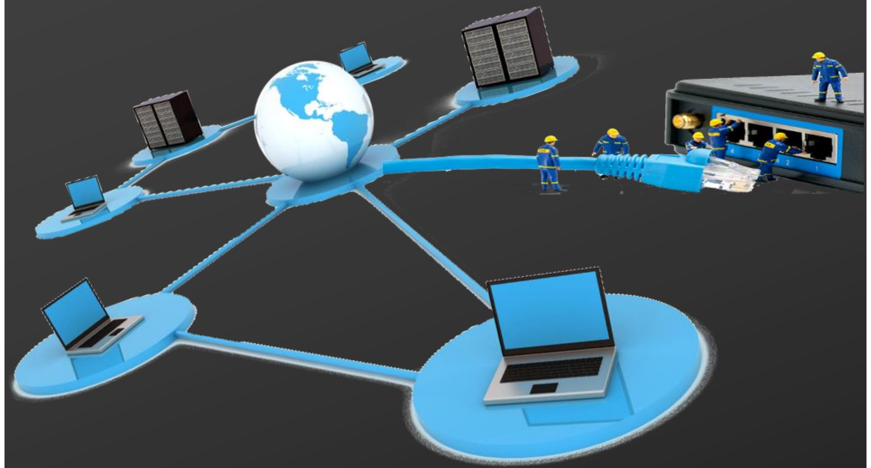
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN – ĐHQG-HCM
Khoa Mạng máy tính & Truyền thông

Access Control List

NT132 – Quản trị mạng và hệ thống

GV: Đỗ Hoàng Hiển

hiendh@uit.edu.vn



Nội dung

ACL

Hôm nay học gì?

1. Khái niệm ACL
2. Cấu hình ACL

Purpose of ACLs

Purpose of ACLs

What is an ACL?

An ACL is a series of IOS commands that are used to filter packets based on information found in the packet header. By default, a router does not have any ACLs configured. When an ACL is applied to an interface, the router performs the additional task of evaluating all network packets as they pass through the interface to determine if the packet can be forwarded.

- An ACL uses a sequential list of permit or deny statements, known as access control entries (ACEs).

Note: ACEs are also commonly called ACL statements.

- When network traffic passes through an interface configured with an ACL, the router compares the information within the packet against each ACE, in sequential order, to determine if the packet matches one of the ACEs. This process is called packet filtering.

Purpose of ACLs

What is an ACL? (Cont.)

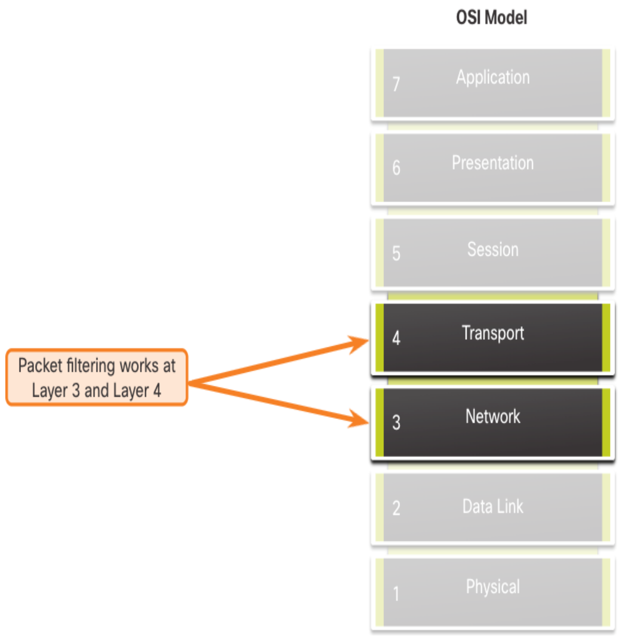
Several tasks performed by routers require the use of ACLs to identify traffic:

- Limit network traffic to increase network performance
- Provide traffic flow control
- Provide a basic level of security for network access
- Filter traffic based on traffic type
- Screen hosts to permit or deny access to network services
- Provide priority to certain classes of network traffic

Purpose of ACLs

Packet Filtering

- Packet filtering controls access to a network by analyzing the incoming and/or outgoing packets and forwarding them or discarding them based on given criteria.
- Packet filtering can occur at Layer 3 or Layer 4.
- Cisco routers support two types of ACLs:
  - **Standard ACLs** - ACLs only filter at Layer 3 using the source IPv4 address only.
  - **Extended ACLs** - ACLs filter at Layer 3 using the source and / or destination IPv4 address. They can also filter at Layer 4 using TCP, UDP ports, and optional protocol type information for finer control.



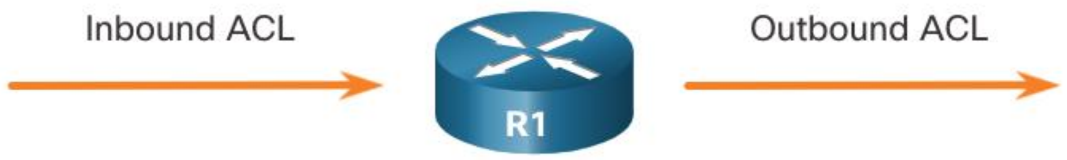
Purpose of ACLs

ACL Operation

- ACLs define the set of rules that give added control for packets that enter inbound interfaces, packets that relay through the router, and packets that exit outbound interfaces of the router.
- ACLs can be configured to apply to inbound traffic and outbound traffic.

Note: ACLs do not act on packets that originate from the router itself.

- An inbound ACL filters packets before they are routed to the outbound interface. An inbound ACL is efficient because it saves the overhead of routing lookups if the packet is discarded.
- An outbound ACL filters packets after being routed, regardless of the inbound interface.



Purpose of ACLs

ACL Operation (Cont.)

When an ACL is applied to an interface, it follows a specific operating procedure. Here are the operational steps used when traffic has entered a router interface with an inbound standard IPv4 ACL configured:

1. The router extracts the source IPv4 address from the packet header.
2. The router starts at the top of the ACL and compares the source IPv4 address to each ACE in a sequential order.
3. When a match is made, the router carries out the instruction, either permitting or denying the packet, and the remaining ACEs in the ACL, if any, are not analyzed.
4. If the source IPv4 address does not match any ACEs in the ACL, the packet is discarded because there is an implicit deny ACE automatically applied to all ACLs.

The last ACE statement of an ACL is always an implicit deny that blocks all traffic. It is hidden and not displayed in the configuration.

Note: An ACL must have at least one permit statement otherwise all traffic will be denied due to the implicit deny ACE statement.

Wildcard Masks in ACLs

Wildcard Masks in ACLs

Wildcard Mask Overview

A wildcard mask is similar to a subnet mask in that it uses the ANDing process to identify which bits in an IPv4 address to match. Unlike a subnet mask, in which binary 1 is equal to a match and binary 0 is not a match, in a wildcard mask, the reverse is true.

- An IPv4 ACE uses a 32-bit wildcard mask to determine which bits of the address to examine for a match.
- Wildcard masks use the following rules to match binary 1s and 0s:
  - **Wildcard mask bit 0** - Match the corresponding bit value in the address
  - **Wildcard mask bit 1** - Ignore the corresponding bit value in the address

Wildcard Mask Types

- **Wildcard to Match a Host:**
  - Assume ACL 10 needs an ACE that only permits the host with IPv4 address 192.168.1.1. Recall that “0” equals a match and “1” equals ignore. To match a specific host IPv4 address, a wildcard mask consisting of all zeroes (i.e., 0.0.0.0) is required.
  - When the ACE is processed, the wildcard mask will permit only the 192.168.1.1 address. The resulting ACE in ACL 10 would be **access-list 10 permit 192.168.1.1 0.0.0.0.**

	Decimal	Binary
IPv4 address	192.168.1.1	**11000000.10101000.00000001.00000001**
Wildcard Mask	0.0.0.0	**00000000.00000000.00000000.00000000**
Permitted IPv4 Address	**192.168.1.1**	**11000000.10101000.00000001.00000001**

Wildcard Mask Types (Cont.)

Wildcard Mask to Match an IPv4 Subnet

- ACL 10 needs an ACE that permits all hosts in the 192.168.1.0/24 network. The wildcard mask 0.0.0.255 stipulates that the very first three octets must match exactly but the fourth octet does not.
- When processed, the wildcard mask 0.0.0.255 permits all hosts in the 192.168.1.0/24 network. The resulting ACE in ACL 10 would be **access-list 10 permit 192.168.1.0 0.0.0.255.**

	Decimal	Binary
IPv4 address	192.168.1.1	**11000000.10101000.00000001.00000001**
Wildcard Mask	0.0.0.255	**00000000.00000000.00000000.11111111**
Permitted IPv4 Address	**192.168.1.0/24**	**11000000.10101000.00000001.00000000**

Wildcard Masks in ACLs

Wildcard Mask Calculation

Calculating wildcard masks can be challenging. One shortcut method is to subtract the subnet mask from 255.255.255.255. Some examples:

- Assume you wanted an ACE in ACL 10 to permit access to all users in the 192.168.3.0/24 network. To calculate the wildcard mask, subtract the subnet mask (255.255.255.0) from 255.255.255.255. This produces the wildcard mask 0.0.0.255. The ACE would be **access-list 10 permit 192.168.3.0 0.0.0.255.**
- Assume you wanted an ACE in ACL 10 to permit network access for the 14 users in the subnet 192.168.3.32/28. Subtract the subnet (i.e., 255.255.255.240) from 255.255.255.255. This produces the wildcard mask 0.0.0.15. The ACE would be **access-list 10 permit 192.168.3.32 0.0.0.15.**
- Assume you needed an ACE in ACL 10 to permit only networks 192.168.10.0 and 192.168.11.0. These two networks could be summarized as 192.168.10.0/23 which is a subnet mask of 255.255.254.0. Subtract 255.255.254.0 subnet mask from 255.255.255.255. This produces the wildcard mask 0.0.1.255. The ACE would be **access-list 10 permit 192.168.10.0 0.0.1.255.**

Wildcard Masks in ACLs

Wildcard Mask Keywords

The Cisco IOS provides two keywords to identify the most common uses of wildcard masking. The two keywords are:

- **host** - This keyword substitutes for the 0.0.0.0 mask. This mask states that all IPv4 address bits must match to filter just one host address.
- **any** - This keyword substitutes for the 255.255.255.255 mask. This mask says to ignore the entire IPv4 address or to accept any addresses.

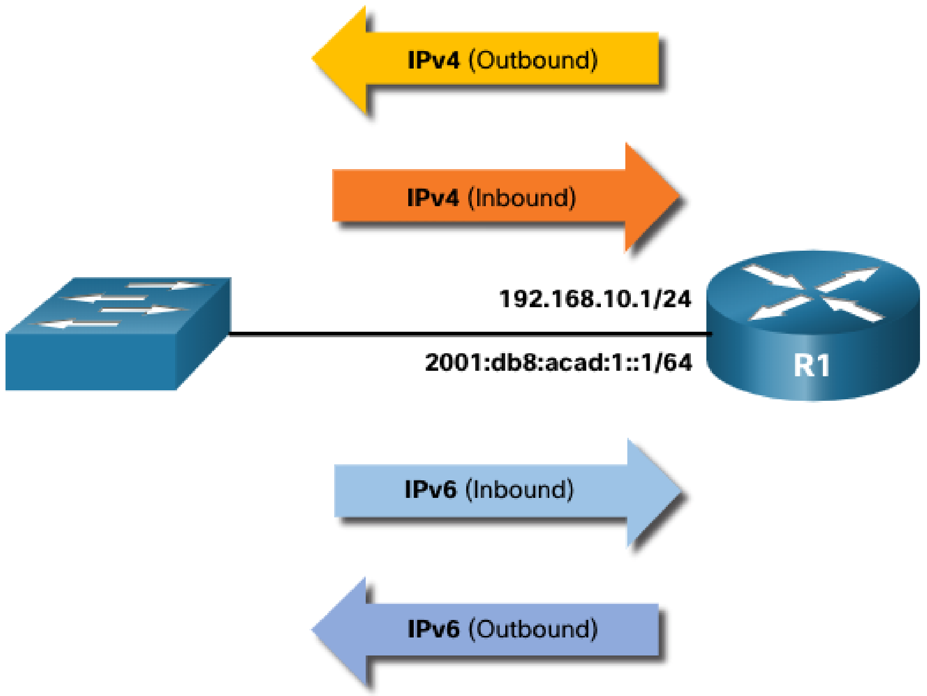
Guidelines for ACL Creation

Guidelines for ACL Creation

Limited Number of ACLs per Interface

- There is a limit on the number of ACLs that can be applied on a router interface. For example, a dual-stacked (i.e, IPv4 and IPv6) router interface can have up to four ACLs applied, as shown in the figure.
- Specifically, a router interface can have:
  - One outbound IPv4 ACL.
  - One inbound IPv4 ACL.
  - One inbound IPv6 ACL.
  - One outbound IPv6 ACL.

