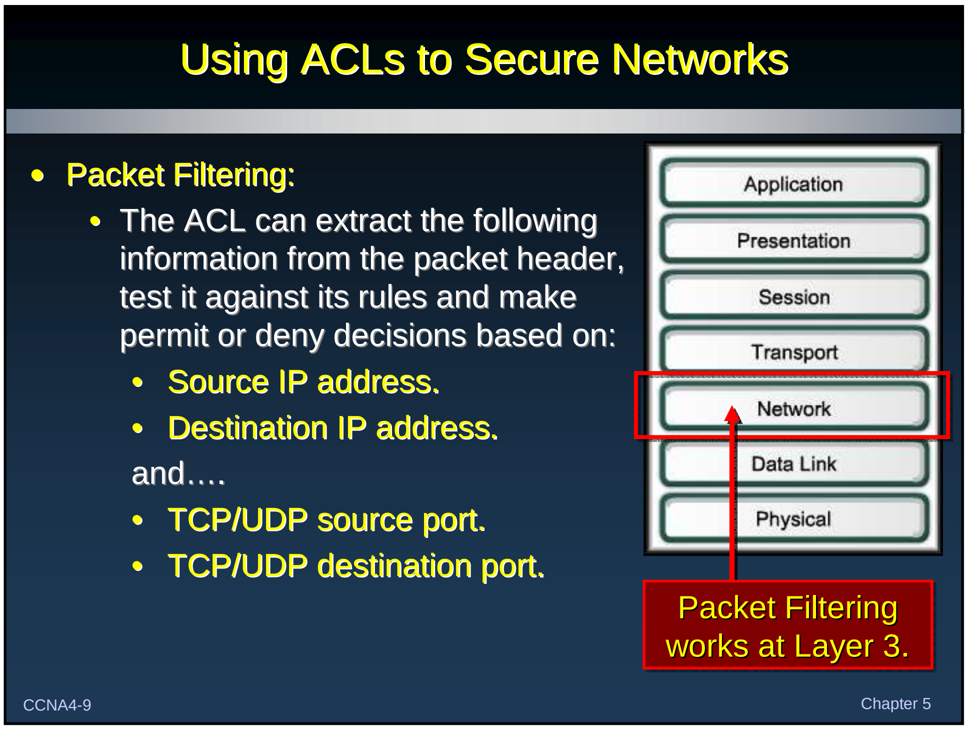
Port Number Range	Port Group
0 to 1023	Well Known (Common) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Registered TCP/UDP Common Ports:
1433 - MS SQL
2948 - WAP (MMS)

Well Known TCP/UDP Common Ports:
53 - DNS
161 - SNMP
531 - AOL Instant Messenger, IRC

Using ACLs to Secure Networks

- Packet Filtering:
  - Controls access to a network by analyzing the incoming and outgoing packets and passing or halting them based on stated criteria.
    - *These criteria are defined using ACLs.*
    - *An Access Control List (ACL) is a sequential list of permit or deny statements that apply to IP addresses or upper-layer protocols.*

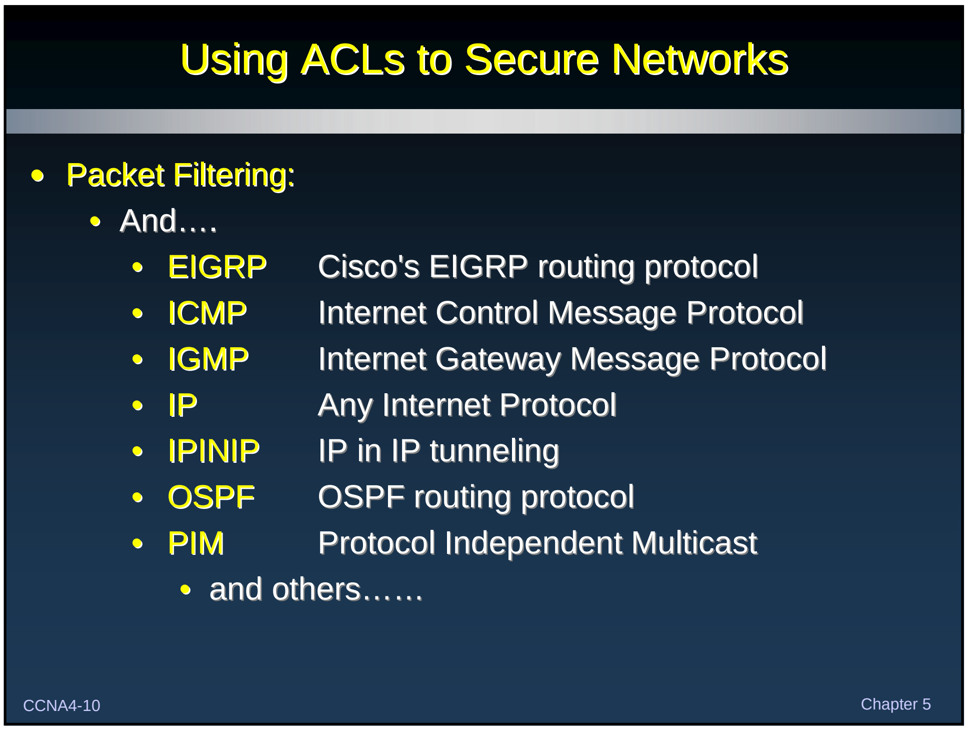


Using ACLs to Secure Networks

- Packet Filtering:
  - The ACL can extract the following information from the packet header, test it against its rules and make permit or deny decisions based on:
    - Source IP address.
    - Destination IP address.

    and….
    - TCP/UDP source port.
    - TCP/UDP destination port.

Application
Presentation
Session
Transport
Network
Data Link
Physical
Packet Filtering works at Layer 3.



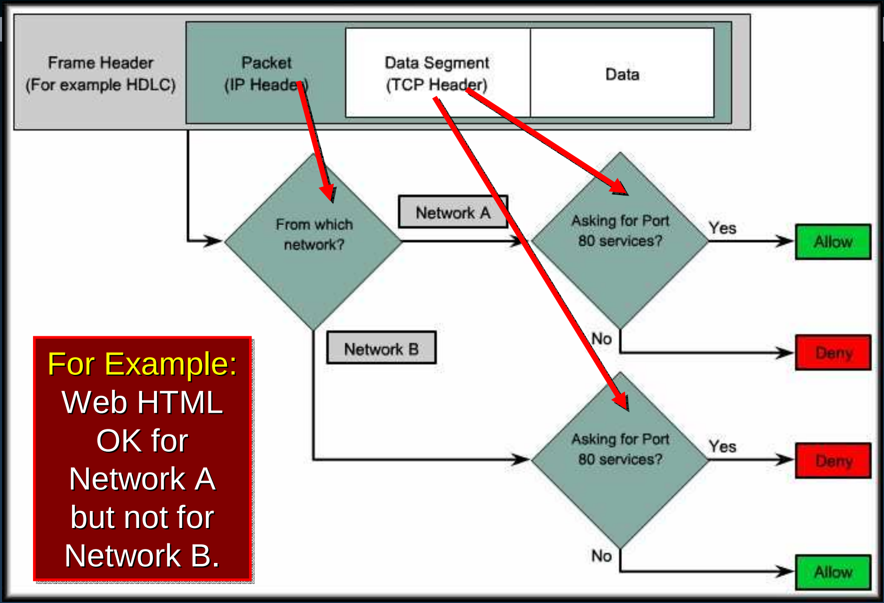
Using ACLs to Secure Networks

- Packet Filtering:
  - And….

EIGRP	Cisco's EIGRP routing protocol
ICMP	Internet Control Message Protocol
IGMP	Internet Gateway Message Protocol
IP	Any Internet Protocol
IPINIP	IP in IP tunneling
OSPF	OSPF routing protocol
PIM	Protocol Independent Multicast

- and others……

Using ACLs to Secure Networks


Frame Header (For example HDLC)
Packet (IP Header)
Data Segment (TCP Header)
Data
From which network?
Network A
Network B
Asking for Port 80 services?
Yes
No
Allow
Deny
Asking for Port 80 services?
Yes
No
Deny
Allow

For Example: Web HTML OK for Network A but not for Network B.

What is an ACL?

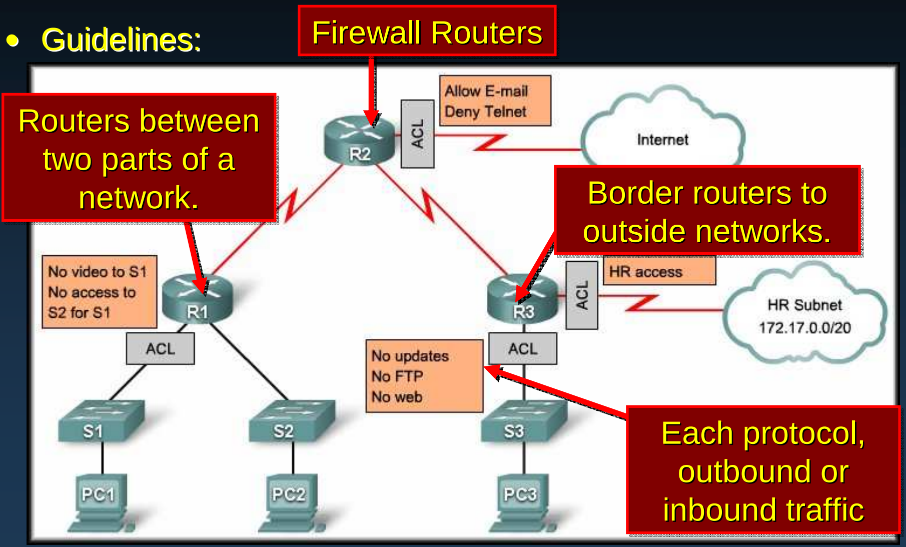
- An Access Control List (ACL) is:
  - A sequential list of permit or deny statements.
    - Apply to IP addresses (Layer 3 header)
    - Apply to upper-layer protocols (Layer 4 header).
  - Controls whether a router permits or denies packets to pass through the router.
  - A commonly used object in the Cisco IOS.
    - Also used to select certain types of traffic to be analyzed, forwarded or processed.
      - e.g. Network Address Translation (NAT), securing Telnet or SSH access to the router.

What is an ACL?

- By default, a router does not have any ACLs.
  - As each packet comes through an interface with an associated ACL:
    - The ACL is checked from top to bottom.
      - One line at a time.
    - Matches the pattern defined in the ACL statement to the specified area of the incoming packet.
    - Stops checking when it finds a matching statement.
      - Takes the defined action (permit or deny).
    - If no match is present, the default is to deny the packet.

What is an ACL?

- Guidelines:



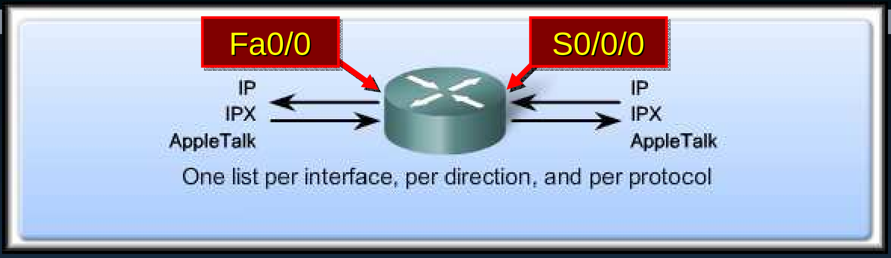
The Three P's

- ACL Functions: (Why do we need them?)
  - Limit network traffic and increase network performance.
  - Provide traffic flow control.
  - Provide a basic level of security for network access.
  - Decide which types of traffic are forwarded or blocked at the router interfaces.
  - Allow an administrator to control what areas a client can access on a network.
  - Screen certain hosts to either allow or deny access to part of a network.
  - Grant or deny user permission to access only certain types of files such as FTP or HTTP.

The Three P's

- One ACL per protocol:
  - An ACL must be defined for each protocol enabled on the interface.
- One ACL per direction:
  - ACLs control traffic in one direction at a time on an interface.
    - Two separate ACLs must be created to control:
      - Inbound Traffic: Traffic coming into the interface.
      - Outbound Traffic: Traffic leaving an interface.
- One ACL per interface:
  - ACLs control traffic for an interface (Fa0/0, s0/0/0).

The Three P's

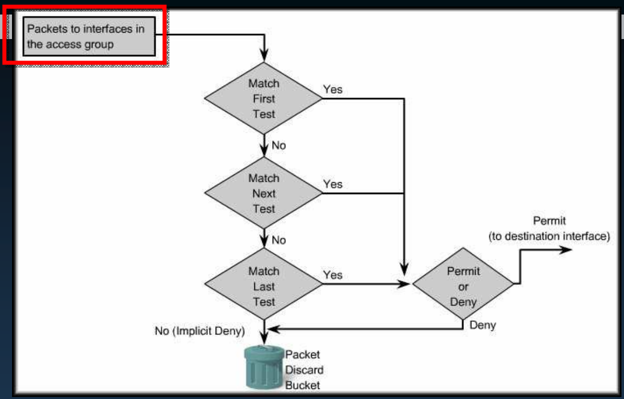


- One Access Control List per **protocol**.
- One Access Control List per **direction**.
- One Access Control List per **interface**.

