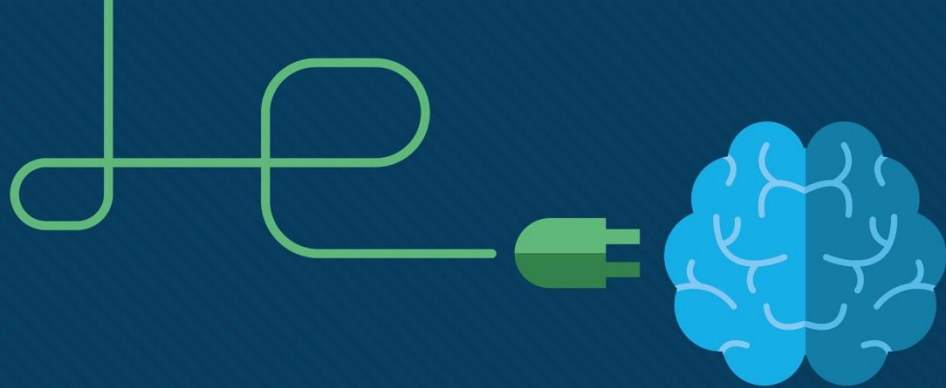




21 ACL



# 单元目标

模块主题：ACL 的配置

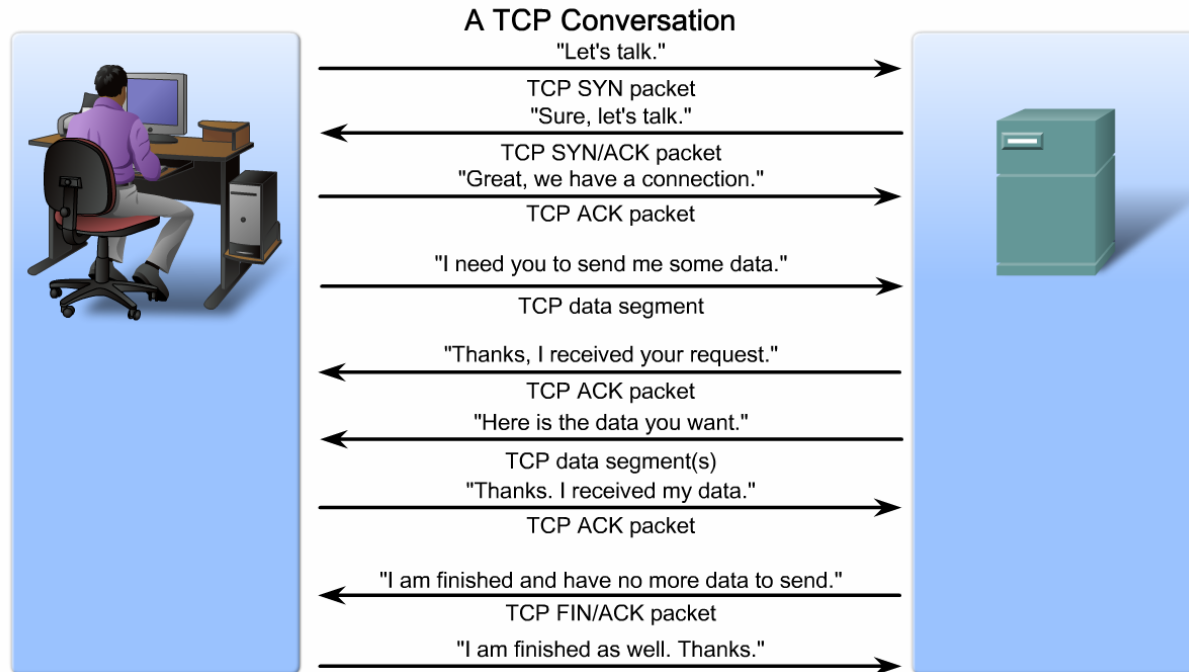
模块目标：说明如何在网络安全策略中使用 ACL。

主题标题	主题目标
ACL 的用途	介绍 ACL 如何过滤流量。
ACL 中的通配符掩码	说明 ACL 如何使用通配符掩码。
ACL 创建原则	说明如何创建 ACL。
IPv4 ACL 的配置	比较标准和扩展 IPv4 ACL。

# 21.1 ACL 的用途

# A TCP conversation

- ACLs enable you to **control traffic** into and out of your network.
- ACLs can be configured to control network traffic based on the **TCP and UDP port**.



# A TCP conversation

- Port Numbers

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)

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

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)