Note: ACLs do not have to be configured in both directions. The number of ACLs and their direction applied to the interface will depend on the security policy of the organization.



Types of IPv4 ACLs

Types of IPv4 ACLs

Standard and Extended ACLs

There are two types of IPv4 ACLs:

- **Standard ACLs** - These permit or deny packets based only on the source IPv4 address.
- **Extended ACLs** - These permit or deny packets based on the source IPv4 address and destination IPv4 address, protocol type, source and destination TCP or UDP ports and more.

Types of IPv4 ACLs

Numbered and Named ACLs

Numbered ACLs

- ACLs numbered 1-99, or 1300-1999 are standard ACLs, while ACLs numbered 100-199, or 2000-2699 are extended ACLs.

```
R1(config)# access-list ?
 <1-99> IP standard access list
 <100-199> IP extended access list
 <1100-1199> Extended 48-bit MAC address access list
 <1300-1999> IP standard access list (expanded range)
 <200-299> Protocol type-code access list
 <2000-2699> IP extended access list (expanded range)
 <700-799> 48-bit MAC address access list
 rate-limit Simple rate-limit specific access list
 template Enable IP template acls
Router(config)# access-list
```

Types of IPv4 ACLs

Numbered and Named ACLs (Cont.)

Named ACLs

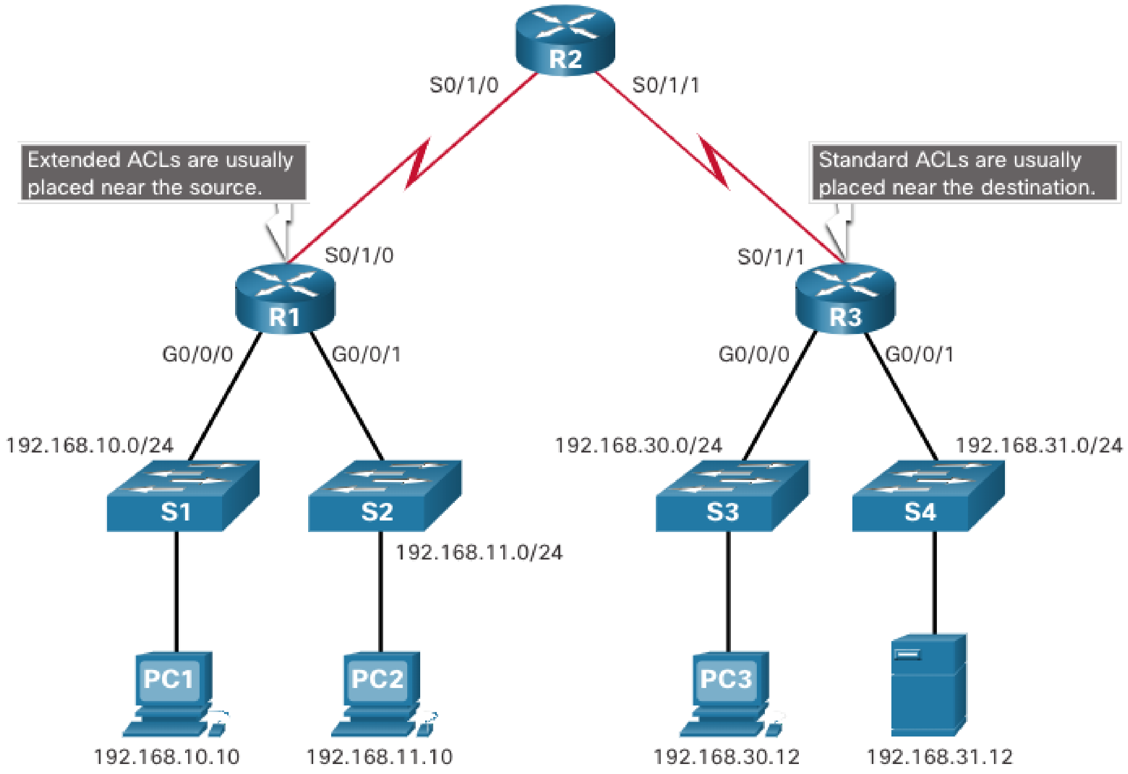
- Named ACLs are the preferred method to use when configuring ACLs. Specifically, standard and extended ACLs can be named to provide information about the purpose of the ACL. For example, naming an extended ACL FTP-FILTER is far better than having a numbered ACL 100.
- The **ip access-list** global configuration command is used to create a named ACL, as shown in the following example.

```
R1(config)# ip access-list extended FTP-FILTER
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq ftp
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq ftp-data
R1(config-ext-nacl)#
```

Types of IPv4 ACLs

Where to Place ACLs

- Every ACL should be placed where it has the greatest impact on efficiency.
- Extended ACLs should be located as close as possible to the source of the traffic to be filtered.
- Standard ACLs should be located as close to the destination as possible.

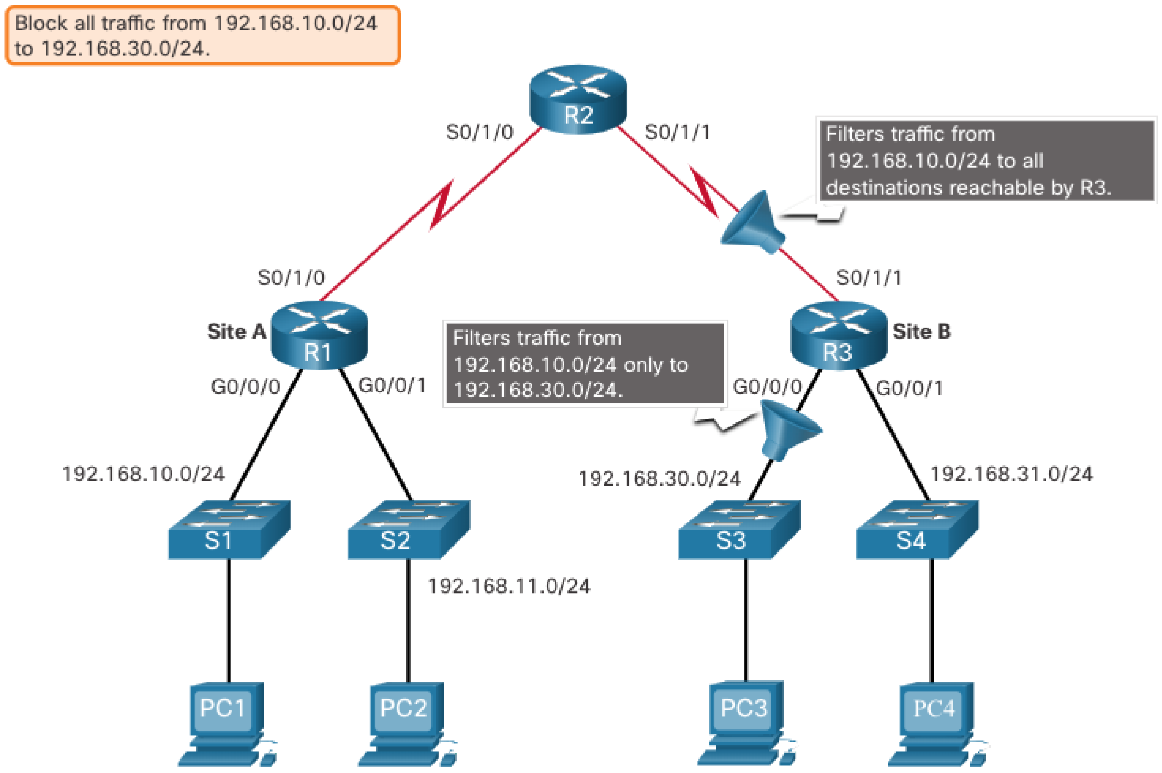


Types of IPv4 ACLs

Standard ACL Placement Example

In the figure, the administrator wants to prevent traffic originating in the 192.168.10.0/24 network from reaching the 192.168.30.0/24 network.

Following the basic placement guidelines, the administrator would place a standard ACL on router R3.



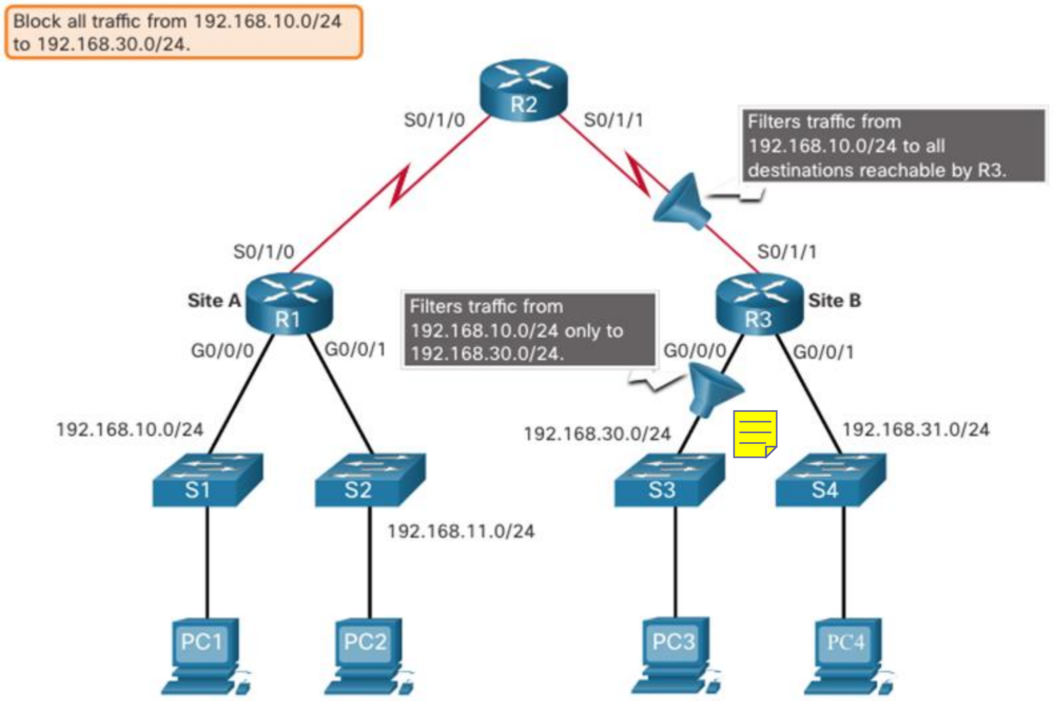
Block all traffic from 192.168.10.0/24 to 192.168.30.0/24.
R2
S0/1/0
S0/1/1
Filters traffic from 192.168.10.0/24 to all destinations reachable by R3.
S0/1/0
S0/1/1
Site A
R1
Site B
R3
Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.
G0/0/0
G0/0/1
G0/0/0
G0/0/1
192.168.10.0/24
192.168.30.0/24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

Types of IPv4 ACLs

Standard ACL Placement Example (Cont.)