- How many possible ACLs?
  - 3 protocols X 2 directions X 2 ports
  - Possibility of 12 separate lists.
    - *Note that the same list can be used on multiple interfaces.*

How ACLs Work

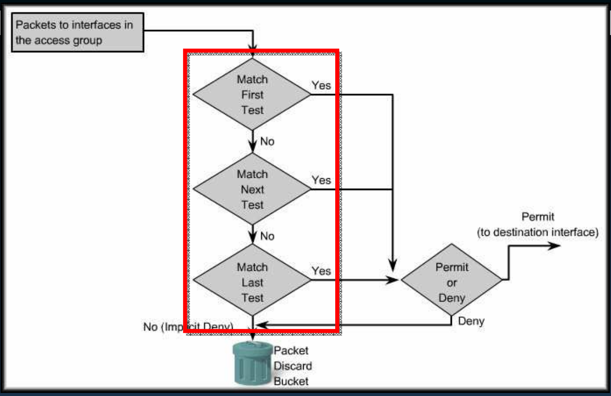
Inbound ACL



- The **access group** command is used to assign the list to the interface and specify the direction of the traffic to be checked.

How ACLs Work

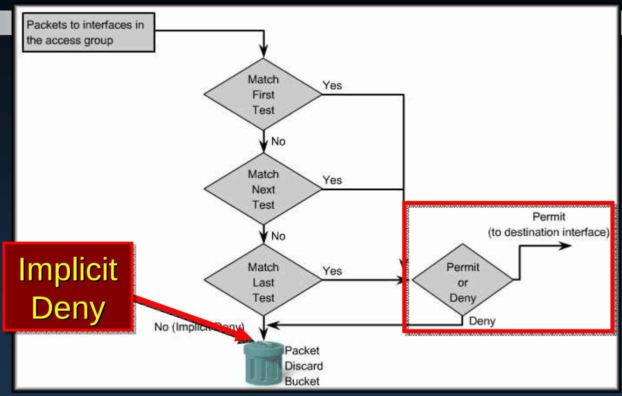
Inbound ACL



- ACL statements are processed in a **sequential, logical order**.
- The **logic used to create the list** and the **order of the list** items is very important.

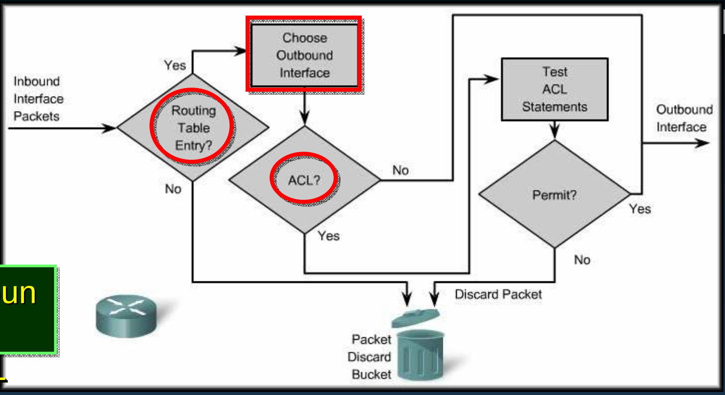
How ACLs Work

Inbound ACL



- If a condition match is **true**, the packet is permitted or denied and the rest of the ACL statements are not checked.
- If all the ACL statements are unmatched, an implicit **deny any** statement is placed at the end of the list **by default**.

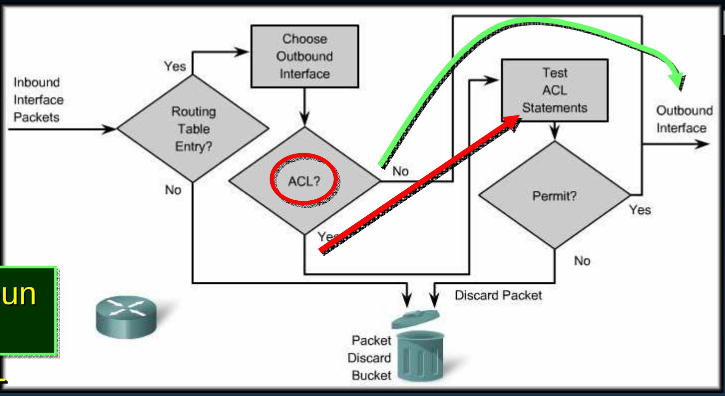
How ACLs Work



Outbound ACL

- Before a packet is forwarded to an outbound interface, the router checks the routing table.
- Next, the router checks to see whether the outbound interface is grouped to an ACL (access group command).

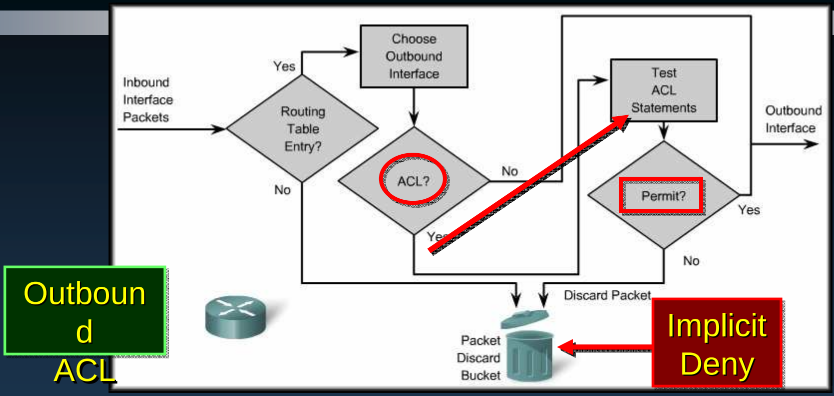
How ACLs Work



Outbound ACL

- If no ACL is present, the packet is forwarded out the interface.
- If an ACL is present, the packet is tested by the combination of ACL statements that are associated with that interface.

How ACLs Work



- The packet is either **permitted** (sent to the outbound interface) or **denied** (dropped).
- If the packet does not meet any of the criteria, it is **dropped** (Implicit Deny).

How ACLs Work

- Access list statements operate in **sequential, logical order**.
- They evaluate packets from the **top - down**.
- Once there is an **access list statement match,** the router skips the rest of the statements.
- If a condition **match is true**, the packet is permitted or denied.
- There can be **only one** access list **per protocol, per interface**.
- There is an **implicit deny any** at the end of every access list.

  - *ACLs do not block packets that originate within the router.* (i.e. pings, telnets, ssh, etc.)

Types of Cisco ACLs

- Two types:
  - Standard ACLs:
    - Standard ACLs allow you to permit or deny traffic based on the source IP addresses.
    - The destination of the packet and the ports involved do not matter.

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

    - Permit all traffic from network 192.168.30.0/24 network.
    - Because of the implied "deny any" at the end, all other traffic is blocked with this ACL.

Types of Cisco ACLs

- Two types:
  - Extended ACLs:
    - Extended ACLs filter IP packets based on several attributes;
      - Protocol type, source and/or destination IP address, source and/or destination TCP or UDP ports.

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

      - Permits traffic originating from any address on the 192.168.30.0/24 network to any destination host port 80 (HTTP).

Types of Cisco ACLs

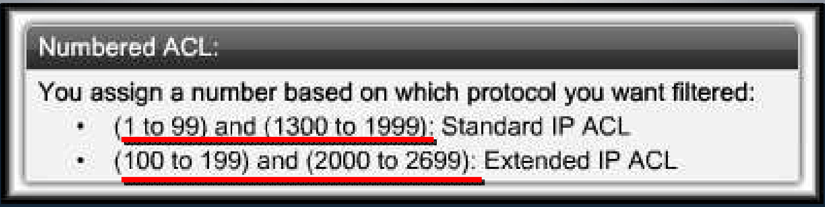
- FYI:
  - For either type:
    - Until you become proficient at creating ACLs it may be better to always add the implied deny any at the end of your list.
    - It may save you some grief.
- Standard:

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

- Extended:

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

Numbering and Naming ACLs

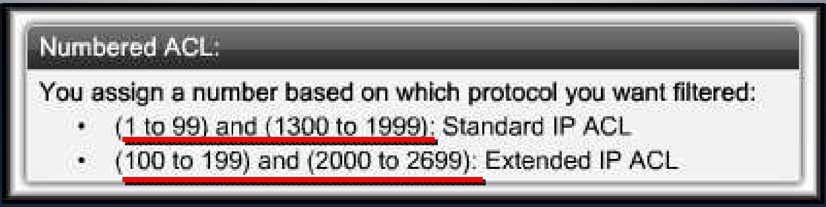

Numbered ACL:

You assign a number based on which protocol you want filtered:

- (1 to 99) and (1300 to 1999): Standard IP ACL
- (100 to 199) and (2000 to 2699): Extended IP ACL

- Using numbered ACLs is an effective method for determining the ACL type on smaller networks with more homogeneously defined traffic.

Numbering and Naming ACLs


Numbered ACL:

You assign a number based on which protocol you want filtered:

- (1 to 99) and (1300 to 1999): Standard IP ACL
- (100 to 199) and (2000 to 2699): Extended IP ACL

- When configuring ACLs on a router, each ACL must be uniquely identified by assigning a number.

One group numbered 8

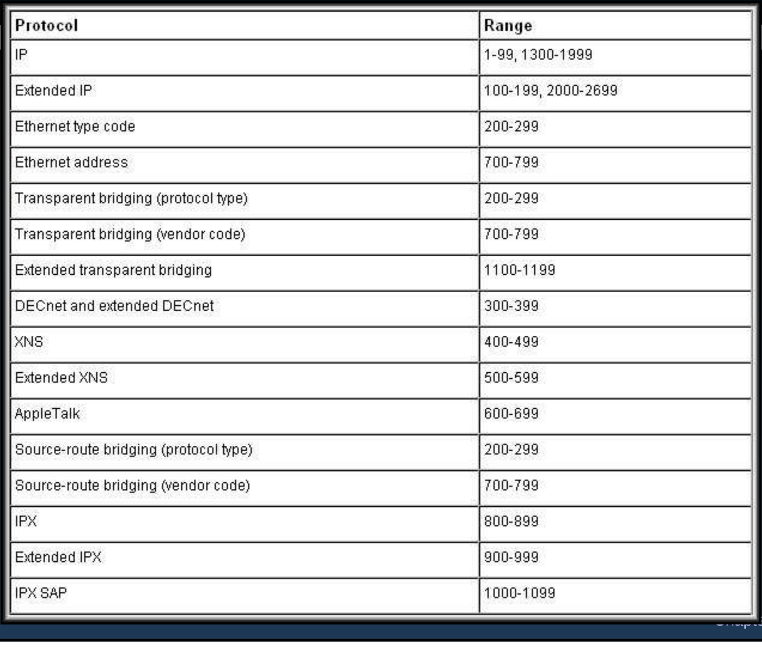
```
access list 8 permit…
access list 8 permit…
access list 8 permit…
access list 8 permit…
```

Multiple groups

```
access list 1 permit…
access list 2 permit…
access list 3 permit…
access list 4 permit…
```

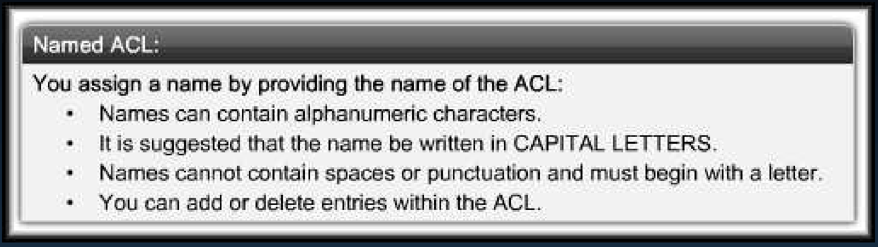
Numbering and Naming ACLs

FYI



Protocol	Range
IP	1-99, 1300-1999
Extended IP	100-199, 2000-2699
Ethernet type code	200-299
Ethernet address	700-799
Transparent bridging (protocol type)	200-299
Transparent bridging (vendor code)	700-799
Extended transparent bridging	1100-1199
DECnet and extended DECnet	300-399
XNS	400-499
Extended XNS	500-599
AppleTalk	600-699
Source-route bridging (protocol type)	200-299
Source-route bridging (vendor code)	700-799
IPX	800-899
Extended IPX	900-999
IPX SAP	1000-1099

Numbering and Naming ACLs


Named ACL:

You assign a name by providing the name of the ACL:

- Names can contain alphanumeric characters.
- It is suggested that the name be written in CAPITAL LETTERS.
- Names cannot contain spaces or punctuation and must begin with a letter.
- You can add or delete entries within the ACL.

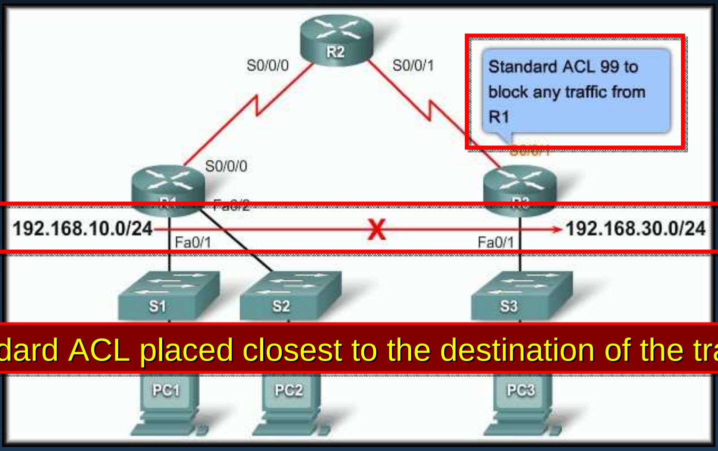
- Using named ACLs:
  - A numbered ACL does not tell you the purpose of the list.
  - Starting with Cisco IOS Release 11.2, you can use a name to identify a Cisco ACL.