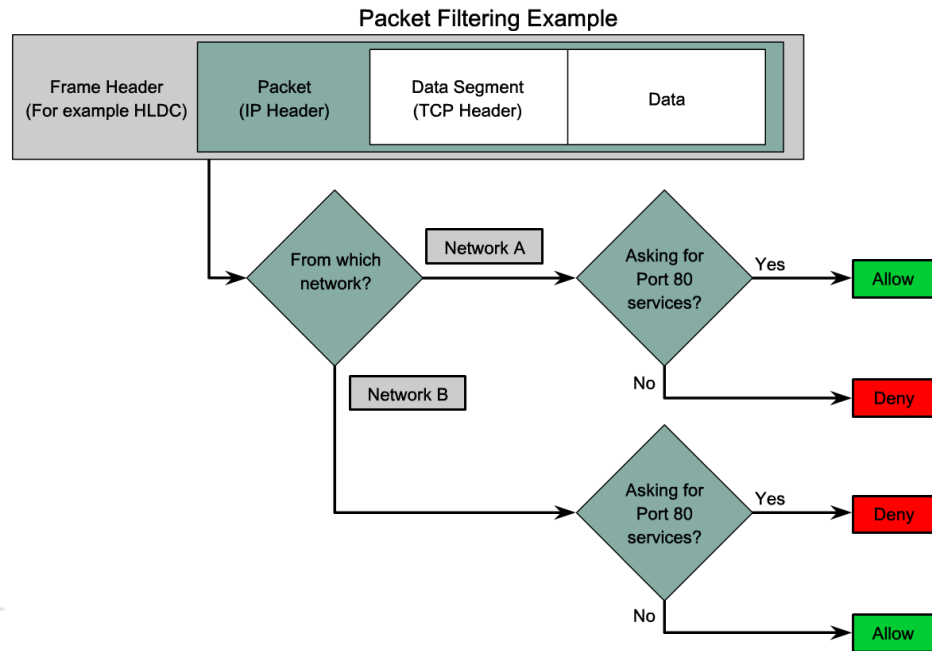
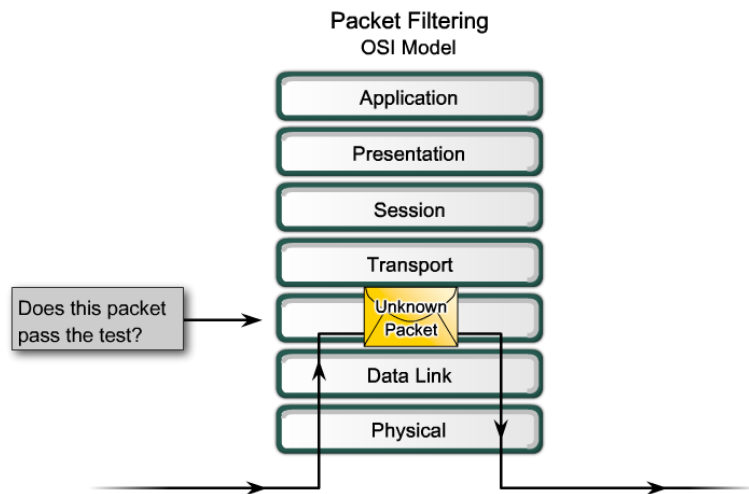
Well Known TCP/UDP Common Ports:  
53 - DNS  
161 - SNMP  
531 - AOL Instant Messenger, IRC

# Packet filtering

- A **router** acts as a packet filter when it **forwards** or **denies** packets according to **filtering rules**.
- The ACL can extract the following information from the packet header, test it against its rules, and make "**allow**" or "**deny**" decisions based on:
  - **Source IP** address
  - **Destination IP** address
  - **ICMP** message type
- The ACL can also extract upper layer information and test it against its rules. Upper layer information includes:
  - **TCP/UDP source** port
  - **TCP/UDP destination** port



# Packet filtering

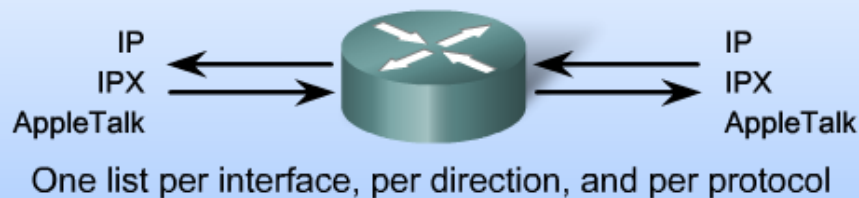


- For this scenario, the packet filter looks at each packet as follows:
  - If the packet is a TCP SYN from **network A** using **port 80**, it is **allowed** to pass. All other access is denied to those users.
  - If the packet is a TCP SYN from **network B** using **port 80**, it is **blocked**. However, all other access is permitted.

# ACL 的用途

## ACL 是什么？(续)

### ACL Traffic Filtering on a Router



With two interfaces and three protocols running, this router could have a total of 12 separate ACLs applied.

#### The three Ps for using ACLs

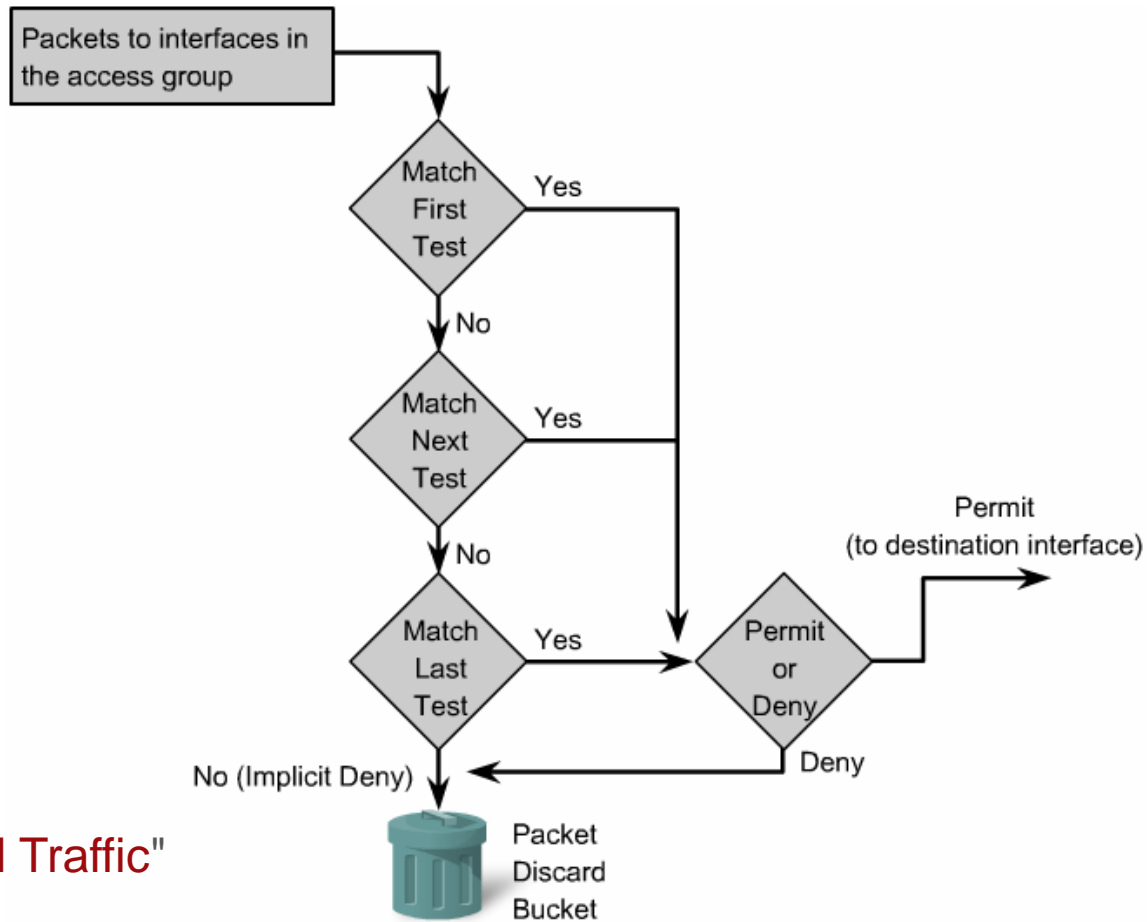
You can only have one ACL per protocol, per interface, and per direction:

- One ACL per protocol (e.g., IP or IPX)
- One ACL per interface (e.g., FastEthernet0/0)
- One ACL per direction (i.e., IN or OUT)



# ACL operation

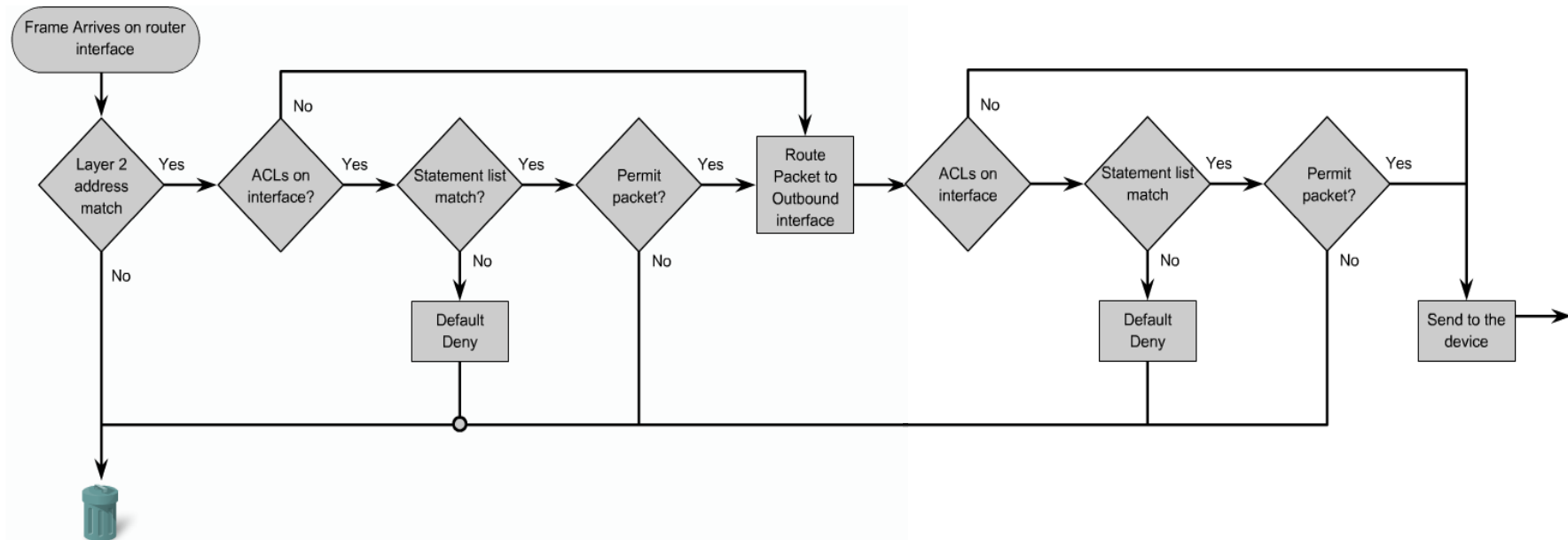
- How ACLs Work



- The implied "**Deny All Traffic**" Criteria Statement

# ACL operation

- ACL and Routing and **ACL Processes** on a Router



# 21.2 IPv4 ACL 的类型

# IPv4 ACL 的类型

## 标准 ACL 和扩展 ACL

IPv4 ACL 有以下两种类型：

- **标准 ACL** - 它们只会根据源IPv4地址来放行或拒绝数据包。
- **扩展 ACL** - 它们可以根据源 IPv4 地址和目的 IPv4 地址、协议类型、源和目的 TCP 或 UDP 端口号来放行和拒绝数据包。

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

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

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

- Source and destination IP addresses
- Source and destination TCP and UDP ports
- Protocol type (IP, ICMP, UDP, TCP, or protocol number)

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

# Numbering and Naming ACLs

- Starting with Cisco IOS Release 11.2, you can use a **name** to identify a Cisco ACL. It inform you of the purpose of the ACL.

## 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

## 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.

- Numbers 200 to 1299 are used by **other protocols**. For example, numbers 600 to 699 are used by AppleTalk, and numbers 800 to 899 are used by IPX.

- 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.