There are two possible interfaces on R3 to apply the standard ACL:

- **R3 S0/1/1 interface (inbound)** - The standard ACL can be applied inbound on the R3 S0/1/1 interface to deny traffic from .10 network. However, it would also filter .10 traffic to the 192.168.31.0/24 (.31 in this example) network. Therefore, the standard ACL should not be applied to this interface.
- **R3 G0/0 interface (outbound)** - The standard ACL can be applied outbound on the R3 G0/0/0 interface. This will not affect other networks that are reachable by R3. Packets from .10 network will still be able to reach the .31 network. This is the best interface to place the standard ACL to meet the traffic requirements.

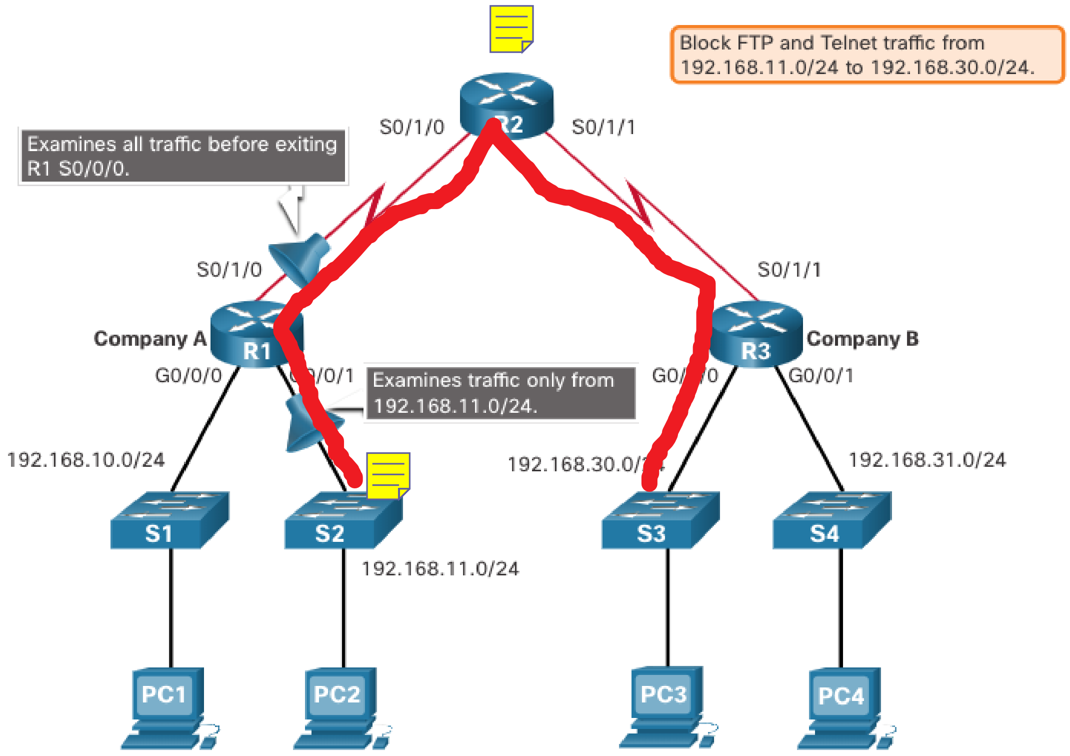


Block all traffic from 192.168.10.0/24 to 192.168.30.0/24.
R2
S0/1/0
S0/1/1
Filters traffic from 192.168.10.0/24 to all destinations reachable by R3.
S0/1/0
S0/1/1
Site A
R1
Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.
R3
Site B
G0/0/0
G0/0/1
G0/0/0
G0/0/1
192.168.10.0/24
192.168.30.0/24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

Types of IPv4 ACLs

Extended ACL Placement Example

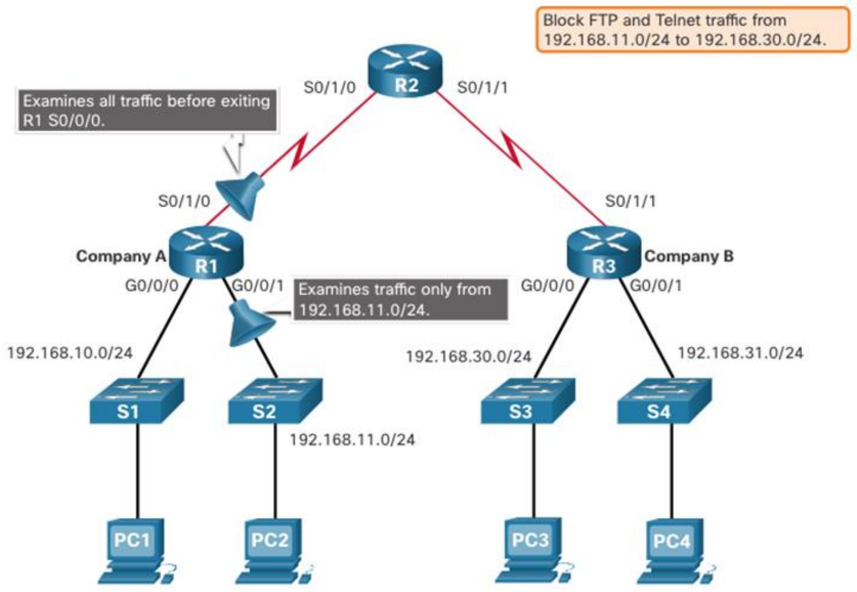
- Extended ACLs should be located as close to the source as possible.
- However, the organization can only place ACLs on devices that they control. Therefore, the extended ACL placement must be determined in the context of where organizational control extends.
- In the figure, for example, Company A wants to deny Telnet and FTP traffic to Company B's 192.168.30.0/24 network from their 192.168.11.0/24 network, while permitting all other traffic.


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

Types of IPv4 ACLs

Extended ACL Placement Example (Cont.)

- An extended ACL on R3 would accomplish the task, but the administrator does not control R3. In addition, this solution allows unwanted traffic to cross the entire network, only to be blocked at the destination.
- The solution is to place an extended ACL on R1 that specifies both source and destination addresses.
- There are two possible interfaces on R1 to apply the extended ACL:
  - **R1 S0/1/0 interface (outbound)** - The extended ACL can be applied outbound on the S0/1/0 interface. This solution will process all packets leaving R1 including packets from 192.168.10.0/24.
  - **R1 G0/0/1 interface (inbound)** - The extended ACL can be applied inbound on the G0/0/1 and only packets from the 192.168.11.0/24 network are subject to ACL processing on R1. Because the filter is to be limited to only those packets leaving the 192.168.11.0/24 network, applying the extended ACL to G0/1 is the best solution.


Block FTP and Telnet traffic from 192.168.11.0/24 to 192.168.30.0/24.
R2
S0/1/0
S0/1/1
Examines all traffic before exiting R1 S0/0/0.
S0/1/0
S0/1/1
Company A
R1
G0/0/0
G0/0/1
Examines traffic only from 192.168.11.0/24.
G0/0/0
R3
Company B
G0/0/1
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

Configure Standard IPv4 ACLs

Configure Standard IPv4 ACLs

Numbered Standard IPv4 ACL Syntax

To create a numbered standard ACL, use the **access-list** command.

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

Parameter	Description
access-list-number	Number range is 1 to 99 or 1300 to 1999
deny	Denies access if the condition is matched
permit	Permits access if the condition is matched
remark *text*	(Optional) text entry for documentation purposes
source	Identifies the source network or host address to filter
source-wildcard	(Optional) 32-bit wildcard mask that is applied to the source
log	(Optional) Generates and sends an informational message when the ACE is matched

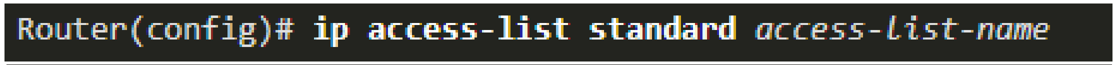
Note: Use the **no access-list** *access-list-number* global configuration command to remove a numbered standard ACL.

Configure Standard IPv4 ACLs

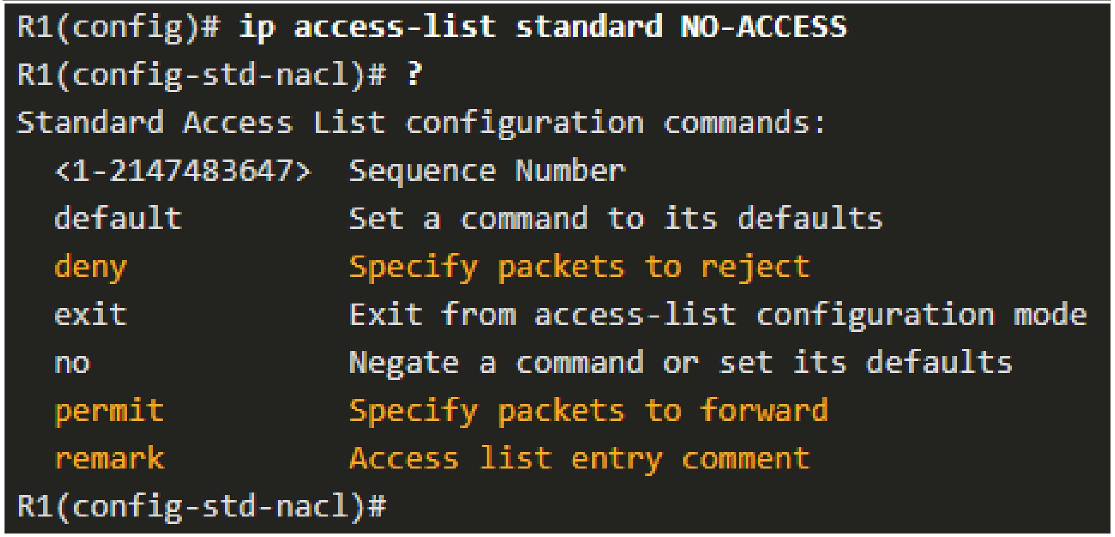
Named Standard IPv4 ACL Syntax

To create a named standard ACL, use the **ip access-list standard** command.

- ACL names are alphanumeric, case sensitive, and must be unique.
- Capitalizing ACL names is not required but makes them stand out when viewing the running-config output.



```
Router(config)# ip access-list standard access-list-name
```



```
R1(config)# ip access-list standard NO-ACCESS
R1(config-std-nacl)# ?
Standard Access List configuration commands:
  <1-2147483647>  Sequence Number
  default         Set a command to its defaults
  deny            Specify packets to reject
  exit            Exit from access-list configuration mode
  no              Negate a command or set its defaults
  permit          Specify packets to forward
  remark          Access list entry comment
R1(config-std-nacl)#
```

Configure Standard IPv4 ACLs

Apply a Standard IPv4 ACL

After a standard IPv4 ACL is configured, it must be linked to an interface or feature.

- The **ip access-group** command is used to bind a numbered or named standard IPv4 ACL to an interface.
- To remove an ACL from an interface, first enter the **no ip access-group** interface configuration command.