Where to Place ACLs

- ACLs can act as firewalls to filter packets and eliminate unwanted traffic.
  - Every ACL should be placed where it has the greatest impact on efficiency.
  - The basic rules are:
    - Standard ACLs do not specify a destination address. Place them as close to the destination as possible.
    - Extended ACLs are located as close as possible to the source of the traffic denied.
      - Undesirable traffic is filtered without crossing the network infrastructure.

Where to Place ACLs

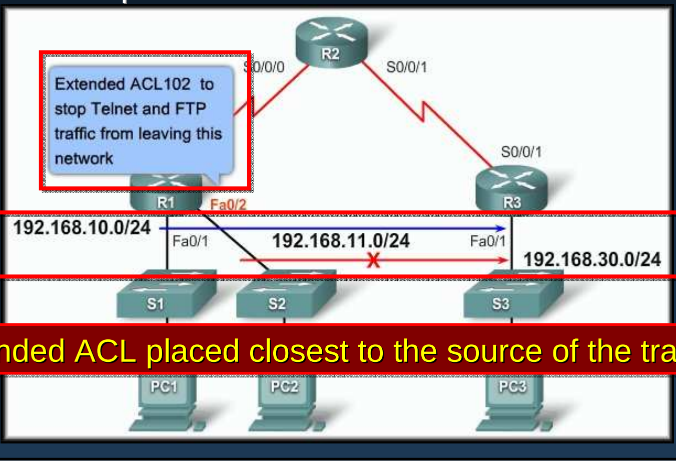
- The administrator wants to prevent traffic originating in the 192.168.10.0/24 network from getting to 192.168.30.0/24.



Standard ACL placed closest to the destination of the traffic.

Where to Place ACLs

- The administrator wants to deny Telnet and FTP traffic from 192.168.11.0/24 to 192.168.30.0/24. At the same time, other traffic must be permitted to leave 192.168.10.0/24.



Extended ACL placed closest to the source of the traffic.

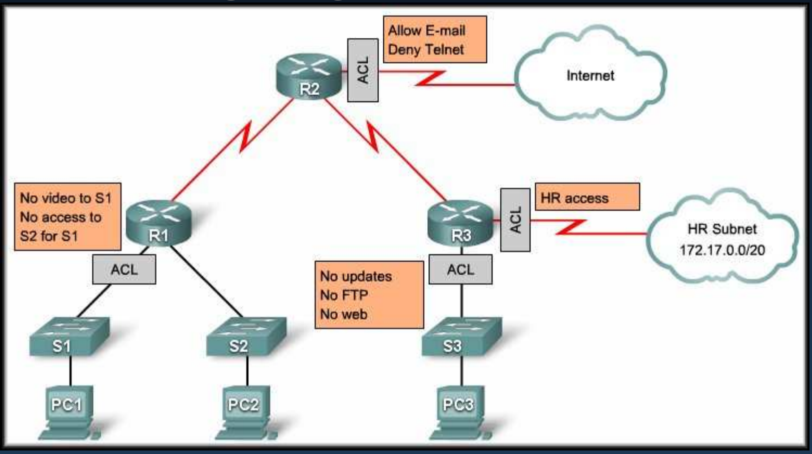
General Guidelines for Creating ACLs

- ACL Best Practices:

Guideline	Benefit
Base your ACLs on the security policy of the organization.	This will ensure you implement organizational security guidelines.
Prepare a description of what you want your ACLs to do.	This will help you avoid inadvertently creating potential access problems.
Use a text editor to create, edit and save ACLs.	This will help you create a library of reusable ACLs.
Test your ACLs on a development network before implementing them on a production network.	This will help you avoid costly errors.

Access Control Lists

Configuring Standard ACLS



Configuring Standard ACLs

- Entering Criteria Statements:
  - Traffic is compared to ACL statements based on the order that the entries occur in the router.
  - The router continues to process the ACL statements until it has a match.
    - You should have the most frequently used ACL entry at the top of the list.
    - If no matches are found when the router reaches the end of the list, the traffic is denied because there is an implied deny for traffic.

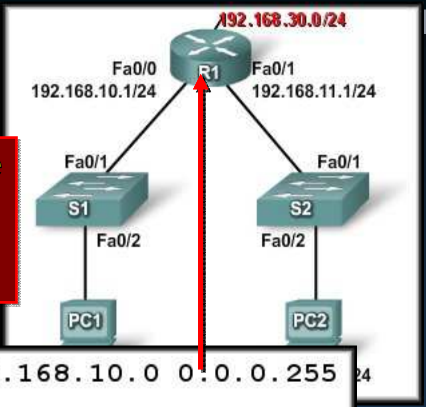
Configuring Standard ACLs

- Entering Criteria Statements:
  - Traffic is compared to ACL statements based on the order that the entries occur in the router.
    - *A single-entry ACL with only one deny entry has the effect of denying all traffic.*
    - You must have at least one permit statement in an ACL or all traffic is blocked.

Configuring Standard ACLs

- Entering Criteria Statements:

Either list would have the same affect for traffic to 192.168.30.0. 192.168.10.0 allowed, 192.168.11.0 blocked.

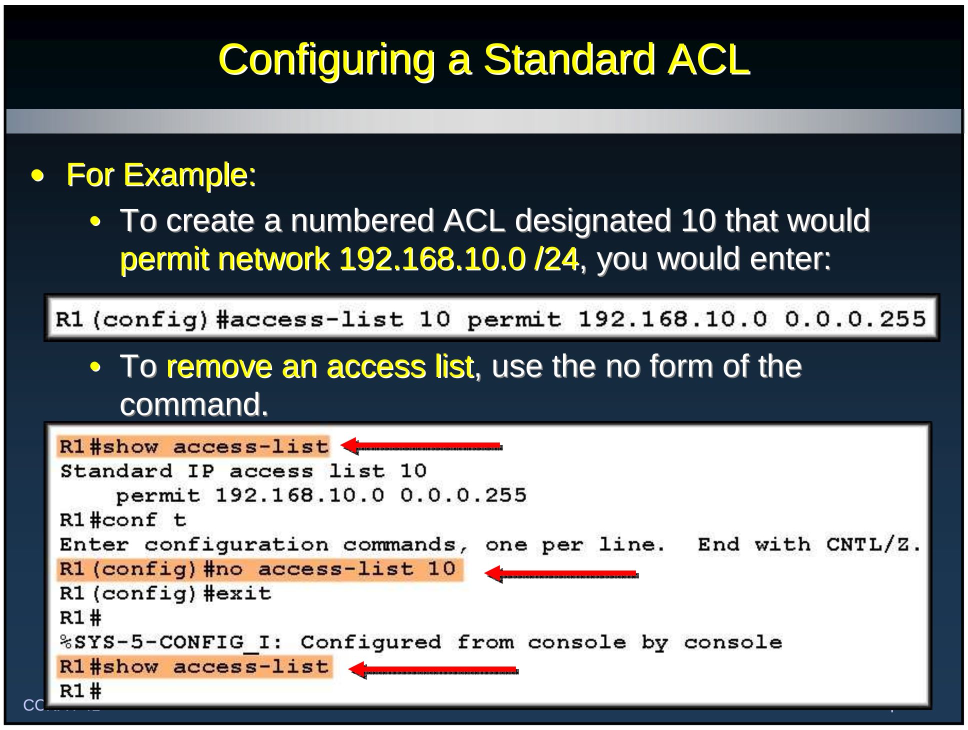


```
access-list 1 permit 192.168.10.0 0.0.0.255

----   OR   ----

access-list 2 permit 192.168.10.0 0.0.0.255
deny any
```

Configuring a Standard ACL

- To configure a standard ACL you must:
  - Create the standard ACL
  - Activate the ACL on an interface.
    - The **access-list** global configuration command defines a standard ACL with a number in the range of 1 to 99 or 1300 to 1399.

```
Router(config)#access-list access-list-number
                           [deny | permit | remark]
                           source [source-wildcard]
                           [log]
```



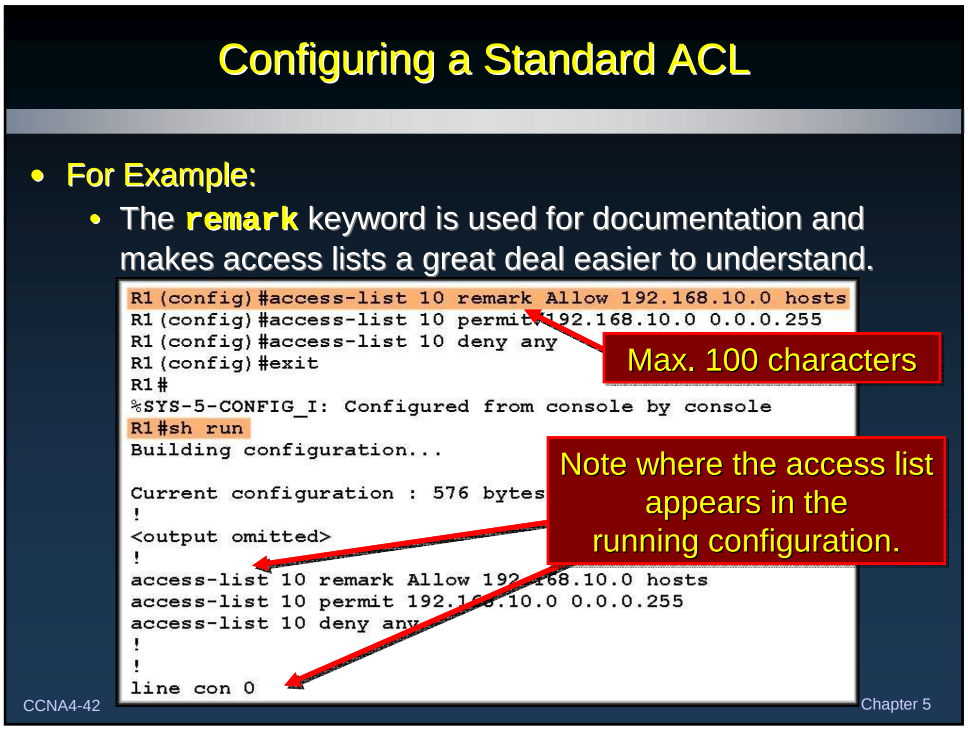
Configuring a Standard ACL

- For Example:
  - To create a numbered ACL designated 10 that would permit network 192.168.10.0 /24, you would enter:

```
R1(config)#access-list 10 permit 192.168.10.0 0.0.0.255
```

  - To remove an access list, use the no form of the command.

```
R1#show access-list
Standard IP access list 10
    permit 192.168.10.0 0.0.0.255
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#no access-list 10
R1(config)#exit
R1#
%SYS-5-CONFIG_I: Configured from console by console
R1#show access-list
R1#
```



Configuring a Standard ACL

- For Example:
  - The **remark** keyword is used for documentation and makes access lists a great deal easier to understand.

```
R1(config)#access-list 10 remark Allow 192.168.10.0 hosts
R1(config)#access-list 10 permit 192.168.10.0 0.0.0.255
R1(config)#access-list 10 deny any
R1(config)#exit
R1#
%SYS-5-CONFIG_I: Configured from console by console
R1#sh run
Building configuration...

Current configuration : 576 bytes
!
<output omitted>
!
access-list 10 remark Allow 192.168.10.0 hosts
access-list 10 permit 192.168.10.0 0.0.0.255
access-list 10 deny any
!
!
line con 0
```

Max. 100 characters

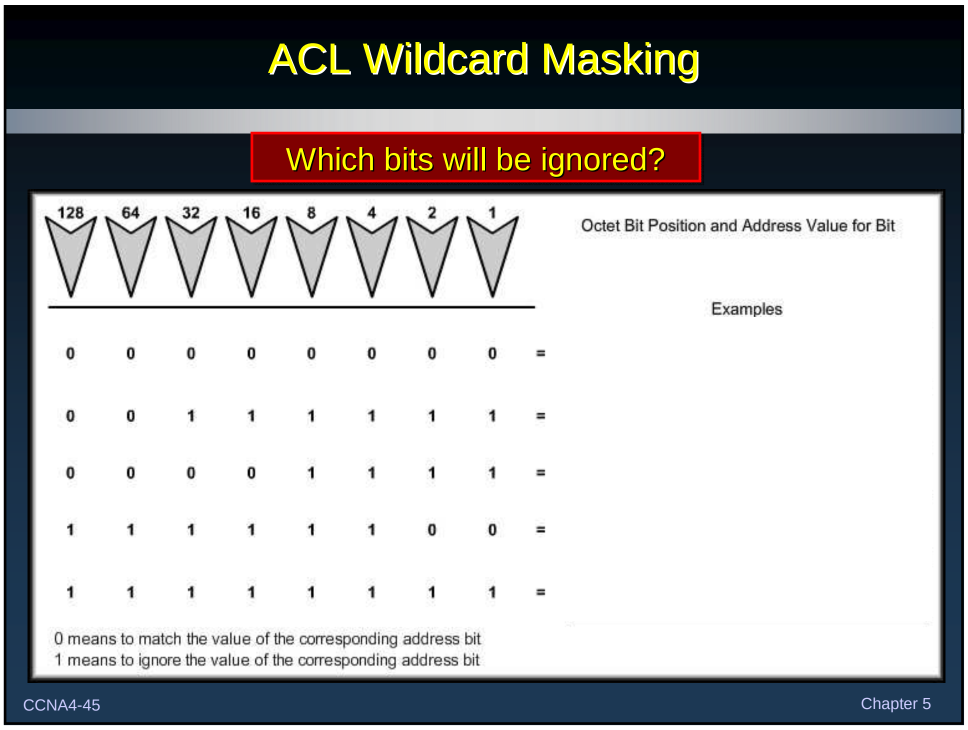
Note where the access list appears in the running configuration.