# ACL 的类型

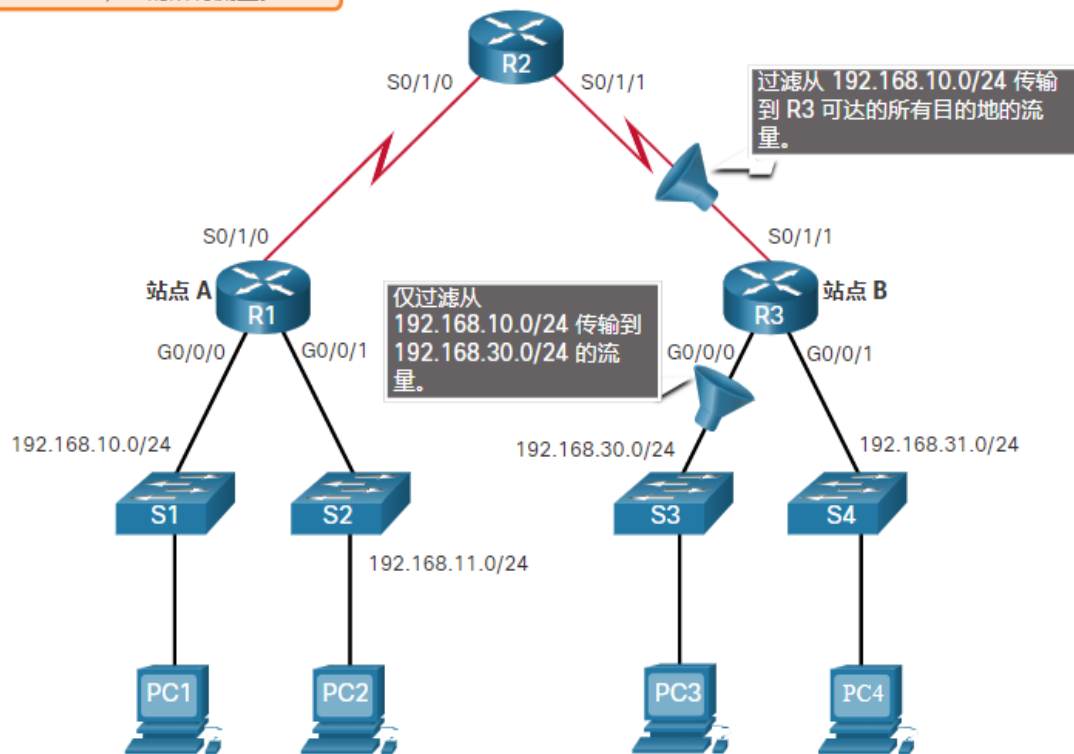
## 标准ACL的应用示例

在图中，管理员希望阻止源自 192.168.10.0/24 网络的流量到达 192.168.30.0/24 网络。

R3上有两个接口都可以应用标准ACL：

- **R3 S0/1/1 接口（入站）**
- **R3 G0/0/0 接口（出站）**

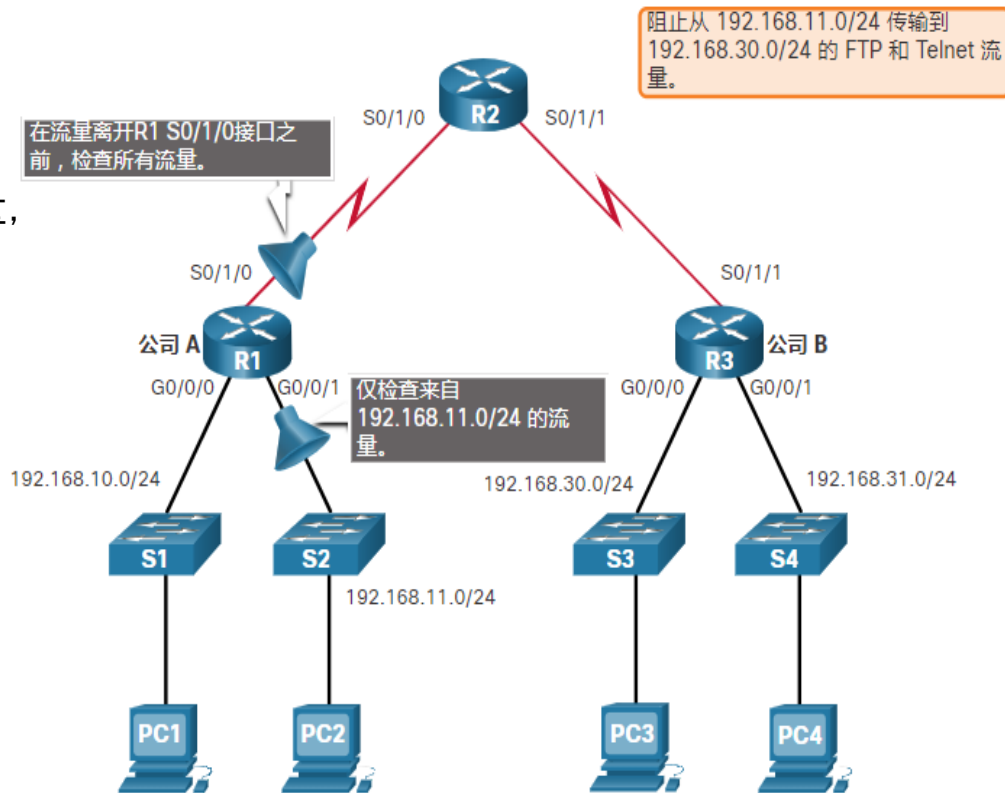
拦截从 192.168.10.0/24 传输到 192.168.30.0/24 的所有流量。



# ACL 的类型

## 扩展ACL的应用示例

- 公司A希望拒绝以下流量:从本地192.168.11.0/24网络去往公司B 192.168.30.0/24网络的Telnet和FTP流量,同时放行所有其他流量。
- R1上有两个接口都可以应用扩展ACL
  - R1 S0/1/0 接口(出站)
  - R1 G0/0/1 接口(入站)





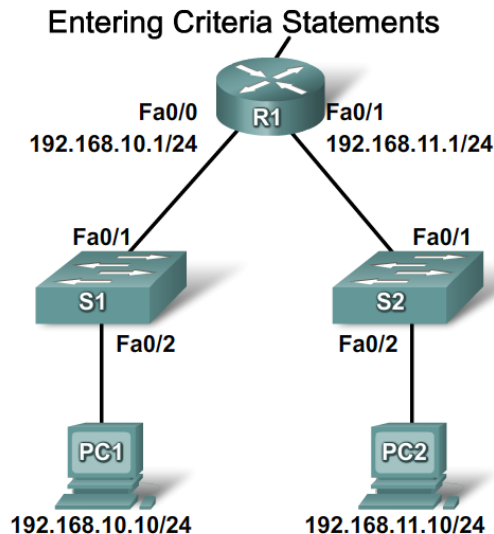
## 21.3 配置标准IPv4 ACL

# Entering Criteria Statements

## Something is Important:

- You should have the **most frequently** used **ACL** entry at the **top of the list**.
- You must have **at least one permit** statement in an ACL or all traffic is blocked.