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

Configure Standard IPv4 ACLs
Numbered Standard ACL Example

The example ACL permits traffic from host 192.168.10.10 and all hosts on the 192.168.20.0/24 network out interface serial 0/1/0 on router R1.

```
R1(config)# access-list 10 remark ACE permits ONLY host 192.168.10.10 to the internet
R1(config)# access-list 10 permit host 192.168.10.10
R1(config)# do show access-lists
Standard IP access list 10
    10 permit 192.168.10.10
R1(config)#
```

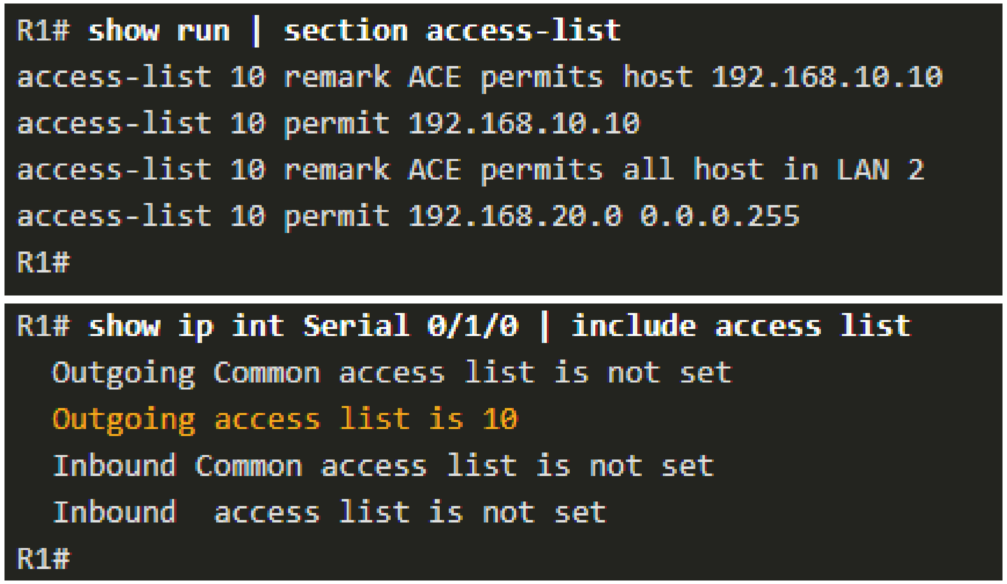
```
R1(config)# access-list 10 remark ACE permits all host in LAN 2
R1(config)# access-list 10 permit 192.168.20.0 0.0.0.255
R1(config)# do show access-lists
Standard IP access list 10
    10 permit 192.168.10.10
    20 permit 192.168.20.0, wildcard bits 0.0.0.255
R1(config)#
```

```
R1(config)# interface Serial 0/1/0
R1(config-if)# ip access-group 10 out
R1(config-if)# end
R1#
```

Configure Standard IPv4 ACLs

Numbered Standard ACL Example (Cont.)

- Use the **show running-config** command to review the ACL in the configuration.
- Use the **show ip interface** command to verify the ACL is applied to the interface.



```
R1# show run | section access-list
access-list 10 remark ACE permits host 192.168.10.10
access-list 10 permit 192.168.10.10
access-list 10 remark ACE permits all host in LAN 2
access-list 10 permit 192.168.20.0 0.0.0.255
R1#
```

```
R1# show ip int Serial 0/1/0 | include access list
  Outgoing Common access list is not set
  Outgoing access list is 10
  Inbound Common access list is not set
  Inbound  access list is not set
R1#
```

Configure Standard IPv4 ACLs

Named Standard ACL Example

The example ACL permits traffic from host 192.168.10.10 and all hosts on the 192.168.20.0/24 network out interface serial 0/1/0 on router R1.

```
R1(config)# no access-list 10
R1(config)# ip access-list standard PERMIT-ACCESS
R1(config-std-nacl)# remark ACE permits host 192.168.10.10
R1(config-std-nacl)# permit host 192.168.10.10
R1(config-std-nacl)#
```

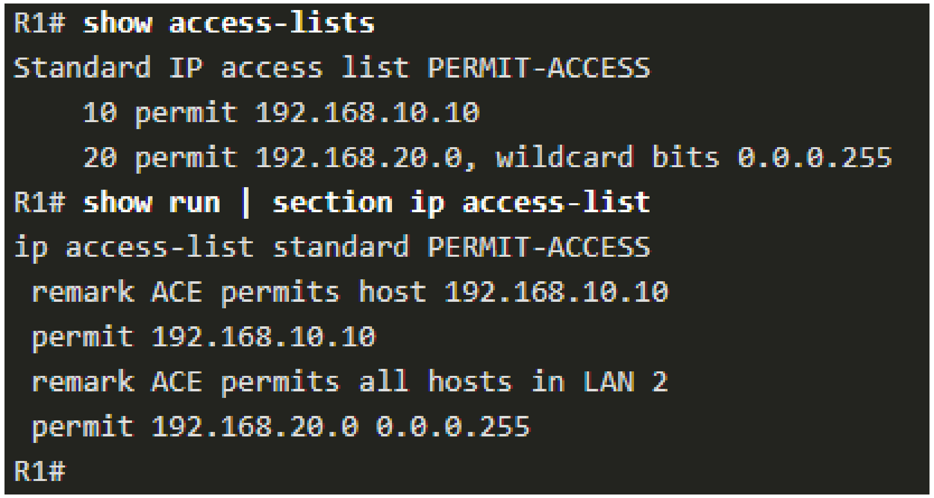
```
R1(config-std-nacl)# remark ACE permits host 192.168.10.10
R1(config-std-nacl)# permit host 192.168.10.10
R1(config-std-nacl)# remark ACE permits all hosts in LAN 2
R1(config-std-nacl)# permit 192.168.20.0 0.0.0.255
R1(config-std-nacl)# exit
R1(config)#
```

```
R1(config)# interface Serial 0/1/0
R1(config-if)# ip access-group PERMIT-ACCESS out
R1(config-if)# end
R1#
```

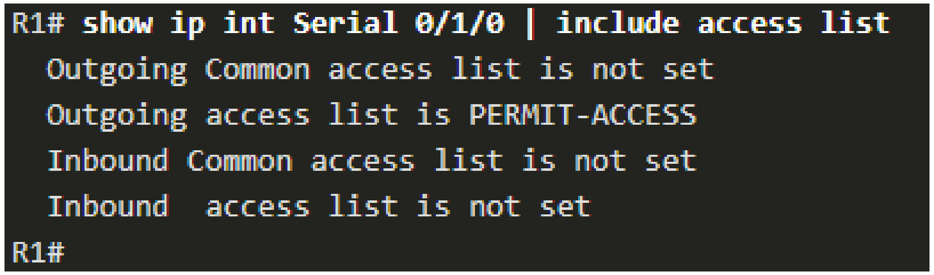
Configure Standard IPv4 ACLs

Named Standard ACL Example (Cont.)

- Use the **show access-list** command to review the ACL in the configuration.
- Use the **show ip interface** command to verify the ACL is applied to the interface.



```
R1# show access-lists
Standard IP access list PERMIT-ACCESS
    10 permit 192.168.10.10
    20 permit 192.168.20.0, wildcard bits 0.0.0.255
R1# show run | section ip access-list
ip access-list standard PERMIT-ACCESS
 remark ACE permits host 192.168.10.10
 permit 192.168.10.10
 remark ACE permits all hosts in LAN 2
 permit 192.168.20.0 0.0.0.255
R1#
```



```
R1# show ip int Serial 0/1/0 | include access list
  Outgoing Common access list is not set
  Outgoing access list is PERMIT-ACCESS
  Inbound Common access list is not set
  Inbound  access list is not set
R1#
```

Modify IPv4 ACLs

Modify IPv4 ACLs

Two Methods to Modify an ACL

After an ACL is configured, it may need to be modified. ACLs with multiple ACEs can be complex to configure. Sometimes the configured ACE does not yield the expected behaviors.

There are two methods to use when modifying an ACL:

- Use a text editor.
- Use sequence numbers.

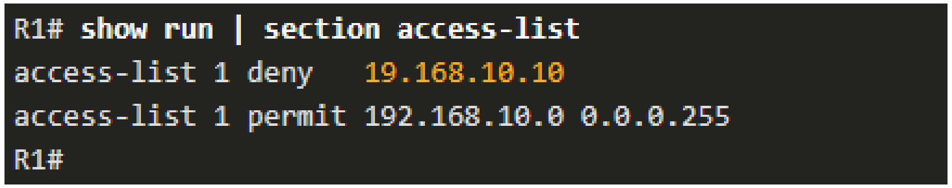
Modify IPv4 ACLs

Text Editor Method

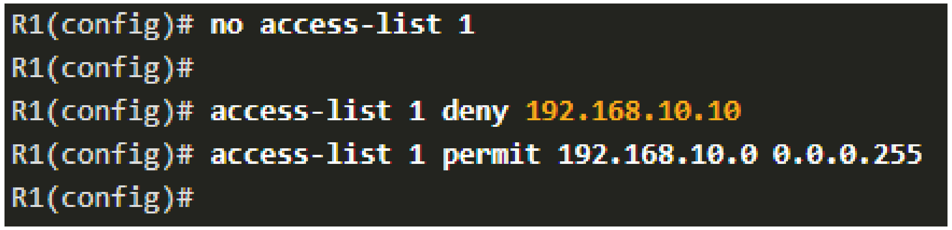
ACLs with multiple ACEs should be created in a text editor. This allows you to plan the required ACEs, create the ACL, and then paste it into the router interface. It also simplifies the tasks to edit and fix an ACL.

To correct an error in an ACL:

- Copy the ACL from the running configuration and paste it into the text editor.
- Make the necessary edits or changes.
- Remove the previously configured ACL on the router.
- Copy and paste the edited ACL back to the router.



```
R1# show run | section access-list
access-list 1 deny   19.168.10.10
access-list 1 permit 192.168.10.0 0.0.0.255
R1#
```



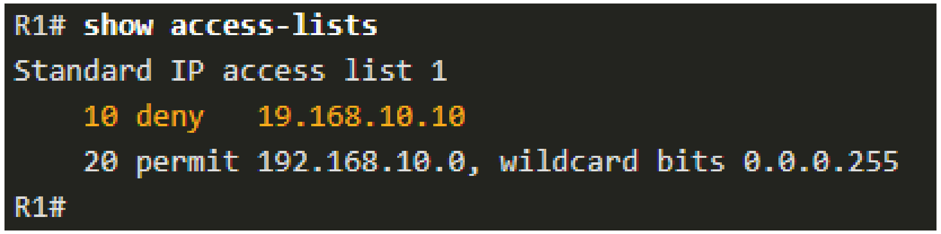
```
R1(config)# no access-list 1
R1(config)#
R1(config)# access-list 1 deny 192.168.10.10
R1(config)# access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)#
```

Modify IPv4 ACLs

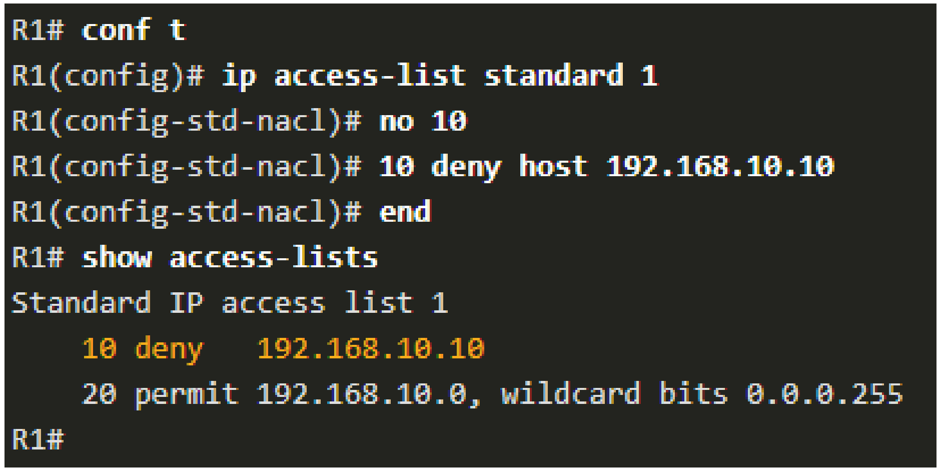
Sequence Number Method

An ACL ACE can be deleted or added using the ACL sequence numbers.