ACL Wildcard Masking

- Wildcard Masking:
  - ACLs statements include wildcard masks.
    - (Remember OSPF network entries?)
  - A wildcard mask is a string of binary digits telling the router to check specific parts of the subnet number.
    - The numbers 1 and 0 in the mask identify how to treat the corresponding IP address bits.
  - Wildcard masks are referred to as an inverse mask.
    - Unlike a subnet mask in which binary 1 is equal to a match (network) and binary 0 is not a match (host), the reverse is true.
    - It also does not have to be contiguous 1's and 0's.

ACL Wildcard Masking

- Wildcard Masking:
  - Wildcard masks use the following rules to match binary 1s and 0s:
    - Wildcard mask bit 0:
      - The corresponding bit value in the IP Address to be tested must match the bit value in the address specified in the ACL.
    - Wildcard mask bit 1:
      - Ignore the corresponding bit value.



ACL Wildcard Masking

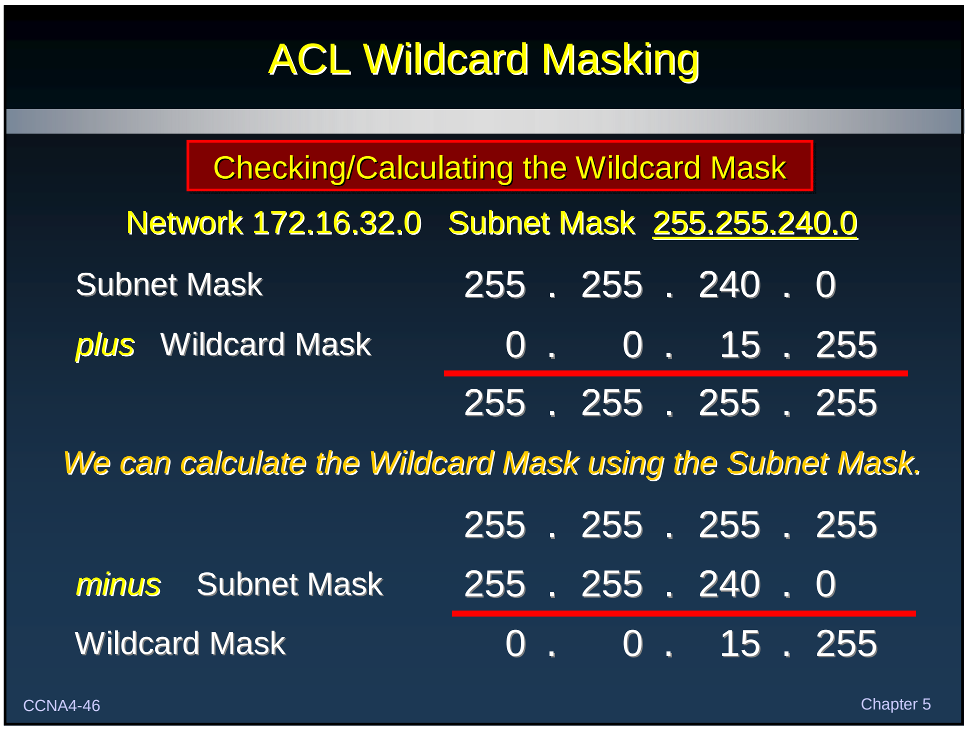
Which bits will be ignored?

Octet Bit Position and Address Value for Bit

Examples

128	64	32	16	8	4	2	1	
0	0	0	0	0	0	0	0	=
0	0	1	1	1	1	1	1	=
0	0	0	0	1	1	1	1	=
1	1	1	1	1	1	0	0	=
1	1	1	1	1	1	1	1	=

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



ACL Wildcard Masking

Checking/Calculating the Wildcard Mask

Network 172.16.32.0 Subnet Mask <u>255.255.240.0</u>

Subnet Mask	255 . 255 . 240 . 0
plus Wildcard Mask	0 . 0 . 15 . 255
	255 . 255 . 255 . 255

We can calculate the Wildcard Mask using the Subnet Mask.

	255 . 255 . 255 . 255
minus Subnet Mask	255 . 255 . 240 . 0
Wildcard Mask	0 . 0 . 15 . 255

Time for some Practice!

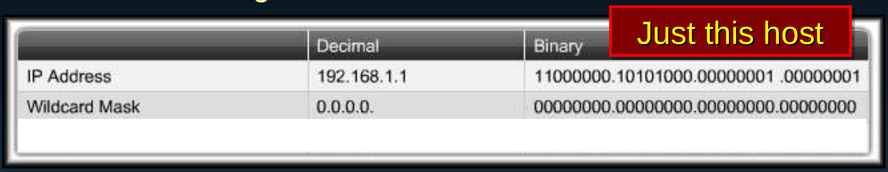
```
RouterB(config)#access-list 10 permit  ?   ?
```

	Permit the following networks:	Address / Wildcard Mask
A	172.16.0.0 255.255.0.0	172.16.0.0 0.0.255.255
B	172.16.1.0 255.255.255.0	172.16.1.0 0.0.0.255
C	192.168.1.0 255.255.255.0	192.168.1.0 0.0.0.255
D	172.16.16.0 255.255.240.0	172.16.32.0 0.0.15.255
E	172.16.128.0 255.255.192.0	172.16.128.0 0.0.63.255

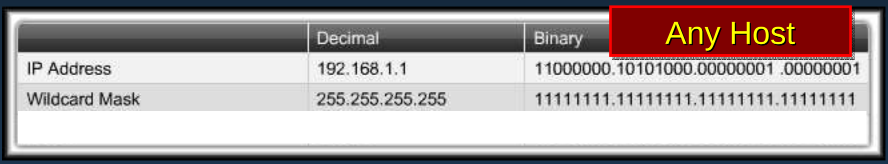
	Permit the following hosts:	
A	172.16.10.100	172.16.10.100 0.0.0.0
B	192.168.1.100	192.168.1.100 0.0.0.0
C	All hosts	0.0.0.0 255.255.255.255

ACL Wildcard Masking

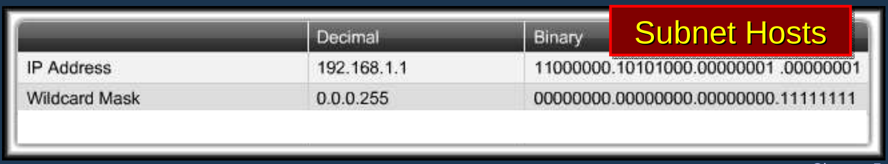
- Wildcard Masking:


Just this host

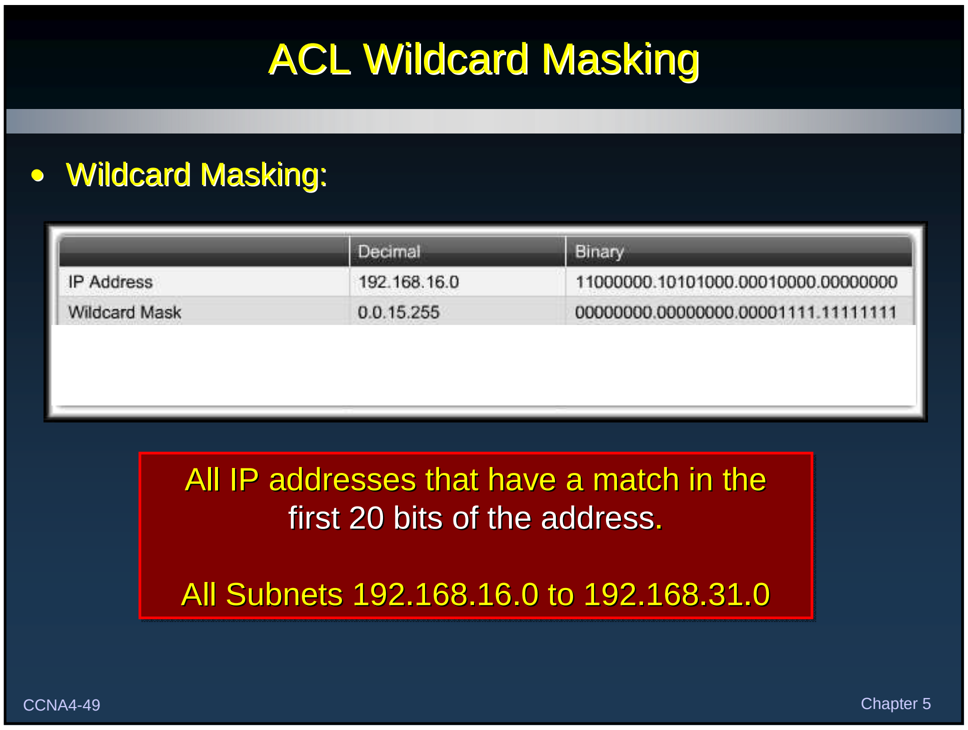
	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001 .00000001
Wildcard Mask	0.0.0.0.	00000000.00000000.00000000.00000000


Any Host

	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001 .00000001
Wildcard Mask	255.255.255.255	11111111.11111111.11111111.11111111


Subnet Hosts

	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001 .00000001
Wildcard Mask	0.0.0.255	00000000.00000000.00000000.11111111



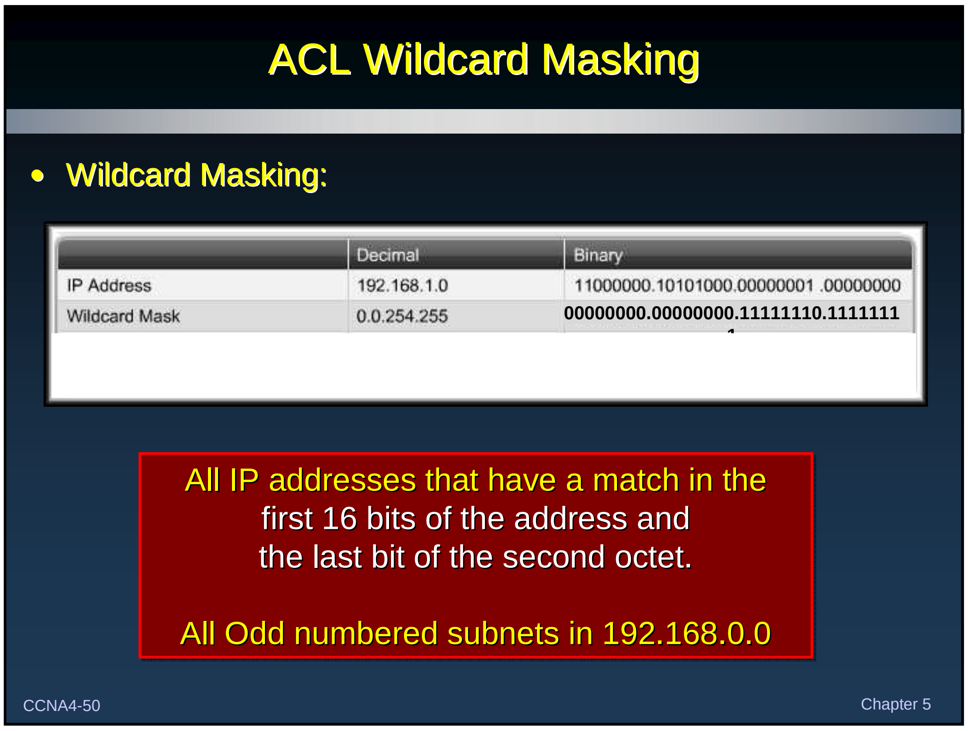
ACL Wildcard Masking

- Wildcard Masking:

	Decimal	Binary
IP Address	192.168.16.0	11000000.10101000.00010000.00000000
Wildcard Mask	0.0.15.255	00000000.00000000.00001111.11111111

All IP addresses that have a match in the first 20 bits of the address.

All Subnets 192.168.16.0 to 192.168.31.0



ACL Wildcard Masking

- Wildcard Masking:

	Decimal	Binary
IP Address	192.168.1.0	11000000.10101000.00000001 .00000000
Wildcard Mask	0.0.254.255	00000000.00000000.11111110.1111111 1

All IP addresses that have a match in the first 16 bits of the address and the last bit of the second octet.

All Odd numbered subnets in 192.168.0.0

ACL Wildcard Masking

- Wildcard Bit Mask Keywords:
  - The keywords host and any help identify the most common uses of wildcard masking.
    - host:
      - Used instead of 0.0.0.0 for the wildcard mask (all IP address bits must match).
    - any:
      - Used instead of 255.255.255.255 for the wildcard mask (accept any addresses).

ACL Wildcard Masking

- Wildcard Bit Mask Keywords:

```
access-list 10 permit 172.16.30.2 0.0.0.0
                     OR
access-list 10 permit host 172.16.30.2
                     OR
access-list 10 permit 172.16.30.2
```

```
access-list 10 deny 0.0.0.0 255.255.255.255
        OR
access-list 10 deny any

access-list 10 permit 0.0.0.0 255.255.255.255
        OR
access-list 10 permit any
```

Applying Standard ACLs to Interfaces

- You can define ACLs without applying them but they **will have no effect** until they are applied to the router's interface.
  - Remember……It is a good practice to:
    - Apply the Standard ACLs on the interface *closest to the destination of the traffic*.
    - Apply Extended ACLs on the interface *closest to the source of the traffic*.