For example, the two ACLs (101 and 102) in the figure have the **same effect**.



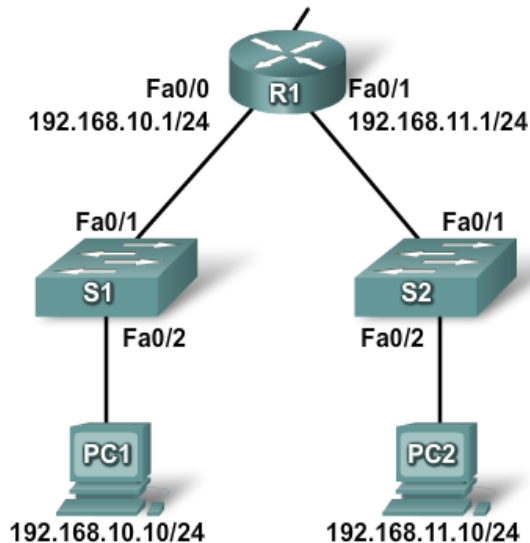
### ACL 101

```
access-list 101 permit ip 192.168.10.0 0.0.0.255 192.168.30.0 0.0.0.255
```

### ACL 102

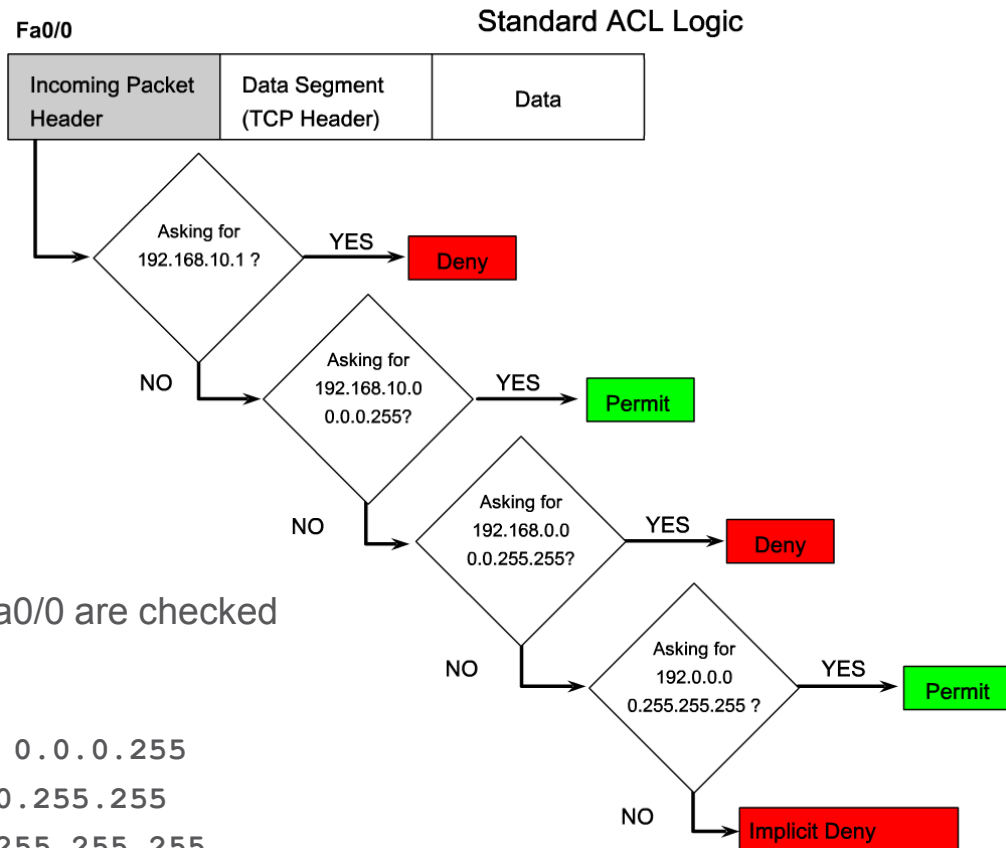
```
access-list 102 permit ip 192.168.10.0 0.0.0.255 192.168.30.0 0.0.0.255
access-list 102 deny ip any any
```

# Configuring a standard ACL



- In the figure, packets that come in R1's Fa0/0 are checked for their **source addresses**:

- `access-list 2 deny 192.168.10.1`
- `access-list 2 permit 192.168.10.0 0.0.0.255`
- `access-list 2 deny 192.168.0.0 0.0.255.255`
- `access-list 2 permit 192.0.0.0 0.255.255.255`



# Configuring a standard ACL

- The full syntax of the standard ACL command is as follows:

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

```
R1# show access-list  
Standard IP access list 10  
  10 permit 192.168.10.0  
R1#  
R1# conf t  
Enter configuration commands, one per line. E  
R1(config)# no access-list 10  
R1(config)# exit  
R1#  
*Oct 25 19:59:41.142: %SYS-5-CONFIG_I: Configu  
R1# show access-list  
R1#
```