- Use the **ip access-list standard** command to edit an ACL.
- Statements cannot be overwritten using an existing sequence number. The current statement must be deleted first with the **no 10** command. Then the correct ACE can be added using sequence number.



```
R1# show access-lists
Standard IP access list 1
    10 deny   19.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
R1#
```



```
R1# conf t
R1(config)# ip access-list standard 1
R1(config-std-nacl)# no 10
R1(config-std-nacl)# 10 deny host 192.168.10.10
R1(config-std-nacl)# end
R1# show access-lists
Standard IP access list 1
    10 deny   192.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
R1#
```

Modify IPv4 ACLs

Modify a Named ACL Example

Named ACLs can also use sequence numbers to delete and add ACEs. In the example an ACE is added to deny hosts 192.168.10.11.

```
R1# show access-lists
Standard IP access list NO-ACCESS
    10 deny   192.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
```

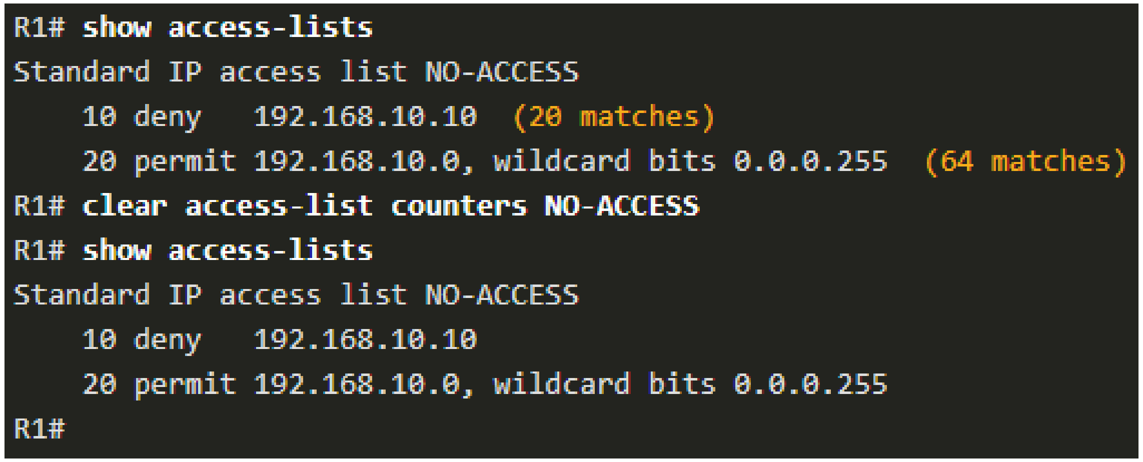
```
R1# configure terminal
R1(config)# ip access-list standard NO-ACCESS
R1(config-std-nacl)# 15 deny 192.168.10.5
R1(config-std-nacl)# end
R1#
R1# show access-lists
Standard IP access list NO-ACCESS
    15 deny   192.168.10.5
    10 deny   192.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
R1#
```

Modify IPv4 ACLs

ACL Statistics

The **show access-lists** command in the example shows statistics for each statement that has been matched.

- The deny ACE has been matched 20 times and the permit ACE has been matched 64 times.
- Note that the implied deny any statement does not display any statistics. To track how many implicit denied packets have been matched, you must manually configure the **deny any** command.
- Use the **clear access-list counters** command to clear the ACL statistics.



```
R1# show access-lists
Standard IP access list NO-ACCESS
    10 deny   192.168.10.10  (20 matches)
    20 permit 192.168.10.0, wildcard bits 0.0.0.255  (64 matches)
R1# clear access-list counters NO-ACCESS
R1# show access-lists
Standard IP access list NO-ACCESS
    10 deny   192.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
R1#
```

Secure VTY Ports with a Standard IPv4 ACL

Secure VTY Ports with a Standard IPv4 ACL

The access-class Command

A standard ACL can secure remote administrative access to a device using the vty lines by implementing the following two steps:

- Create an ACL to identify which administrative hosts should be allowed remote access.
- Apply the ACL to incoming traffic on the vty lines.

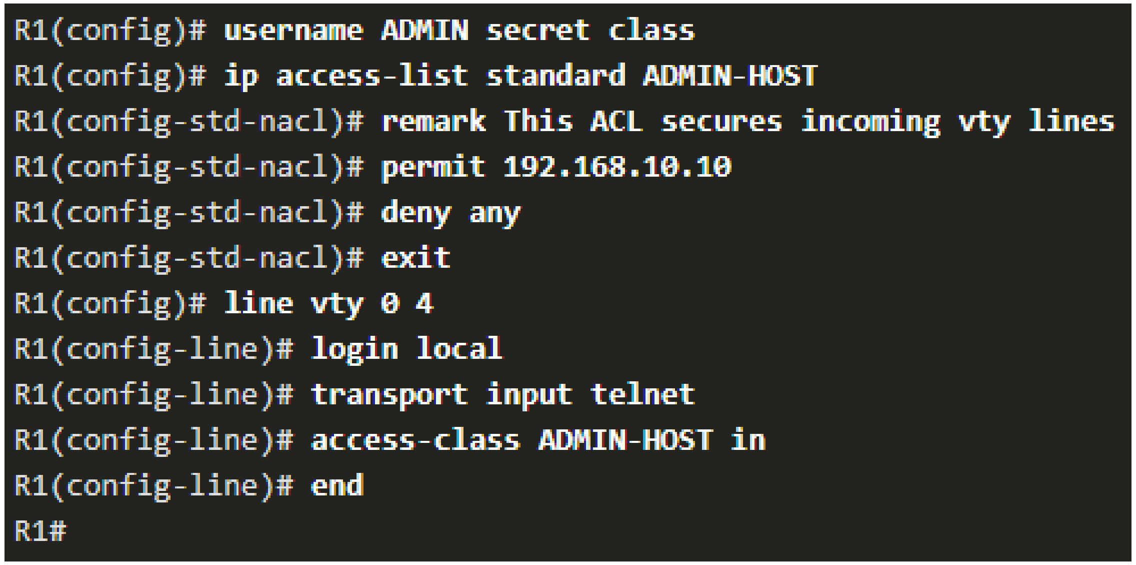
```
R1(config-line)# access-class {access-list-number | access-list-name} { in | out }
```

Secure VTY Ports with a Standard IPv4 ACL

Secure VTY Access Example

This example demonstrates how to configure an ACL to filter vty traffic.

- First, a local database entry for a user **ADMIN** and password **class** is configured.
- The vty lines on R1 are configured to use the local database for authentication, permit SSH traffic, and use the ADMIN-HOST ACL to restrict traffic.



```
R1(config)# username ADMIN secret class
R1(config)# ip access-list standard ADMIN-HOST
R1(config-std-nacl)# remark This ACL secures incoming vty lines
R1(config-std-nacl)# permit 192.168.10.10
R1(config-std-nacl)# deny any
R1(config-std-nacl)# exit
R1(config)# line vty 0 4
R1(config-line)# login local
R1(config-line)# transport input telnet
R1(config-line)# access-class ADMIN-HOST in
R1(config-line)# end
R1#
```

Secure VTY Ports with a Standard IPv4 ACL

Verify the VTY Port is Secured

After an ACL to restrict access to the vty lines is configured, it is important to verify it works as expected.

To verify the ACL statistics, issue the **show access-lists** command.

- The match in the permit line of the output is a result of a successful SSH connection by host with IP address 192.168.10.10.
- The match in the deny statement is due to the failed attempt to create a SSH connection from a device on another network.

```
R1#
Oct  9 15:11:19.544: %SEC_LOGIN-5-LOGIN_SUCCESS: Login Success [user: admin] [Source: 192.168.10.10]
[localport: 23] at 15:11:19 UTC Wed Oct 9 2019
R1# show access-lists
Standard IP access list ADMIN-HOST
    10 permit 192.168.10.10  (2 matches)
    20 deny   any  (2 matches)
R1#
```

Configure Extended IPv4 ACLs

Configure Extended IPv4 ACLs

Extended ACLs

Extended ACLs provide a greater degree of control. They can filter on source address, destination address, protocol (i.e., IP, TCP, UDP, ICMP), and port number.

Extended ACLs can be created as:

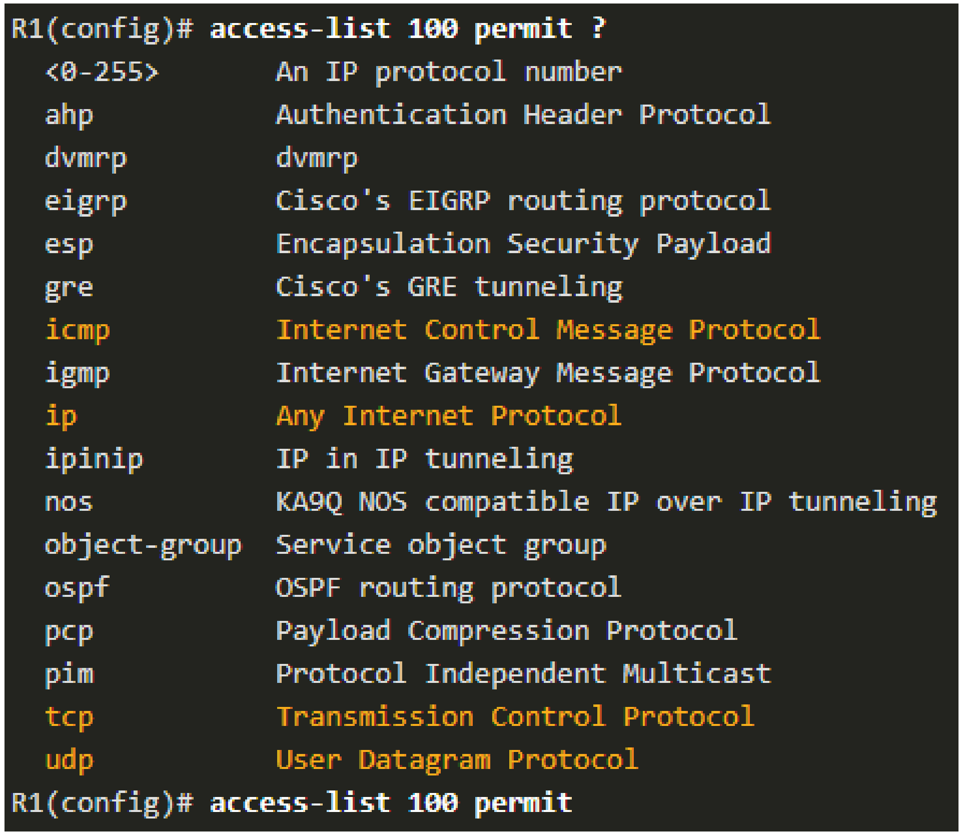
- **Numbered Extended ACL** - Created using the **access-list** *access-list-number* global configuration command.
- **Named Extended ACL** - Created using the **ip access-list extended** *access-list-name*.

Configure Extended IPv4 ACLs

Protocols and Ports

Extended ACLs can filter on internet protocols and ports. Use the **?** to get help when entering a complex ACE. The four highlighted protocols are the most popular options.

Protocol Options

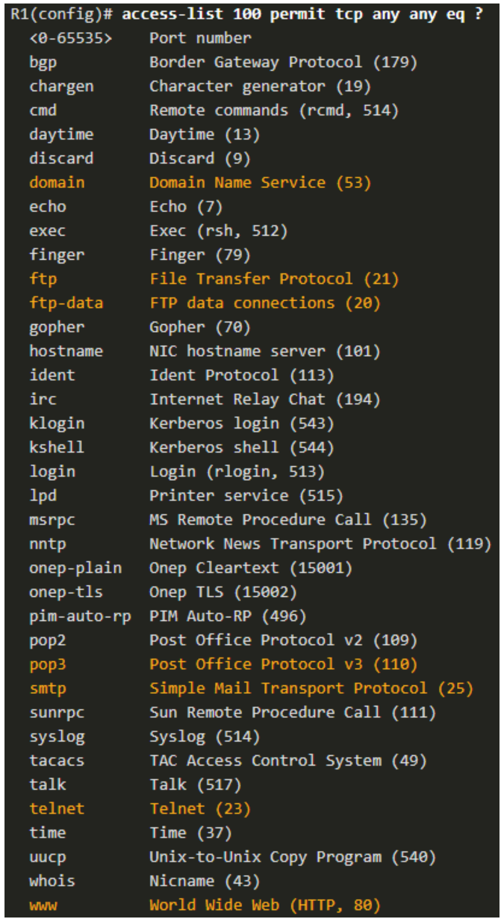


```
R1(config)# access-list 100 permit ?
  <0-255>        An IP protocol number
  ahp            Authentication Header Protocol
  dvmrp          dvmrp
  eigrp          Cisco's EIGRP routing protocol
  esp            Encapsulation Security Payload
  gre            Cisco's GRE tunneling
  icmp           Internet Control Message Protocol
  igmp           Internet Gateway Message Protocol
  ip             Any Internet Protocol
  ipinip         IP in IP tunneling
  nos            KA9Q NOS compatible IP over IP tunneling
  object-group   Service object group
  ospf           OSPF routing protocol
  pcp            Payload Compression Protocol
  pim            Protocol Independent Multicast
  tcp            Transmission Control Protocol
  udp            User Datagram Protocol
R1(config)# access-list 100 permit
```

Configure Extended IPv4 ACLs

Protocols and Ports (Cont.)