Applying Standard ACLs to Interfaces

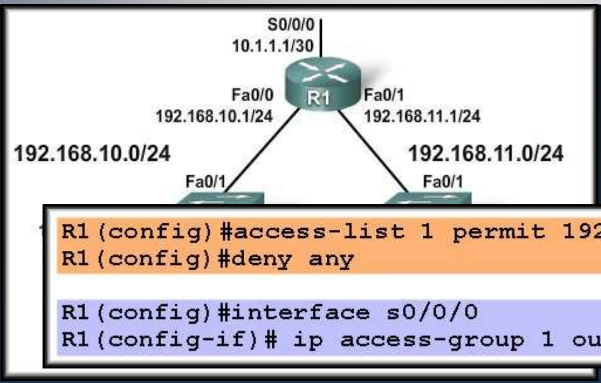
- Apply the standard ACL to an interface using the following command:

The number or name assigned during the **`access-list`** configuration.

```
R1(config-if)#access-group
                         {access-list number | access-list-name}
                         {in | out}
```

Consider the traffic from the router's viewpoint.
`in`: Traffic that is arriving on the interface.
`out`: Traffic that has already been routed by the router and is leaving the interface.

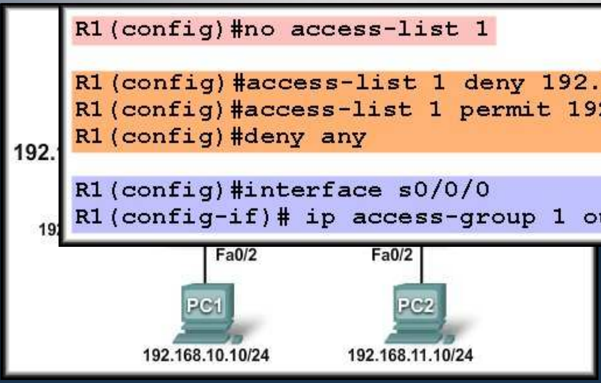
Applying Standard ACLs to Interfaces



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

R1(config)#interface s0/0/0
R1(config-if)# ip access-group 1 out
```

- Example 1:
  - Allow only traffic from network 192.168.10.0 to exit the network on S0/0/0. Block any traffic from any other network.

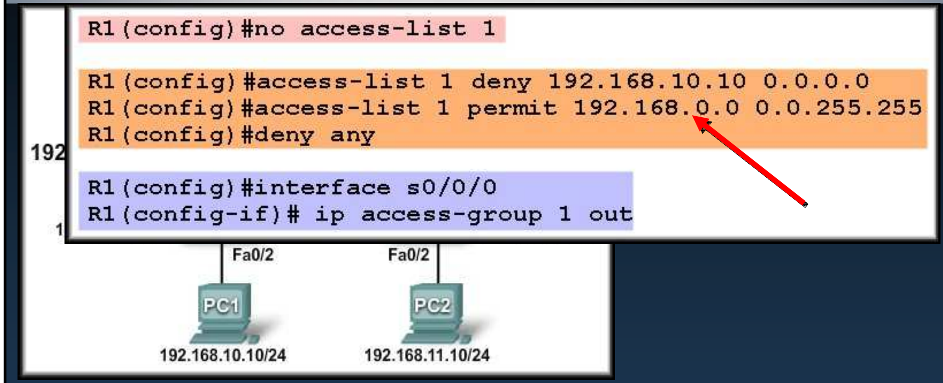
Applying Standard ACLs to Interfaces



```
R1(config)#no access-list 1

R1(config)#access-list 1 deny 192.168.10.10 0.0.0.0
R1(config)#access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)#deny any

R1(config)#interface s0/0/0
R1(config-if)# ip access-group 1 out
```

- Example 2:
  - Deny any traffic from host 192.168.10.10 and allow any other 192.160.10.0 traffic to exit the network on S0/0/0. Block any traffic from any other network.

Applying Standard ACLs to Interfaces



```
R1(config)#no access-list 1

R1(config)#access-list 1 deny 192.168.10.10 0.0.0.0
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255
R1(config)#deny any

R1(config)#interface s0/0/0
R1(config-if)# ip access-group 1 out
```

- Example 3:
  - Deny any traffic from host 192.168.10.10 and allow any other subnet traffic to exit the network on S0/0/0.

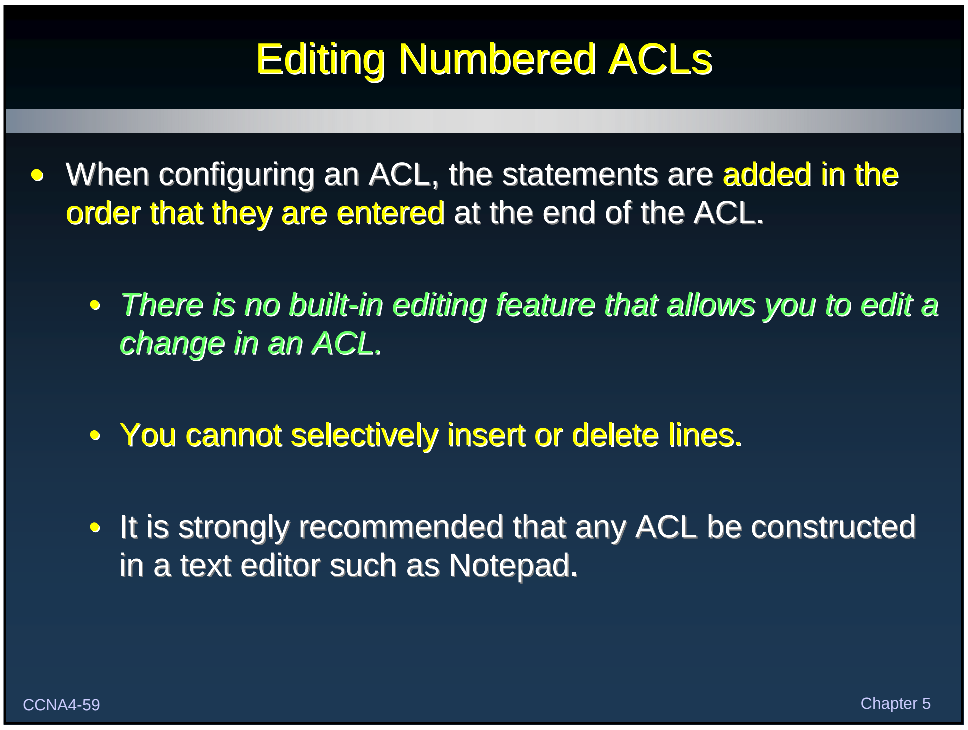
Applying Standard ACLs to Interfaces

- Using an ACL to Control VTY Access:
  - If your router does not support SSH, this technique allows you to define which IP addresses are allowed Telnet access to the router EXEC process.

```
access-class access-list-number {in [vrf-also] | out}
```

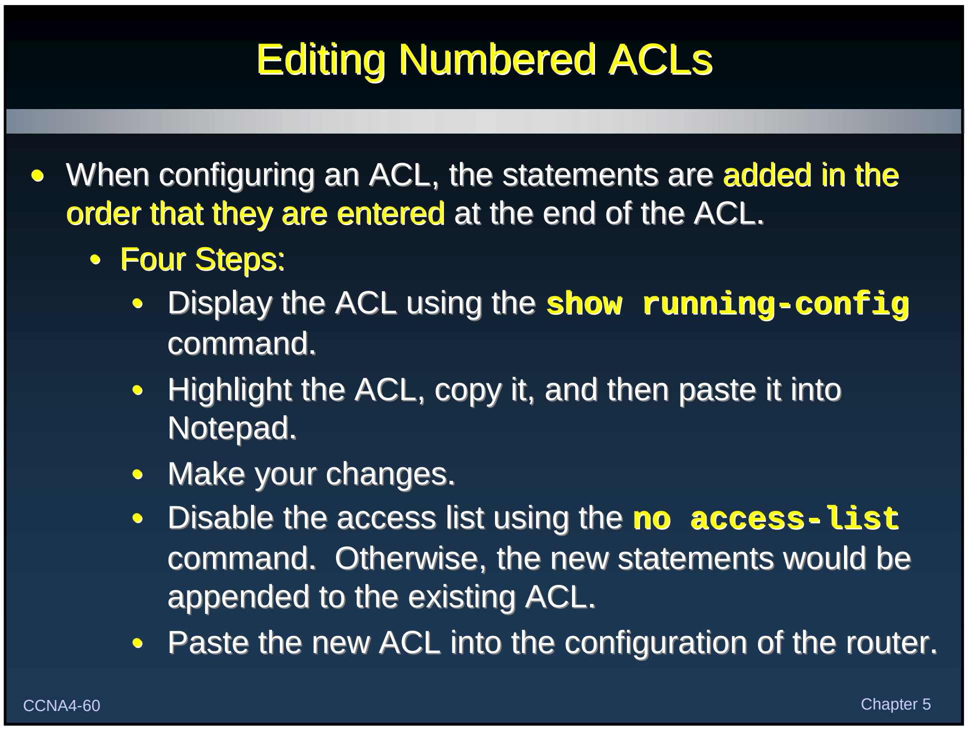
List number

in – restricts incoming connections
out – restricts outgoing connections



Editing Numbered ACLs

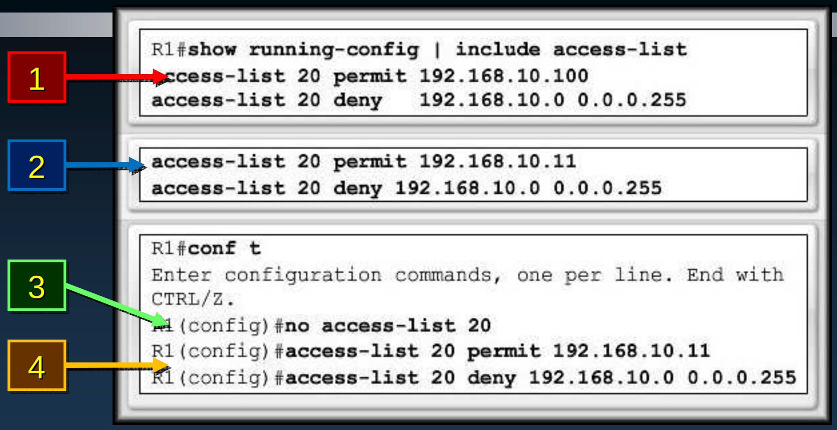
- When configuring an ACL, the statements are added in the order that they are entered at the end of the ACL.
  - *There is no built-in editing feature that allows you to edit a change in an ACL.*
  - You cannot selectively insert or delete lines.
  - It is strongly recommended that any ACL be constructed in a text editor such as Notepad.



Editing Numbered ACLs

- When configuring an ACL, the statements are added in the order that they are entered at the end of the ACL.
  - Four Steps:
    - Display the ACL using the **`show running-config`** command.
    - Highlight the ACL, copy it, and then paste it into Notepad.
    - Make your changes.
    - Disable the access list using the **`no access-list`** command. Otherwise, the new statements would be appended to the existing ACL.
    - Paste the new ACL into the configuration of the router.

Editing Numbered ACLs



```
R1#show running-config | include access-list
access-list 20 permit 192.168.10.100
access-list 20 deny   192.168.10.0 0.0.0.255
```

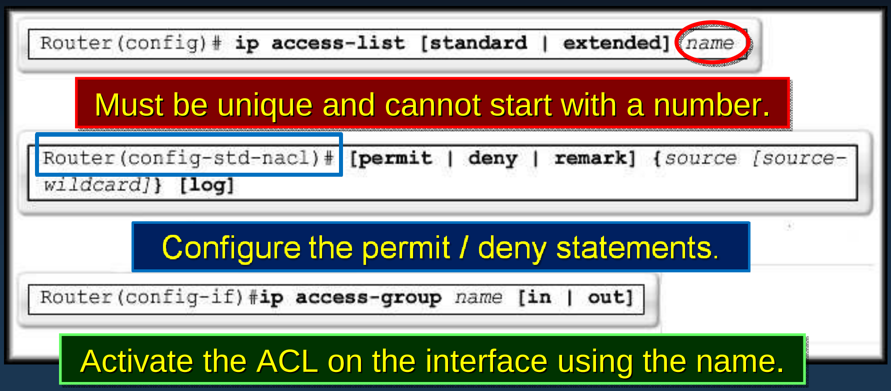
```
access-list 20 permit 192.168.10.11
access-list 20 deny 192.168.10.0 0.0.0.255
```

```
R1#conf t
Enter configuration commands, one per line. End with
CTRL/Z.
R1(config)#no access-list 20
R1(config)#access-list 20 permit 192.168.10.11
R1(config)#access-list 20 deny 192.168.10.0 0.0.0.255
```

- Be aware that when you use the **`no access-list`** command, no ACL is protecting your network.
  - If you make an error in the new list, you have to disable it and troubleshoot the problem.

Creating Standard Named ACLs

- Naming an ACL makes it easier to understand.



```
Router(config)# ip access-list [standard | extended] name
```

Must be unique and cannot start with a number.