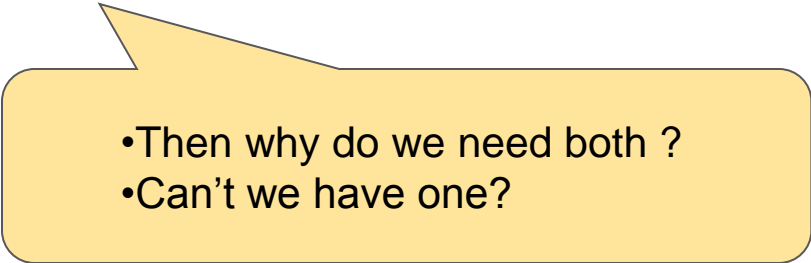
Remove ACL

```
R1# conf t  
Enter configuration commands, one per line. End with CNTL/Z.  
R1(config)# access-list 10 remark Permit hosts from the 192.168.10.0 LAN  
R1(config)# access-list 10 permit 192.168.10.0  
R1(config)# exit  
R1#  
*Oct 25 20:12:13.781: %SYS-5-CONFIG_I: Configured from console by console  
R1# show run  
Building configuration...  
!  
<output omitted>  
!  
access-list 10 remark Permit hosts from the 192.168.10.0 LAN  
access-list 10 permit 192.168.10.0
```

Remark

# ACL Wildcard Masking

- Subnet Mask and Wild Card Mask
  - Most important point is – Both do bit pattern matching
  - Subnet Mask : 255.255.255.0
  - Wild card Mask: 0.255.255.255

- 
- Then why do we need both ?
  - Can't we have one?

# ACL Wildcard Masking

- Subnet Mask

- Take this network – 11.22.33.44/**24**



- What does subnet mask /24 do?

- Why we have to separate?
  - We live in a classless world.
  - Suppose we had only classfull networks
  - No need of Subnet mask 😊
  - But ... We live in a Classless world

# ACL Wildcard Masking

- Next – Does this Subnet mask work
  - 11.22.33.44 **255.255.255.0**
  - Yes or No
  - 11.22.33.44 **0.255.0.255**
    - 0 means – Host part
    - 255 means network part
  - Yes or No

# ACL Wildcard Masking

- **Discontiguous Subnet Mask** are not allowed.
- Subnet mask's **only** function is to separate network part and host part – **That's it**
- But there are other features for which we need something more than Subnet mask.
- Something which gives much more finer control over bits than subnet mask.
- Something which allows discontiguous bit(s) matching capability
  - Enter **WILD CARD MASK** 😊



# ACL Wildcard Masking

- Wild card mask does bit pattern matching only.

That's it.

No separation of host part and network part

Just bit matching –

- 0 means match
- 255 do not match **or** don't care

Note: **WILD CARD MASK IS NOT THE INVERSE OF SUBNET MASK.**

# ACL Wildcard Masking

- Some simple wild card masks
- 0.255.255.255
  - Match the first byte and ignore the rest of bits

- Easy one

0.255.255.0

- Match the first byte and the last byte and ignore the rest of bits

# ACL Wildcard Masking

- Wildcard masks Examples:

	Decimal	Binary
IP Address	192.168.16.0	11000000.10101000.00010000.00000000
Wildcard Mask	0.0.15.255	00000000.00000000.00001111.11111111
Result Range	192.168.16.0 to 192.168.31.0	11000000.10101000.00010000.00000000 to 11000000.10101000.00011111.00000000

	Decimal	Binary
IP Address	192.168.1.0	11000000.10101000.00000001.00000000
Wildcard Mask	0.0.254.255	00000000.00000000.11111110.11111111
Result	192.168.1.0	11000000.10101000.00000001.00000000
	All odd numbered subnets in the 192.168.0.0 major network	

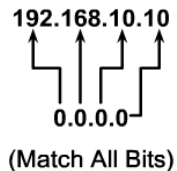
# ACL Wildcard Masking

## Wildcard Bit Mask Abbreviations

### Example 1:

- 192.168.10.10 0.0.0.0 matches all of the address bits
- Abbreviate this wildcard mask using the IP address preceded by the keyword host (host 192.168.10.10)

Wildcard Mask:



### Example 2:

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

Wildcard Mask:



### Example 1:

```
R1(config)#access-list 1 permit 0.0.0.0 255.255.255.255  
R1(config)#access-list 1 permit any
```

### Example 2:

```
R1(config)#access-list 1 permit 192.168.10.10 0.0.0.0  
R1(config)#access-list 1 permit host 192.168.10.10  
R1(config)#access-list 1 permit 192.168.10.10
```

# Apply Standard ACLs to Interface

- After a standard ACL is configured, it is linked to an **interface** using the `ip access-group` command:

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

## Procedure for Configuring Standard ACLs

**Step 1** Use the `access-list` global configuration command to create an entry in a standard IPv4 ACL.

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

Enter the global `no access-list` command to remove the entire ACL. The example statement matches any address that starts with 192.168.10.x. Use the `remark` option to add a description to your ACL.

**Step 2** Use the interface configuration command to select an interface to which to apply the ACL

```
R1(config)# interface FastEthernet 0/0
```

**Step 3** Use the `ip access-group` interface configuration command to activate the existing ACL on an interface.

```
R1(config-if)# ip access-group 1 out
```

To remove an IP ACL from an interface, enter the `no ip access-group` command on the interface. This example activates the standard IPv4 ACL 1 on the interface as an outbound filter.