Selecting a protocol influences port options. Many TCP port options are available, as shown in the output.



```
R1(config)# access-list 100 permit tcp any any eq ?
  <0-65535>    Port number
  bgp          Border Gateway Protocol (179)
  chargen      Character generator (19)
  cmd          Remote commands (rcmd, 514)
  daytime      Daytime (13)
  discard      Discard (9)
  domain       Domain Name Service (53)
  echo         Echo (7)
  exec         Exec (rsh, 512)
  finger       Finger (79)
  ftp          File Transfer Protocol (21)
  ftp-data     FTP data connections (20)
  gopher       Gopher (70)
  hostname     NIC hostname server (101)
  ident        Ident Protocol (113)
  irc          Internet Relay Chat (194)
  klogin       Kerberos login (543)
  kshell       Kerberos shell (544)
  login        Login (rlogin, 513)
  lpd          Printer service (515)
  msrpc        MS Remote Procedure Call (135)
  nntp         Network News Transport Protocol (119)
  onep-plain   Onep Cleartext (15001)
  onep-tls     Onep TLS (15002)
  pim-auto-rp  PIM Auto-RP (496)
  pop2         Post Office Protocol v2 (109)
  pop3         Post Office Protocol v3 (110)
  smtp         Simple Mail Transport Protocol (25)
  sunrpc       Sun Remote Procedure Call (111)
  syslog       Syslog (514)
  tacacs       TAC Access Control System (49)
  talk         Talk (517)
  telnet       Telnet (23)
  time         Time (37)
  uucp         Unix-to-Unix Copy Program (540)
  whois        Nicname (43)
  www          World Wide Web (HTTP, 80)
```

Configure Extended IPv4 ACLs

Protocols and Port Numbers Configuration Examples

Extended ACLs can filter on different port number and port name options.

This example configures an extended ACL 100 to filter HTTP traffic. The first ACE uses the **www** port name. The second ACE uses the port number **80**. Both ACEs achieve exactly the same result.

```
R1(config)# access-list 100 permit tcp any any eq www
!or...
R1(config)# access-list 100 permit tcp any any eq 80
```

Configuring the port number is required when there is not a specific protocol name listed such as SSH (port number 22) or an HTTPS (port number 443), as shown in the next example.

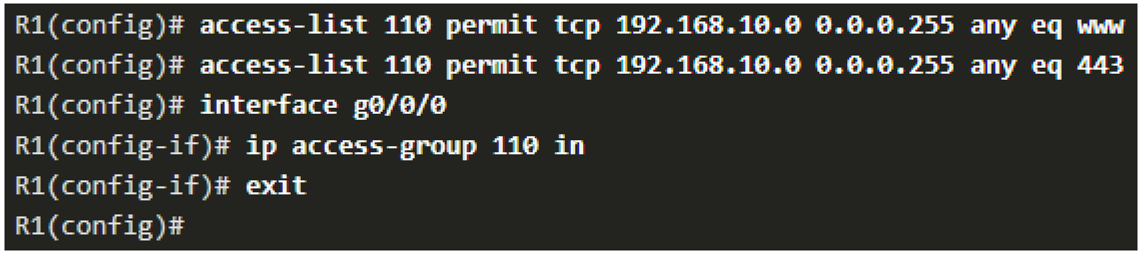
```
R1(config)# access-list 100 permit tcp any any eq 22
R1(config)# access-list 100 permit tcp any any eq 443
R1(config)#
```

Configure Extended IPv4 ACLs

Apply a Numbered Extended IPv4 ACL

In this example, the ACL permits both HTTP and HTTPS traffic from the 192.168.10.0 network to go to any destination.

Extended ACLs can be applied in various locations. However, they are commonly applied close to the source. Here ACL 110 is applied inbound on the R1 G0/0/0 interface.



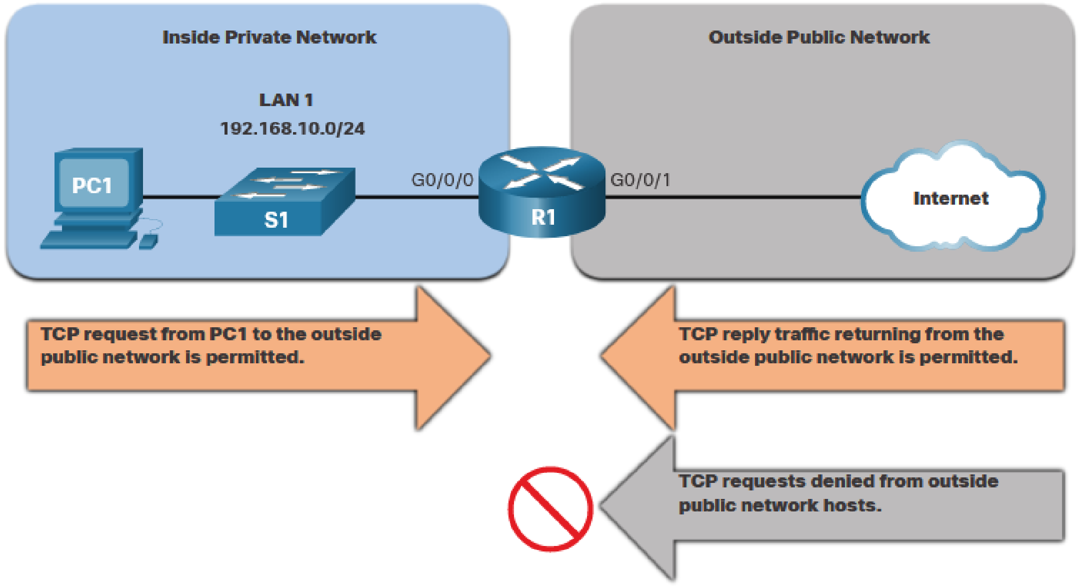
```
R1(config)# access-list 110 permit tcp 192.168.10.0 0.0.0.255 any eq www
R1(config)# access-list 110 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config)# interface g0/0/0
R1(config-if)# ip access-group 110 in
R1(config-if)# exit
R1(config)#
```

Configure Extended IPv4 ACLs

TCP Established Extended ACL

TCP can also perform basic stateful firewall services using the TCP **established** keyword.

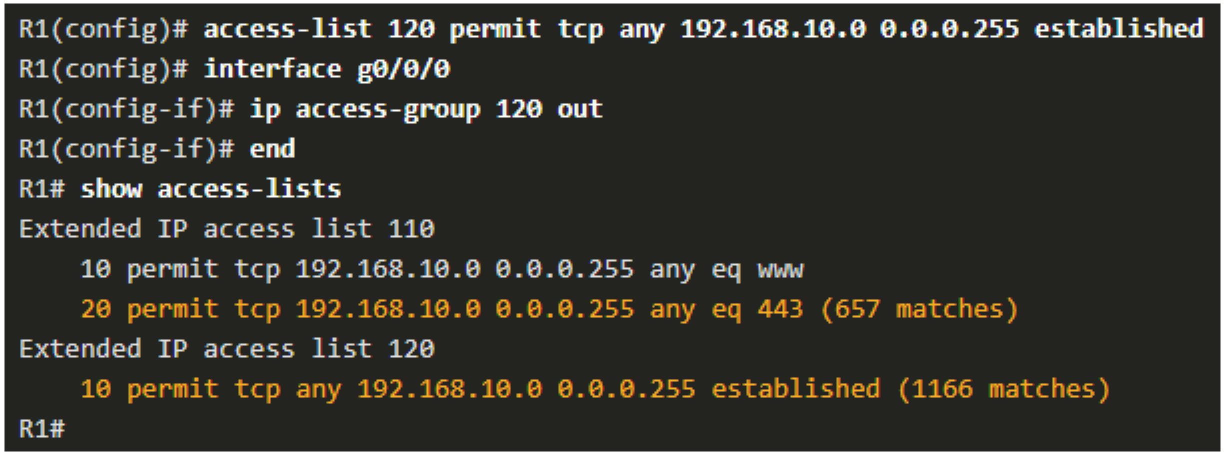
- The **established** keyword enables inside traffic to exit the inside private network and permits the returning reply traffic to enter the inside private network.
- TCP traffic generated by an outside host and attempting to communicate with an inside host is denied.


Inside Private Network
LAN 1
192.168.10.0/24
PC1
S1
G0/0/0
R1
G0/0/1
Outside Public Network
Internet
TCP request from PC1 to the outside public network is permitted.
TCP reply traffic returning from the outside public network is permitted.
TCP requests denied from outside public network hosts.

TCP Established Extended ACL (Cont.)

- ACL 120 is configured to only permit returning web traffic to the inside hosts. The ACL is then applied outbound on the R1 G0/0/0 interface.
- The **show access-lists** command shows that inside hosts are accessing the secure web resources from the internet.

Note: A match occurs if the returning TCP segment has the ACK or reset (RST) flag bits set, indicating that the packet belongs to an existing connection.



```
R1(config)# access-list 120 permit tcp any 192.168.10.0 0.0.0.255 established
R1(config)# interface g0/0/0
R1(config-if)# ip access-group 120 out
R1(config-if)# end
R1# show access-lists
Extended IP access list 110
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443 (657 matches)
Extended IP access list 120
    10 permit tcp any 192.168.10.0 0.0.0.255 established (1166 matches)
R1#
```

Configure Extended IPv4 ACLs

Named Extended IPv4 ACL Syntax

Naming an ACL makes it easier to understand its function. To create a named extended ACL, use the **ip access-list extended** configuration command.