```
Router(config-std-nacl)# [permit | deny | remark] {source [source-
wildcard]} [log]
```

Configure the permit / deny statements.

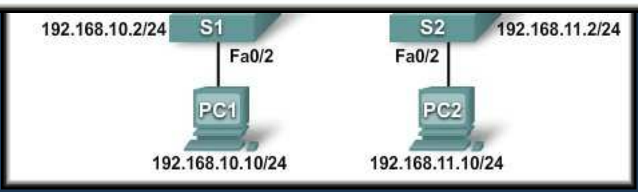
```
Router(config-if)#ip access-group name [in | out]
```

Activate the ACL on the interface using the name.

Creating Standard Named ACLs

- Naming an ACL makes it easier to understand.

```
R1(config)#ip access-list standard NO_ACCESS

R1(config-std-nacl)#deny host 192.168.11.10
R1(config-std-nacl)#permit 192.168.11.0 0.0.0.255
R1(config-std-nacl)#deny any
R1(config-std-nacl)#end

R1(config)#interface Fa0/1
R1(config-if)# ip access-group NO_ACCESS out
```



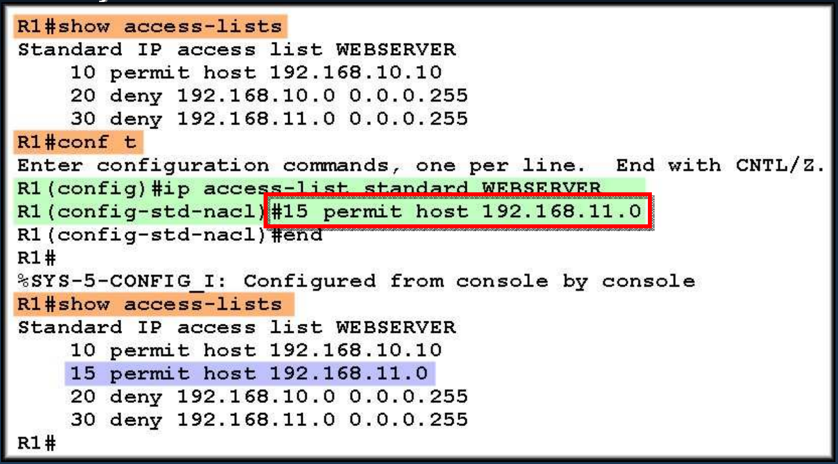
Monitoring and Verifying ACLs

```
R1#show access-lists

Standard IP access list SALES
    10 deny   10.1.1.0 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 192.168.10.10 any eq telnet (25 matches)
    20 permit tcp host 192.168.10.10 any eq ftp
    30 permit tcp host 192.168.10.10 any eq ftp-data
```

Remember that there is an implied **deny any** at the end of each access control list.

Editing Named ACLs

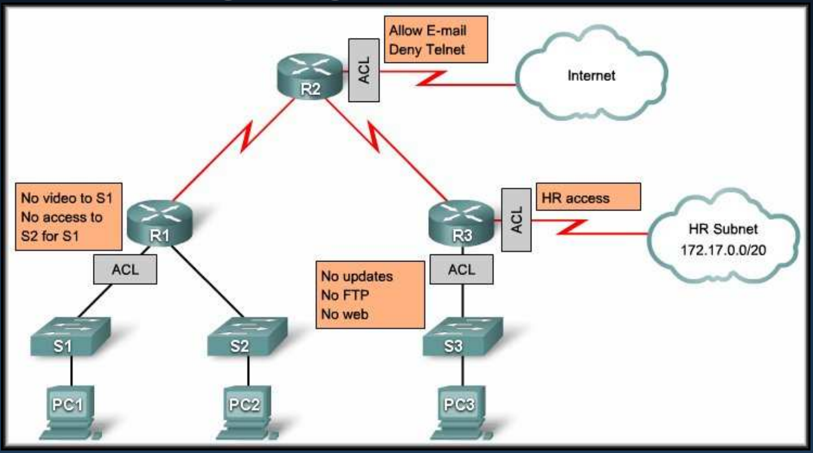
- Named ACLs have a big advantage over numbered ACLs in that they are easier to edit.



```
R1#show access-lists
Standard IP access list WEBSERVER
    10 permit host 192.168.10.10
    20 deny 192.168.10.0 0.0.0.255
    30 deny 192.168.11.0 0.0.0.255
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#ip access-list standard WEBSERVER
R1(config-std-nacl)#15 permit host 192.168.11.0
R1(config-std-nacl)#end
R1#
%SYS-5-CONFIG_I: Configured from console by console
R1#show access-lists
Standard IP access list WEBSERVER
    10 permit host 192.168.10.10
    15 permit host 192.168.11.0
    20 deny 192.168.10.0 0.0.0.255
    30 deny 192.168.11.0 0.0.0.255
R1#
```

Access Control Lists

Configuring Extended ACLS



Extended ACLs

- Extended ACLs are used more often than standard ACLs because they provide a greater range of control.
  - Extended ACLs can check:
    - Source packet address.
    - Destination address.
    - Protocol.
    - Port number or service.
  - Full Syntax:

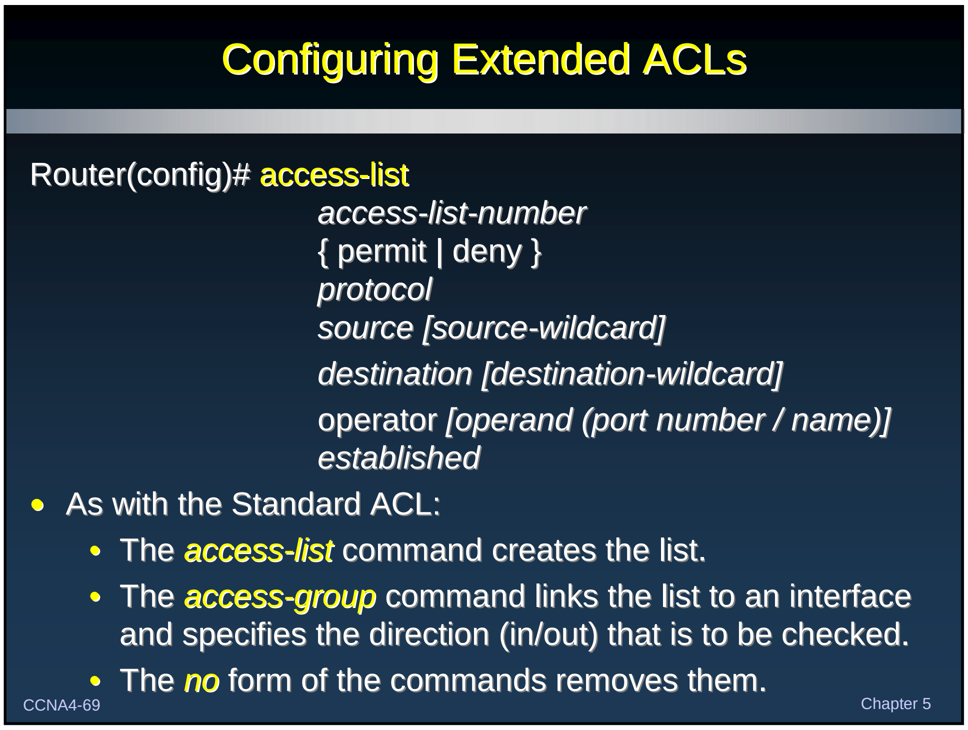
```
access-list access-list-number {deny | permit | remark}
protocol source [source-wildcard] [operator operand]
[port port-number or name] destination [destination-wildcard]
[operator operand] [port port-number or name] [established]
```

Extended ACLs

- The ability to filter on protocol and port number allows you to build very specific extended ACLs.