# Apply Standard ACLs to Interface

```
R1(config)# access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)# interface S0/0/0
R1(config-if)# ip access-group 1 out
```

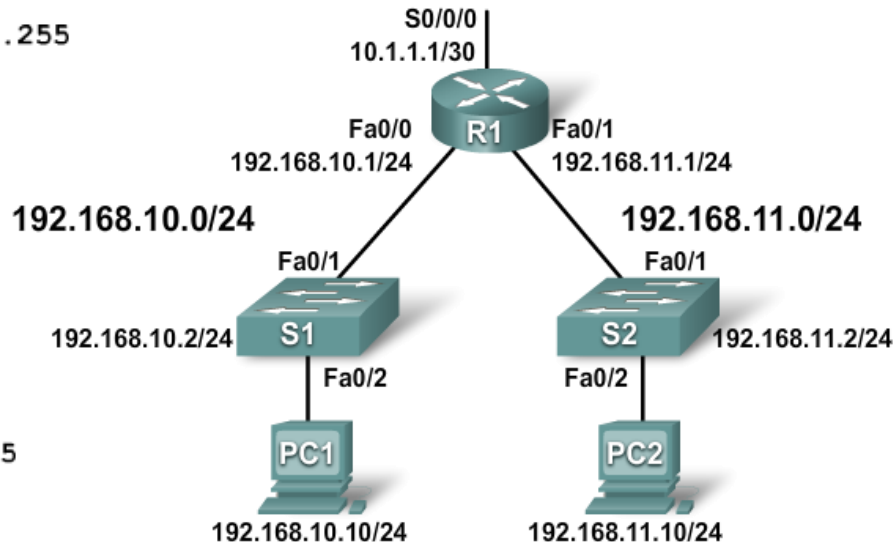
- **Example1:** an ACL to **permit** a single network.

```
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)#interface S0/0/0
R1(config-if)#ip access-group 1 out
```

- **Example2:** an ACL that **denies** a specific host.

```
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)#interface S0/0/0
R1(config-if)#ip access-group 1 out
```

- **Example3:** an ACL that **denies** a specific subnet.



## 编号的标准 ACL 示例(续)

- 我们可以使用 **show running-config** 命令来查看路由器配置中的 ACL。
- 我们可以使用 **show ip interface** 命令来验证接口上应用的ACL。

```
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#
```

# 配置标准 IPv4 ACL

## 命名的标准 ACL 示例

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

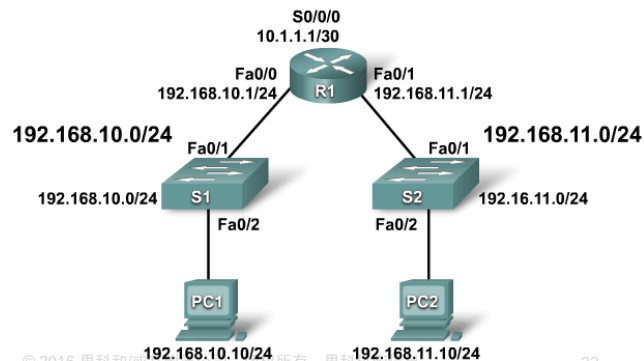
- Alphanumeric name string must be unique and cannot begin with a number

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

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

- Activates the named IP ACL on an interface

Named ACL Example





## 配置标准 IPv4 ACL

# 命名的标准 ACL 示例

本例中的ACL放行了来自主机192.168.10.10的流量, 以及192.168.20.0/24网络中素有主机的流量, 允许它们从路由器R1的S0/1/0接口发出。

```
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#
```

## 命名的标准 ACL 示例(续)

- 我们可以使用 **show access-list** 命令来查看配置中的 ACL。
- 我们可以使用 **show ip interface** 命令来验证接口上应用的ACL。

```
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#
```

# 21.4 修改 IPv4 ACL

## 修改 ACL 的两种方法

在配置好ACL后，我们可能会需要对它进行修改。一个拥有多条ACE的ACL配置可能相当复杂。有时我们配置的ACE并不能精确地产生我们想要的结果。

修改 ACL 有以下两种方法：

- 使用文本编辑器
- 使用序列号

# 修改 IPv4 ACL 使用文本编辑器

我们应该在文本编辑器中创建具有多条ACE的ACL。您可以规划所需的ACE、创建ACL，然后把它粘贴到路由器接口上。这种方法简化了编辑和优化ACL的工作。

我们可以这样修改ACL中的错误：

- 从运行配置中复制这个ACL，把它粘贴到文本编辑器中。
- 执行必要的编辑更改。
- 删除路由器上已配置的ACL。
- 把编辑后的ACL复制并粘贴到路由器中。

```
R1# show run | section access-list
access-list 1 deny 192.168.10.10
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)#
```

# 修改 IPv4 ACL 使用序列号

我们还可以使用ACL序列号来删除或添加ACE。

- 我们可以使用 **ip access-list standard** 命令来编辑 ACL。
- 使用与现有语句相同的序列号并不能覆盖语句。我们必须首先使用 **no 10** 命令删除当前的语句。然后, 可以再使用序列号来添加正确的 ACE 配置。

```
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#
```

# 修改命名 ACL 的示例

对于命名的ACL, 我们也可以使用序列号来删除和添加ACE。本例中添加了一个 ACE 来拒绝主机 192.168.10.11的流量。

```
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#
```

## 修改 IPv4 ACL ACL 的统计信息

示例中的 **show access-lists** 命令显示出了每个已匹配的语句的统计信息。

- deny语句匹配了20次, permit语句匹配了64次。
- 需要注意的是, ACL末尾的隐式deny any语句不会显示任何统计信息。要想知道有多少数据包匹配了隐式deny语句, 我们必须在ACL末尾手动配置 **deny any** 命令。
- 我们可以使用 **clear access-list counters** 命令来清除 ACL 统计信息。

```
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#
```