In the example, a named extended ACL called NO-FTP-ACCESS is created and the prompt changed to named extended ACL configuration mode. ACE statements are entered in the named extended ACL sub configuration mode.

```
Router(config)# ip access-list extended access-list-name
```

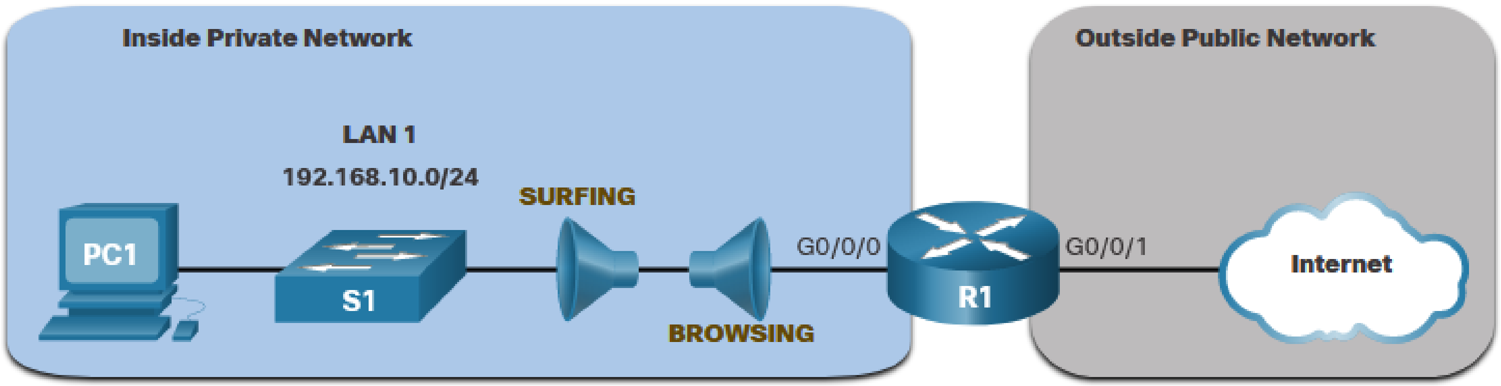
```
R1(config)# ip access-list extended NO-FTP-ACCESS
R1(config-ext-nacl)#
```

Configure Extended IPv4 ACLs

Named Extended IPv4 ACL Example

The topology below is used to demonstrate configuring and applying two named extended IPv4 ACLs to an interface:

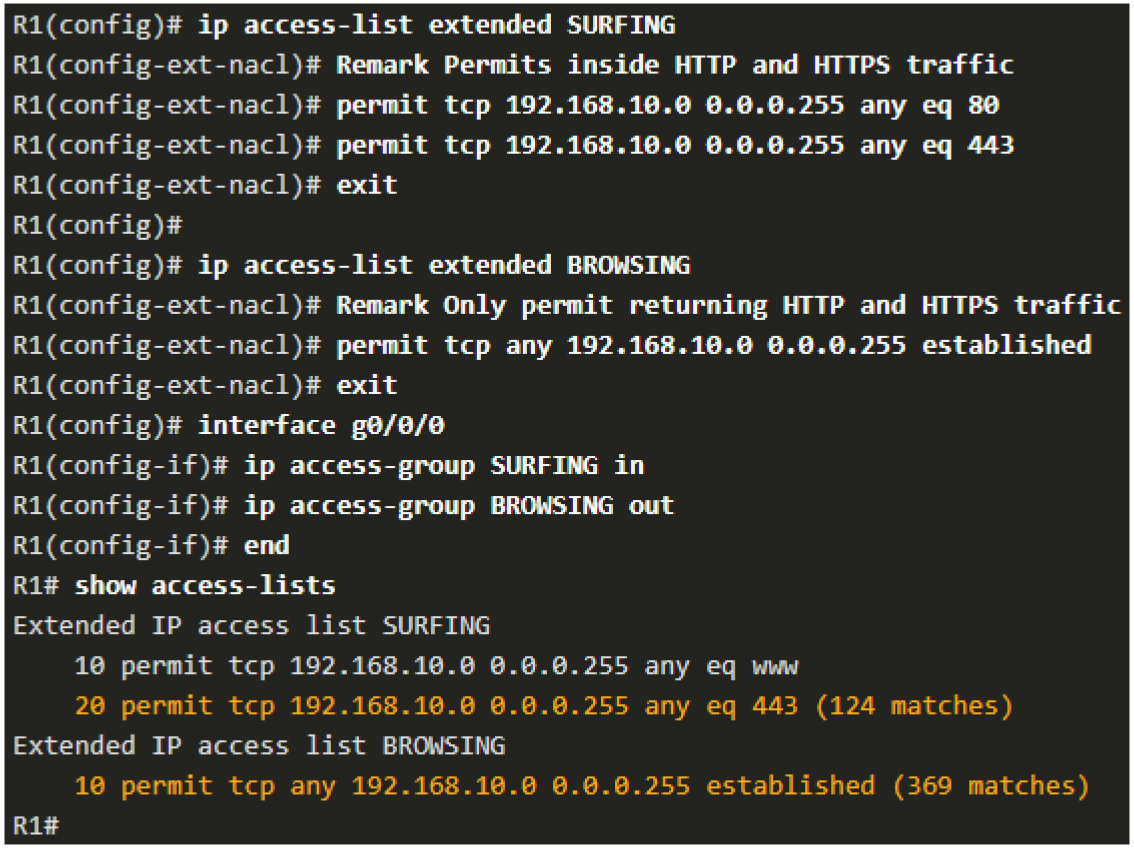
- **SURFING** - This will permit inside HTTP and HTTPS traffic to exit to the internet.
- **BROWSING** - This will only permit returning web traffic to the inside hosts while all other traffic exiting the R1 G0/0/0 interface is implicitly denied.



Configure Extended IPv4 ACLs

Named Extended IPv4 ACL Example (Cont.)

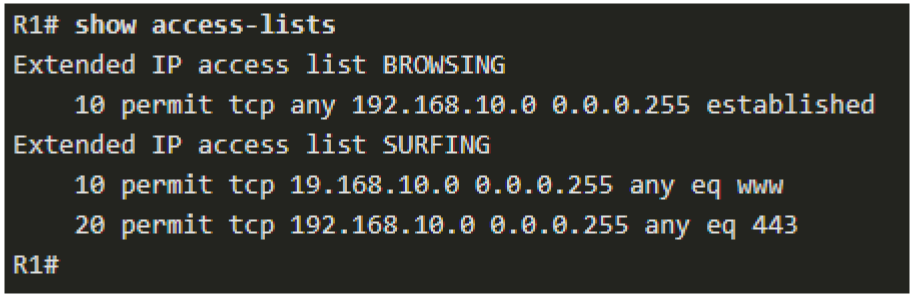
- The SURFING ACL permits HTTP and HTTPS traffic from inside users to exit the G0/0/1 interface connected to the internet. Web traffic returning from the internet is permitted back into the inside private network by the BROWSING ACL.
- The SURFING ACL is applied inbound and the BROWSING ACL is applied outbound on the R1 G0/0/0 interface.



```
R1(config)# ip access-list extended SURFING
R1(config-ext-nacl)# Remark Permits inside HTTP and HTTPS traffic
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config-ext-nacl)# exit
R1(config)#
R1(config)# ip access-list extended BROWSING
R1(config-ext-nacl)# Remark Only permit returning HTTP and HTTPS traffic
R1(config-ext-nacl)# permit tcp any 192.168.10.0 0.0.0.255 established
R1(config-ext-nacl)# exit
R1(config)# interface g0/0/0
R1(config-if)# ip access-group SURFING in
R1(config-if)# ip access-group BROWSING out
R1(config-if)# end
R1# show access-lists
Extended IP access list SURFING
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443 (124 matches)
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established (369 matches)
R1#
```

Named Extended IPv4 ACL Example (Cont.)

The show access-lists command is used to verify the ACL statistics. Notice that the permit secure HTTPS counters (i.e., eq 443) in the SURFING ACL and the return established counters in the BROWSING ACL have increased.



```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 19.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
```

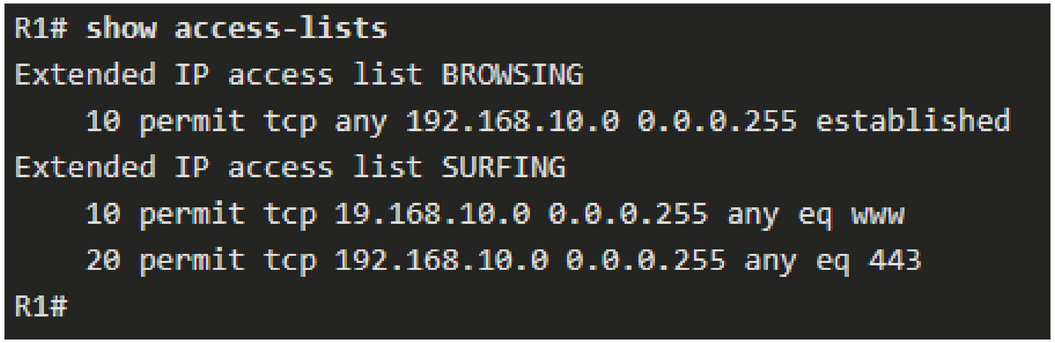
Configure Extended IPv4 ACLs

Edit Extended ACLs

An extended ACL can be edited using a text editor when many changes are required. Or, if the edit applies to one or two ACEs, then sequence numbers can be used.

Example:

- The ACE sequence number 10 in the SURFING ACL has an incorrect source IP networks address.



```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 19.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
```

Configure Extended IPv4 ACLs

Edit Extended ACLs (Cont.)

- To correct this error the original statement is removed with the **no** *sequence_#* command and the corrected statement is added replacing the original statement.
- The **show access-lists** command output verifies the configuration change.

```
R1# configure terminal
R1(config)# ip access-list extended SURFING
R1(config-ext-nacl)# no 10
R1(config-ext-nacl)# 10 permit tcp 192.168.10.0 0.0.0.255 any eq www
R1(config-ext-nacl)# end
```

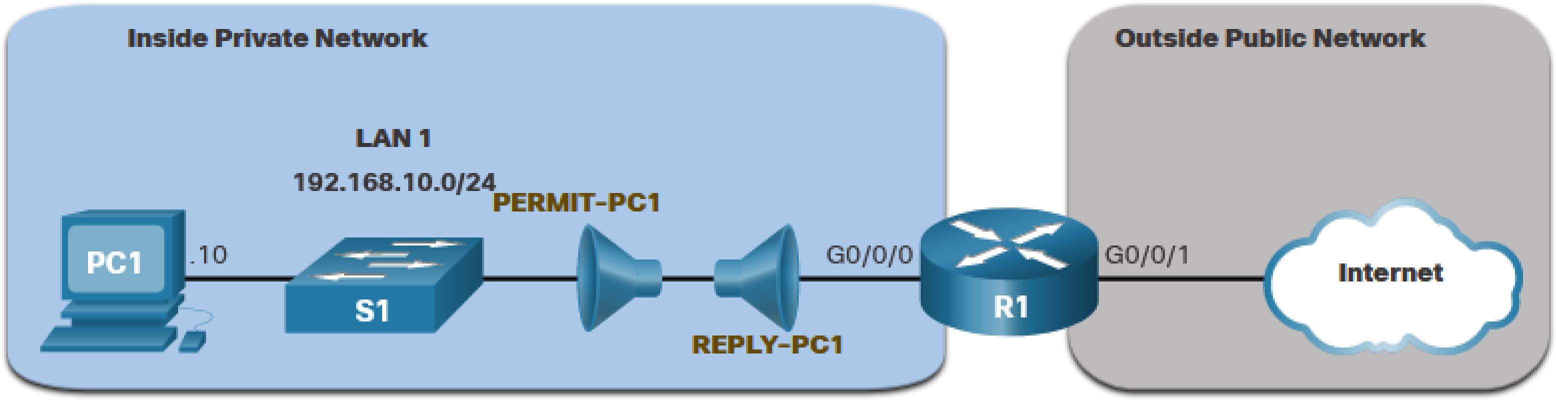
```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
```

Configure Extended IPv4 ACLs

Another Extended IPv4 ACL Example

Two named extended ACLs will be created:

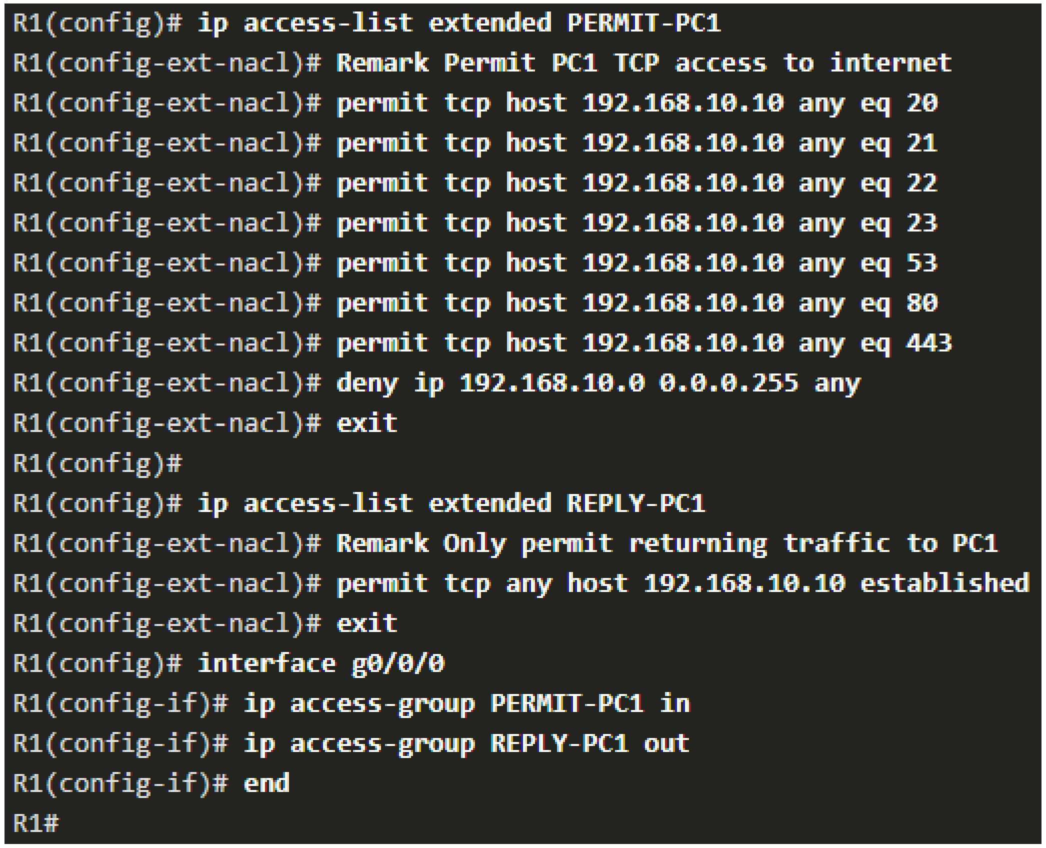
- **PERMIT-PC1** - This will only permit PC1 TCP access to the internet and deny all other hosts in the private network.
- **REPLY-PC1** - This will only permit specified returning TCP traffic to PC1 implicitly deny all other traffic.