```
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 23
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 21
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 20
```

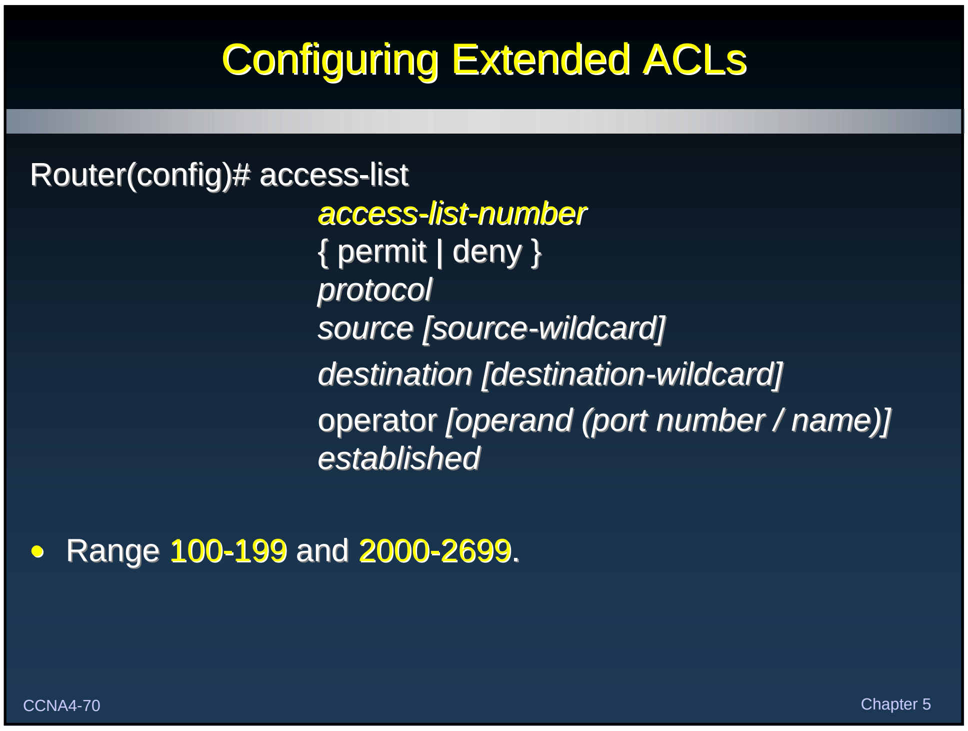
```
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq telnet
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp-data
```



Configuring Extended ACLs

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

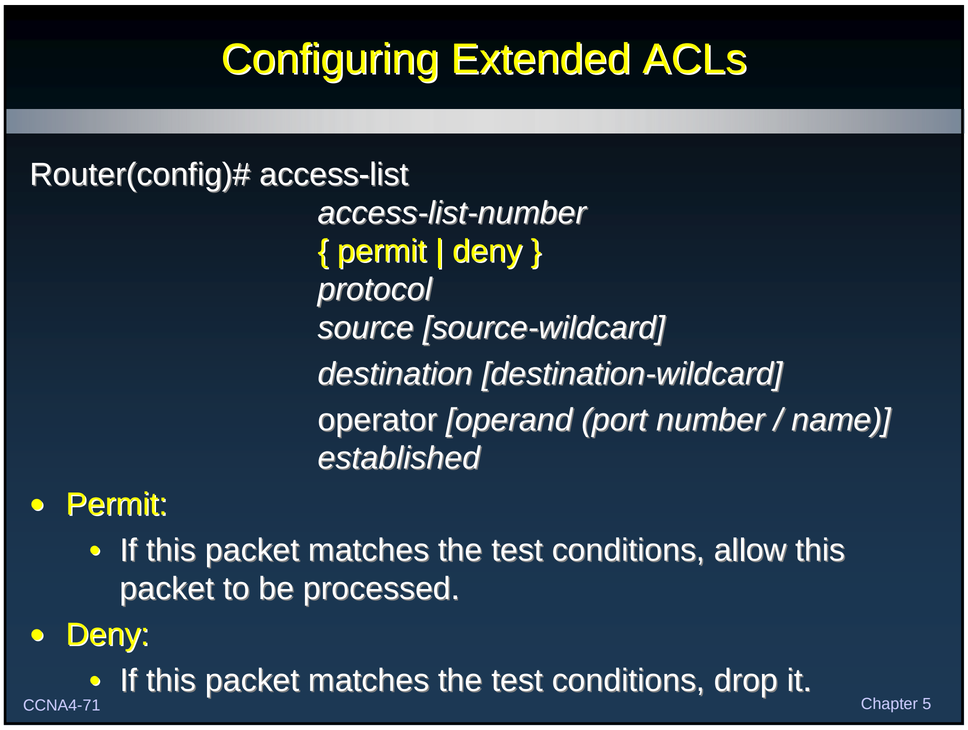
- As with the Standard ACL:
  - The *access-list* command creates the list.
  - The *access-group* command links the list to an interface and specifies the direction (in/out) that is to be checked.
  - The *no* form of the commands removes them.



Configuring Extended ACLs

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

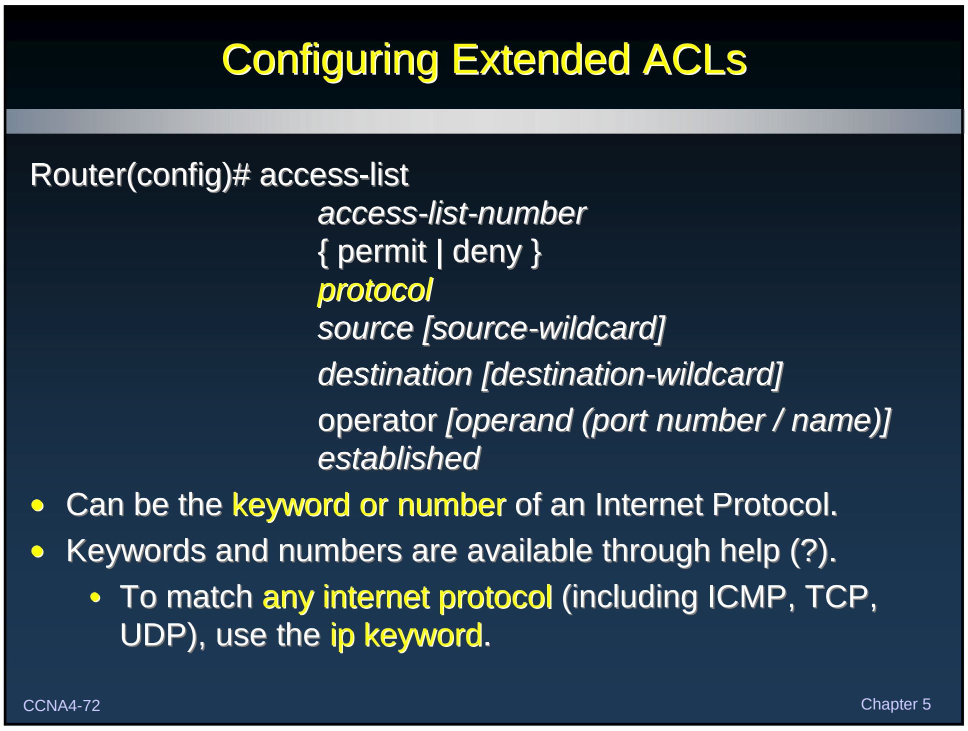
- Range 100-199 and 2000-2699.



Configuring Extended ACLs

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

- Permit:
  - If this packet matches the test conditions, allow this packet to be processed.
- Deny:
  - If this packet matches the test conditions, drop it.



Configuring Extended ACLs

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

- Can be the keyword or number of an Internet Protocol.
- Keywords and numbers are available through help (?).
  - To match any internet protocol (including ICMP, TCP, UDP), use the ip keyword.

Configuring Extended ACLs

Router(config)# access-list
access-list-number
{ permit | deny }
protocol
source [source-wildcard]
destination [destination-wildcard]
operator *[operand (port number / name)]*
established

- The source and destination IP address and wildcard mask.
- The format and usage of the wildcard mask is the same as in the standard ACL.
- The keywords any and host can be used in the same manner as the standard ACL.

Configuring Extended ACLs

Router(config)# access-list
access-list-number
{ permit | deny }
protocol
source [source-wildcard]
destination [destination-wildcard]
operator *[operand (port number / name)]*
established

- (Optional) compares the source or destination ports that are specified in the operand.
  - Includes: lt (less than), gt (greater than), eq (equal), neq (not equal) and range (inclusive range).

Configuring Extended ACLs

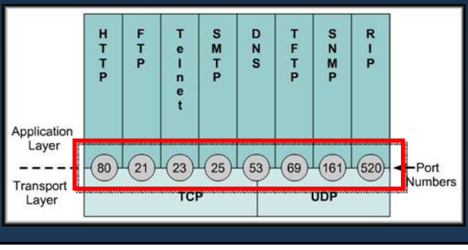
Router(config)# access-list
access-list-number
{ permit | deny }
protocol
source [source-wildcard]
destination [destination-wildcard]
operator *[operand (port number / name)]*
established

- If the operator and operand is positioned after the source and source-wildcard, it refers to the source port.
- If the operator and operand is positioned after the destination and destination-wildcard, it refers to the destination port.

Configuring Extended ACLs

Router(config)# access-list
access-list-number
{ permit | deny }
protocol
source [source-wildcard]
destination [destination-wildcard]
operator *[operand (port number / name)]*
established

- (Optional) The decimal number or name of a TCP or UDP port.



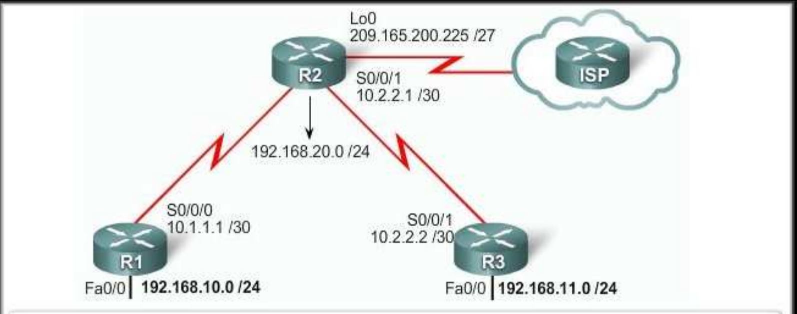
Configuring Extended ACLs

Router(config)# access-list
access-list-number
{ permit | deny }
protocol
source [source-wildcard]
destination [destination-wildcard]
operator *[operand (port number / name)]*
established

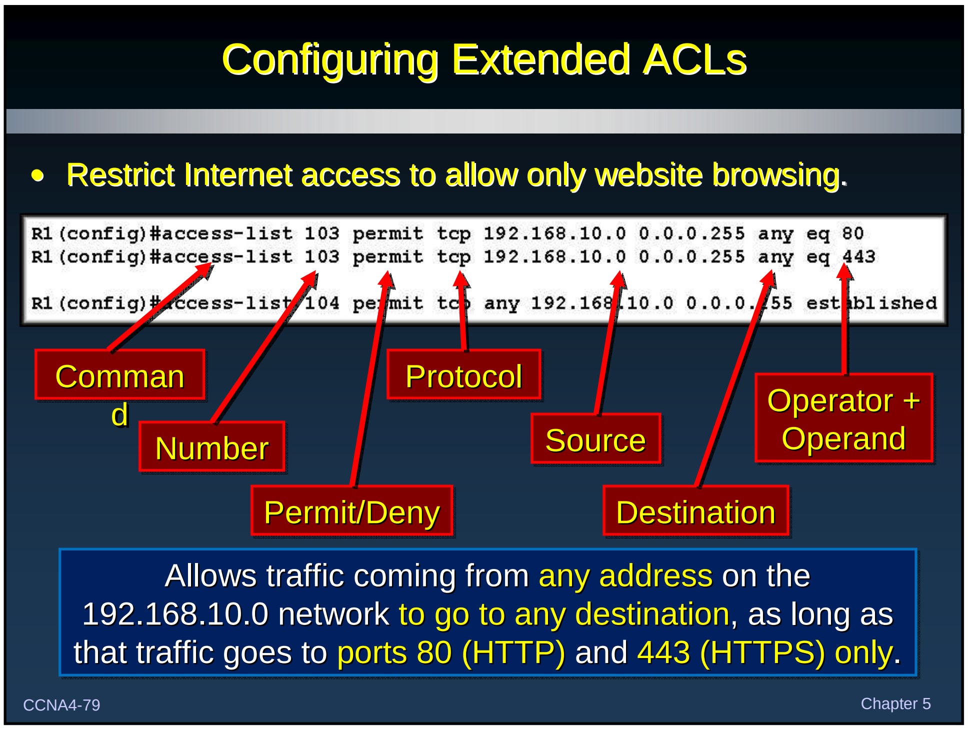
- This parameter allows responses to traffic that originates from the source network to return inbound.
- With the established parameter, the router will allow only the established traffic to come back in and block all other traffic.

Configuring Extended ACLs

- Restrict Internet access to allow only website browsing.
  - ACL 103 applies to traffic leaving the network.
  - ACL 104 to traffic coming into the network.



```
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config)#access-list 104 permit tcp any 192.168.10.0 0.0.0.255 established
```



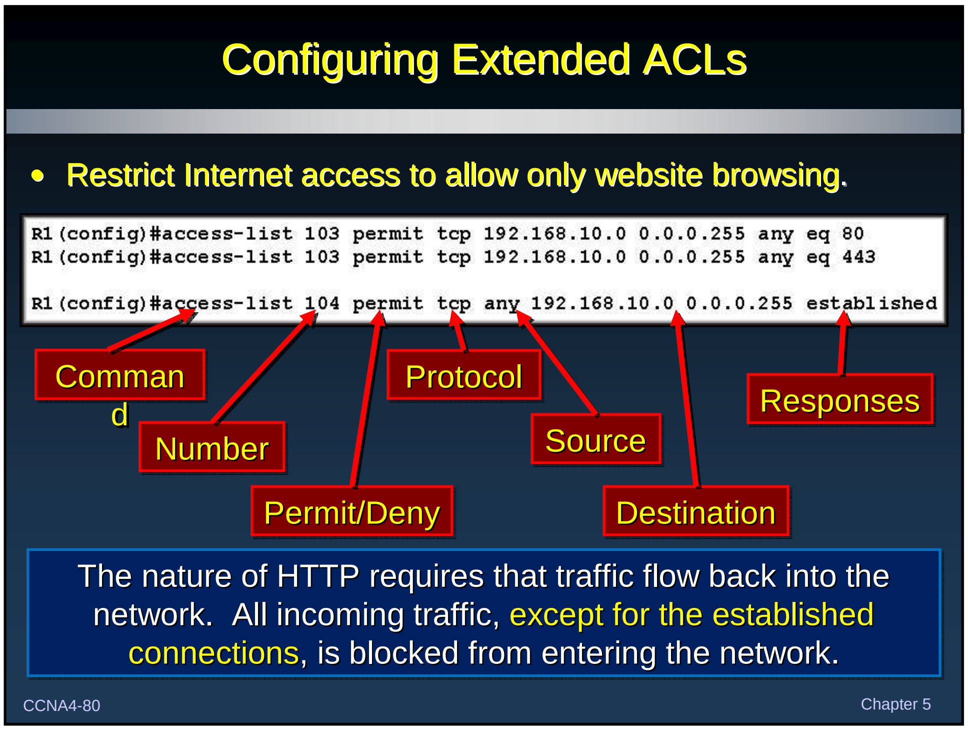
Configuring Extended ACLs

- Restrict Internet access to allow only website browsing.

```
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 443

R1(config)#access-list 104 permit tcp any 192.168.10.0 0.0.0.255 established
```

Comman
d
Number
Permit/Deny
Protocol
Source
Destination
Operator +
Operand

Allows traffic coming from any address on the 192.168.10.0 network to go to any destination, as long as that traffic goes to ports 80 (HTTP) and 443 (HTTPS) only.



Configuring Extended ACLs

- Restrict Internet access to allow only website browsing.

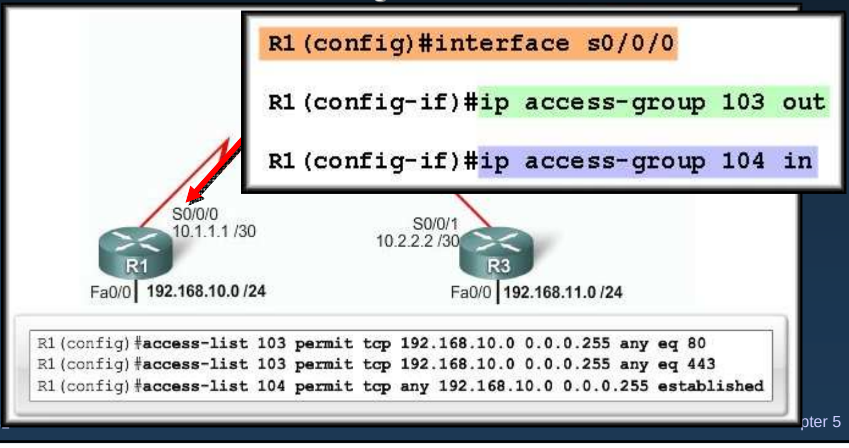
```
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 443

R1(config)#access-list 104 permit tcp any 192.168.10.0 0.0.0.255 established
```

Comman
d
Number
Permit/Deny
Protocol
Source
Destination
Responses

The nature of HTTP requires that traffic flow back into the network. All incoming traffic, except for the established connections, is blocked from entering the network.

Applying Extended ACLs to Interfaces

- Restrict Internet access to allow only website browsing.
  - ACL 103 applies to traffic leaving the network.
  - ACL 104 to traffic coming into the network.

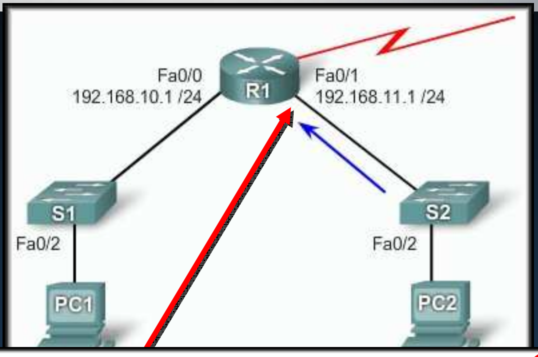


```
R1(config)#interface s0/0/0
R1(config-if)#ip access-group 103 out
R1(config-if)#ip access-group 104 in
```

```
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config)#access-list 104 permit tcp any 192.168.10.0 0.0.0.255 established
```

Applying Extended ACLs to Interfaces

- Deny FTP:
  - Deny all ftp from 192.168.11.0.

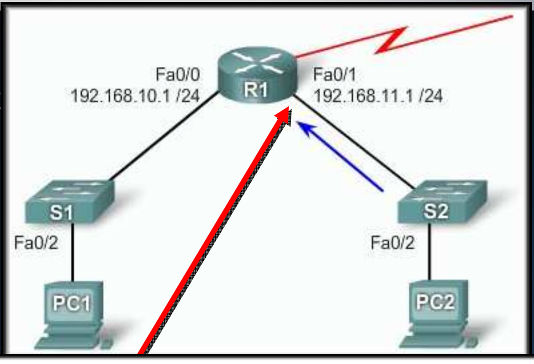


```
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 any eq 21
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 any eq 20
R1(config)#access-list 101 permit ip any any
R1(config)# interface Fa0/1
R1(config-if)#ip access-group 101 in
```

eq ftp
eq ftp-data

Applying Extended ACLs to Interfaces

- Deny Telnet:
  - Deny all telnet from 192.168.11.0.