Configure Extended IPv4 ACLs

Another Extended IPv4 ACL Example (Cont.)

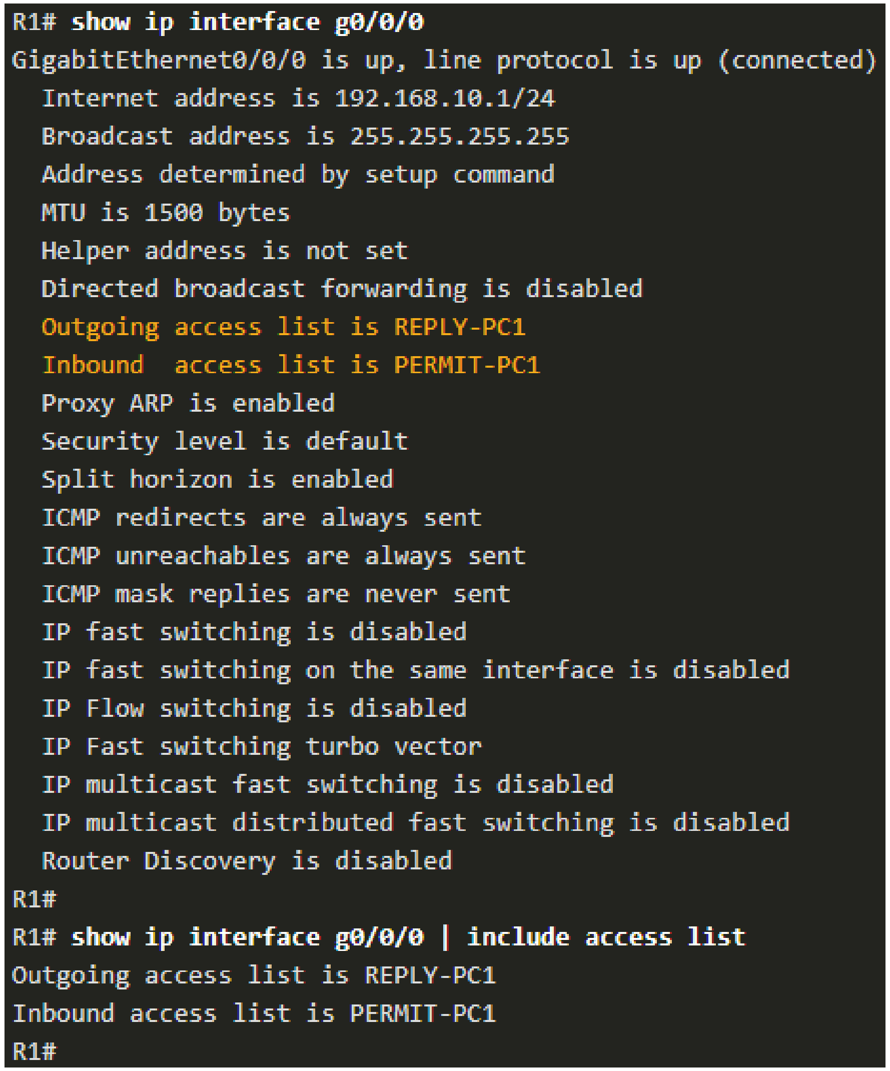
- The **PERMIT-PC1** ACL permits PC1 (192.168.10.10) TCP access to the FTP, SSH, Telnet, DNS , HTTP, and HTTPS traffic.
- The **REPLY-PC1** ACL will permit return traffic to PC1.
- The **PERMIT-PC1** ACL is applied inbound and the **REPLY-PC1** ACL applied outbound on the R1 G0/0/0 interface.



```
R1(config)# ip access-list extended PERMIT-PC1
R1(config-ext-nacl)# Remark Permit PC1 TCP access to internet
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 20
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 21
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 22
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 23
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 53
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 80
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 443
R1(config-ext-nacl)# deny ip 192.168.10.0 0.0.0.255 any
R1(config-ext-nacl)# exit
R1(config)#
R1(config)# ip access-list extended REPLY-PC1
R1(config-ext-nacl)# Remark Only permit returning traffic to PC1
R1(config-ext-nacl)# permit tcp any host 192.168.10.10 established
R1(config-ext-nacl)# exit
R1(config)# interface g0/0/0
R1(config-if)# ip access-group PERMIT-PC1 in
R1(config-if)# ip access-group REPLY-PC1 out
R1(config-if)# end
R1#
```

Verify Extended ACLs

The **show ip interface** command is used to verify the ACL on the interface and the direction in which it was applied.



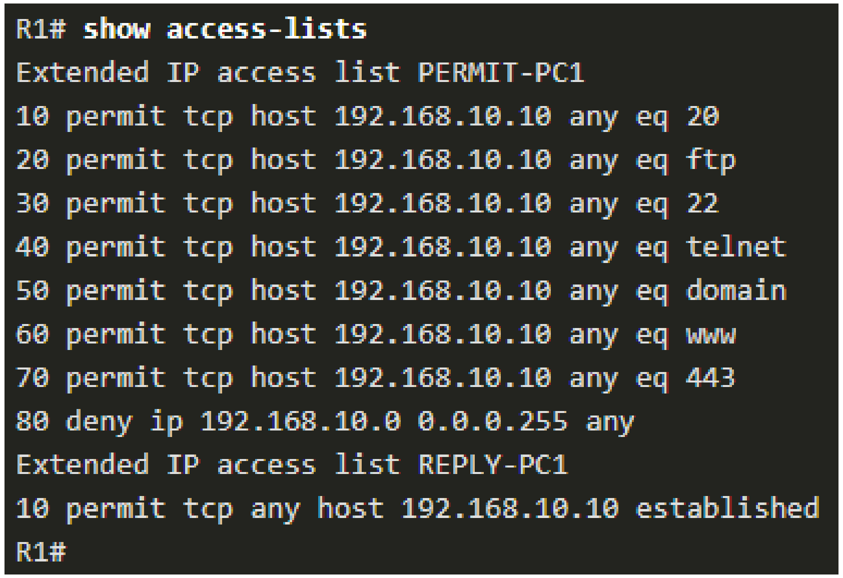
```
R1# show ip interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up (connected)
  Internet address is 192.168.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is REPLY-PC1
  Inbound  access list is PERMIT-PC1
  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 disabled
  IP fast switching on the same interface is disabled
  IP Flow switching is disabled
  IP Fast switching turbo vector
  IP multicast fast switching is disabled
  IP multicast distributed fast switching is disabled
  Router Discovery is disabled
R1#
R1# show ip interface g0/0/0 | include access list
Outgoing access list is REPLY-PC1
Inbound access list is PERMIT-PC1
R1#
```

Configure Extended IPv4 ACLs

Verify Extended ACLs (Cont.)

The **show access-lists** command can be used to confirm that the ACLs work as expected. The command displays statistic counters that increase whenever an ACE is matched.

Note: Traffic must be generated to verify the operation of the ACL.

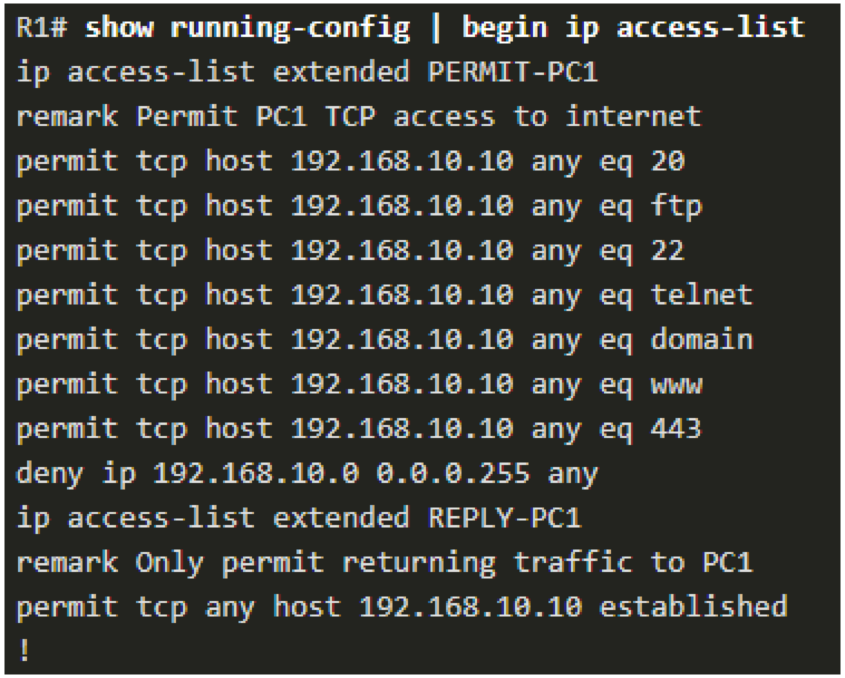


```
R1# show access-lists
Extended IP access list PERMIT-PC1
10 permit tcp host 192.168.10.10 any eq 20
20 permit tcp host 192.168.10.10 any eq ftp
30 permit tcp host 192.168.10.10 any eq 22
40 permit tcp host 192.168.10.10 any eq telnet
50 permit tcp host 192.168.10.10 any eq domain
60 permit tcp host 192.168.10.10 any eq www
70 permit tcp host 192.168.10.10 any eq 443
80 deny ip 192.168.10.0 0.0.0.255 any
Extended IP access list REPLY-PC1
10 permit tcp any host 192.168.10.10 established
R1#
```

Configure Extended IPv4 ACLs

Verify Extended ACLs (Cont.)

The **show running-config** command can be used to validate what was configured. The command also displays configured remarks.



```
R1# show running-config | begin ip access-list
ip access-list extended PERMIT-PC1
remark Permit PC1 TCP access to internet
permit tcp host 192.168.10.10 any eq 20
permit tcp host 192.168.10.10 any eq ftp
permit tcp host 192.168.10.10 any eq 22
permit tcp host 192.168.10.10 any eq telnet
permit tcp host 192.168.10.10 any eq domain
permit tcp host 192.168.10.10 any eq www
permit tcp host 192.168.10.10 any eq 443
deny ip 192.168.10.0 0.0.0.255 any
ip access-list extended REPLY-PC1
remark Only permit returning traffic to PC1
permit tcp any host 192.168.10.10 established
!
```

Today end,
See you next week!