```
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 any eq 23
R1(config)access-list 101 permit ip any any
R1(config)# interface Fa0/1
R1(config-if)#ip access-group 101 in
```

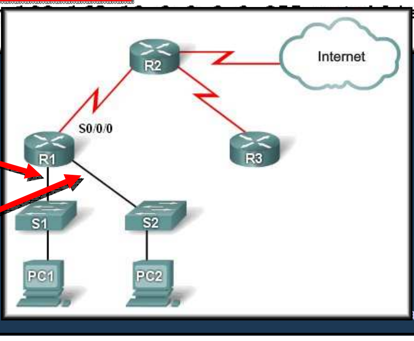
eq telnet

Creating Named Extended ACLs

- Essentially the same way that standard names ACLs are created:

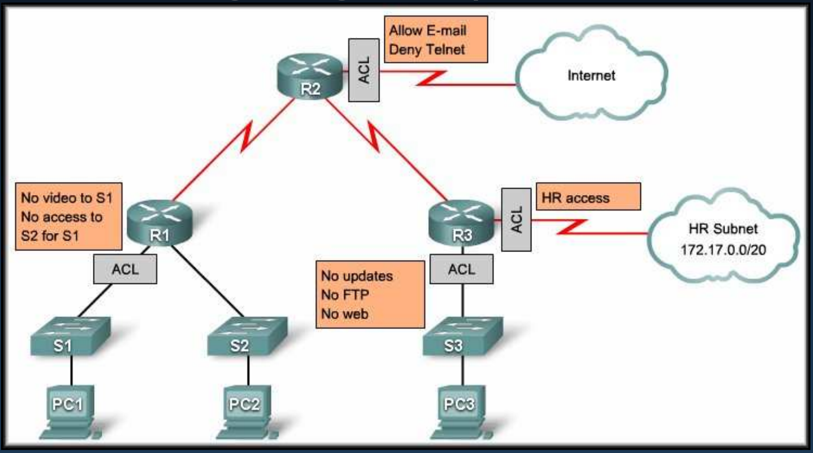
```
R1(config)#ip access-list extended SURFING
R1(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 eq 80
R1(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 eq 443
R1(config-ext-nacl)#exit

R1(config)#ip access-list extended BROWSING
R1(config-ext-nacl)#permit tcp any [illegible]shed
R1(config-ext-nacl)#exit
```

Don't forget to apply the ACL
to all interfaces that
require the filter.



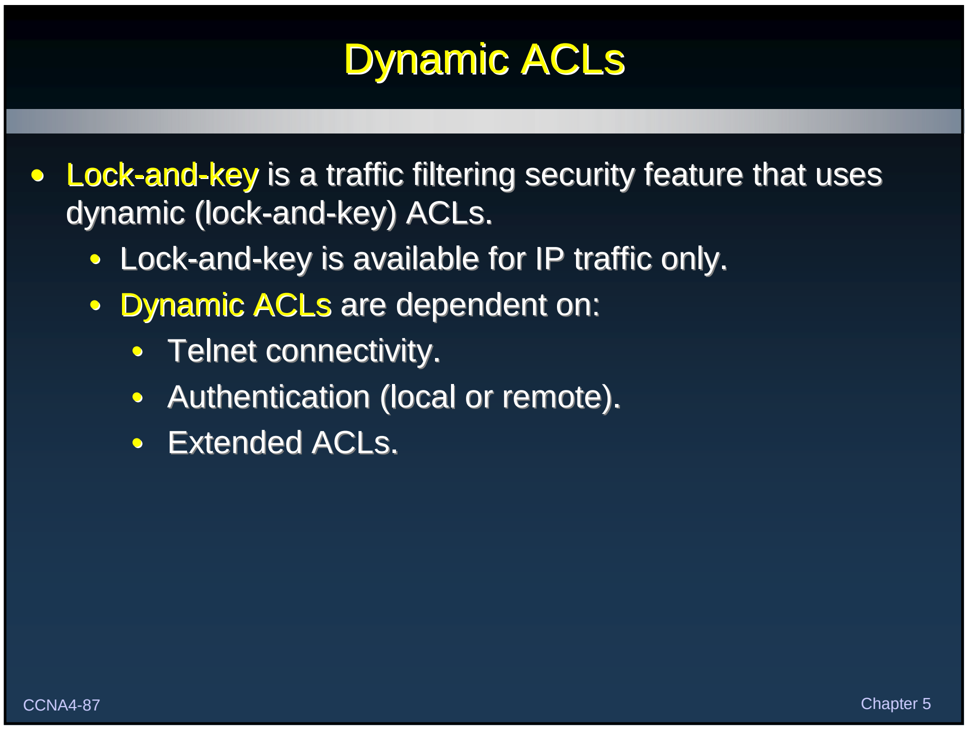
Access Control Lists

Configuring Complex ACLS



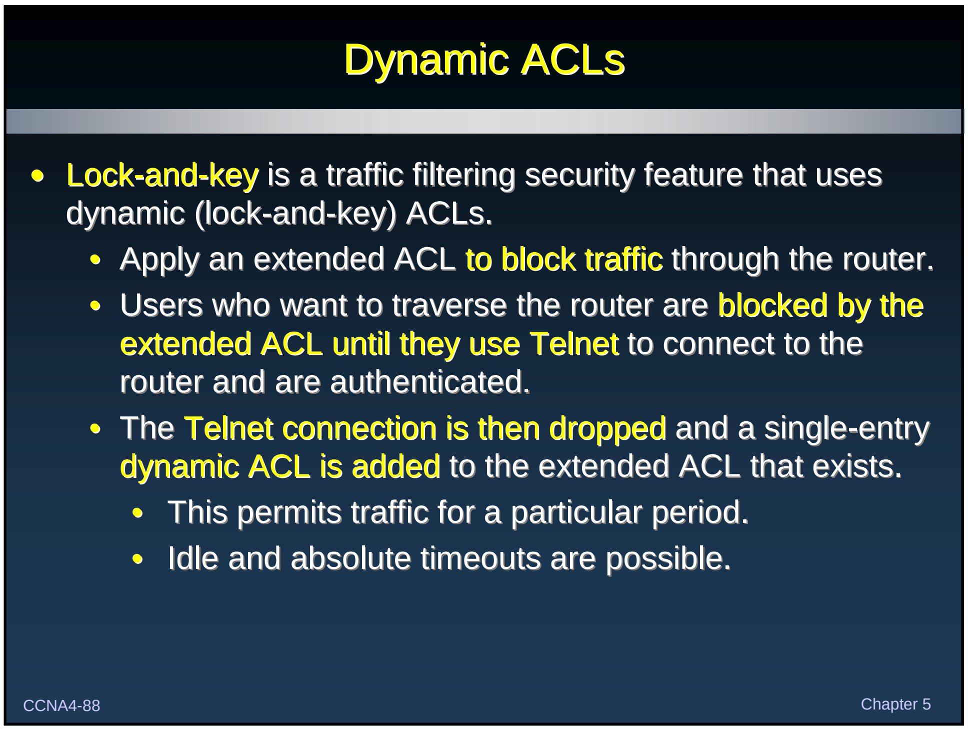
What Are Complex ACLs?

- Three Types:
  - Dynamic (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:
    - Allows outbound traffic and limits inbound traffic in response to sessions that originate inside the router.
  - Time-based:
    - Allows for access control based on the time of day and week.



Dynamic ACLs

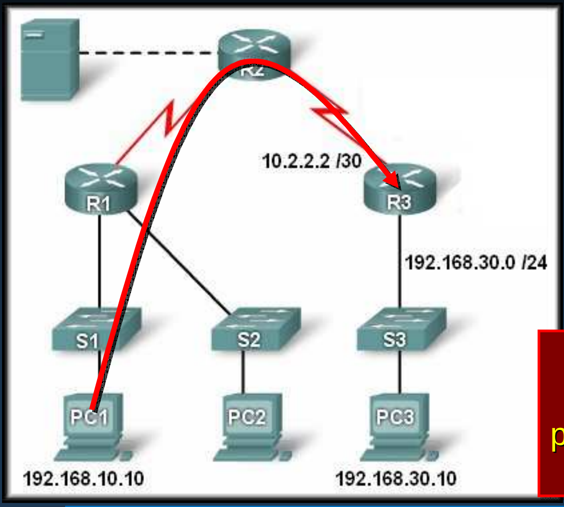
- Lock-and-key is a traffic filtering security feature that uses dynamic (lock-and-key) ACLs.
  - Lock-and-key is available for IP traffic only.
  - Dynamic ACLs are dependent on:
    - Telnet connectivity.
    - Authentication (local or remote).
    - Extended ACLs.



Dynamic ACLs

- Lock-and-key is a traffic filtering security feature that uses dynamic (lock-and-key) ACLs.
  - Apply an extended ACL to block traffic through the router.
  - Users who want to traverse the router are blocked by the extended ACL until they use Telnet to connect to the router and are authenticated.
  - The Telnet connection is then dropped and a single-entry dynamic ACL is added to the extended ACL that exists.
    - This permits traffic for a particular period.
    - Idle and absolute timeouts are possible.

Dynamic ACLs



Set up username and password.

Create dynamic ACL with a 15 minute timeout.

Apply to interface.

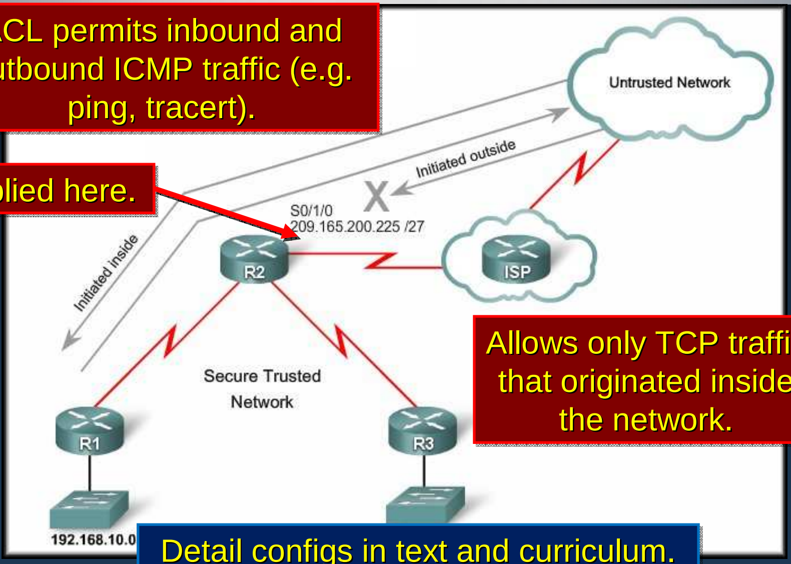
When user connects, validated with ID and password. 5 minute idle timeout disconnects.

Detail configs in text and curriculum.

Reflexive ACLs

- Allow IP traffic for sessions originating inside the network while denying IP traffic for sessions originating outside the network.
  - The router examines the outbound traffic and when it sees a new connection, it adds an entry to a temporary ACL to allow replies back in.
  - Reflexive ACLs contain only temporary entries.
  - These entries are automatically created when a new IP session begins, for example, with an outbound packet, and the entries are automatically removed when the session ends.

Reflexive ACLs



ACL permits inbound and outbound ICMP traffic (e.g. ping, tracert).

Applied here.

Allows only TCP traffic that originated inside the network.

Detail configs in text and curriculum.

Time-based ACLs

- Time-based ACLs are similar to extended ACLs in function, but they allow for access control based on time.
  - To implement time-based ACLs:
    - Create a time range that defines specific times of the day and week.
    - You identify the time range with a name and then refer to it by a function.
    - The time restrictions are imposed on the function itself.

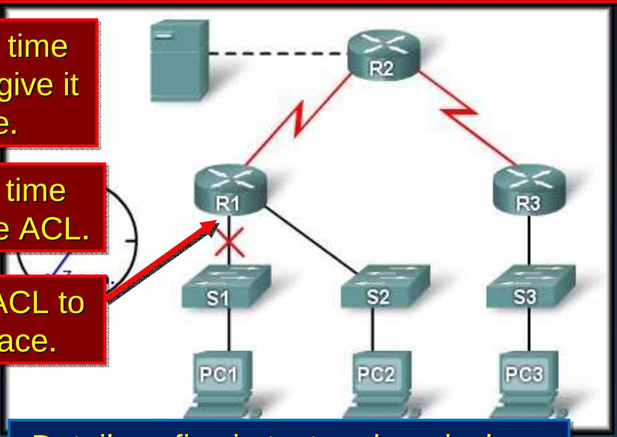
Time-based ACLs

Telnet connection is permitted from the inside network to the outside network on Monday, Wednesday, and Friday during business hours.

Define the time range and give it a name.

Apply the time range to the ACL.

Apply the ACL to the interface.



Detail configs in text and curriculum.

Troubleshooting Common ACL Errors

- Remember that ACL statements are processed in sequence from the top down. Make sure that the sequence of the ACL statements is correct.
- Make sure that you permit/deny the proper protocol. Make the correct use of the TCP, UDP and IP keywords.
- Always double check the use of the any keyword.
- Make sure that you have applied the ACL to the correct interface and for the correct direction.
    - There are specific examples of the above in the text and the curriculum.