# 21.5 使用标准 IPv4 ACL 保护 VTY 端口

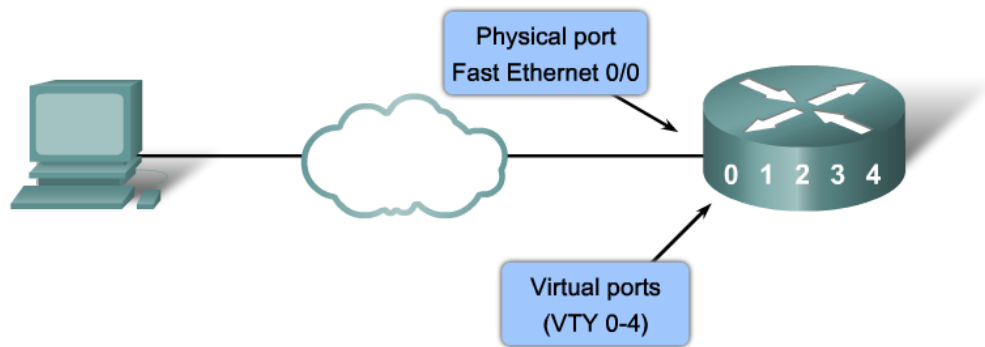
# 使用标准 IPv4 ACL 保护 VTY 端口

## access-class 命令

我们可以使用标准ACL来保护vty线路的远程管理访问，具体分为以下两个步骤：

- 创建ACL，在其中指定有哪些管理主机能够发起远程访问。
- 在vty线路上针对入站流量应用ACL。

Standard ACLs to Control Virtual Terminal Access



```
R1(config)#access-list 21 permit 192.168.10.0 0.0.0.255
R1(config)#access-list 21 permit 192.168.11.0 0.0.0.255
R1(config)#access-list 21 deny any

R1(config)#line vty 0 4
R1(config-line)#login
R1(config-line)#password secret
R1(config-line)#access-class 21 in
```

## 使用标准 IPv4 ACL 保护 VTY 端口 保护 VTY 访问的示例

本例展示出如何配置ACL来过滤vty流量。

- 首先在本地数据库中配置用户 **ADMIN** 和密码 **class** 。
- 示例中配置了R1 vty线路来使用本地数据库做认证，放行SSH流量，使用ADMIN-HOST ACL来过滤流量。

```
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#
```

## 使用标准 IPv4 ACL 保护 VTY 端口 验证 VTY 端口是否安全

在 vty 线路上使用 ACL 对方向进行了限制后，我们一定要验证它的效果是否于预期相同。

要想验证 ACL 的统计信息，可以使用 **show access-lists** 命令。

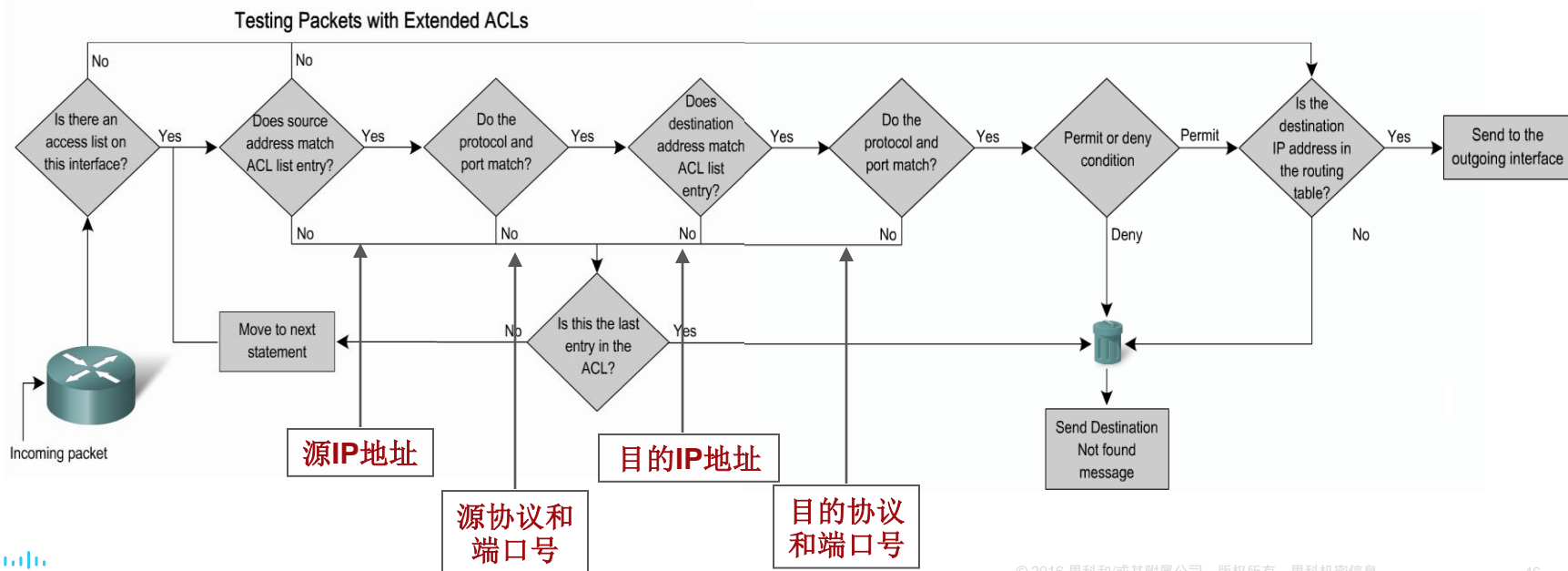
- 输出中 permit 语句的匹配项是因主机 192.168.10.10 的 SSH 连接成功而产生的。
- deny 语句的匹配项是因另一个网络中的一个设备尝试创建 SSH 连接失败而产生的。

```
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#
```

# 21.6 配置扩展IPv4 ACL

# 配置扩展IPv4 ACL 扩展ACL

For more precise traffic-filtering control, you can use extended ACLs numbered **100 to 199** and **2000 to 2699** providing a total of **800** possible extended ACLs. Extended ACLs can also be **named**.



# Configuring Extended ACLs

- The procedural steps for configuring extended ACLs are the same as for standard ACLs-you **first create the extended ACL** and then **activate it on an interface**.

## Configuring Extended ACLs

```
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]
```

Parameter	Description
<i>access-list-number</i>	Identifies the access list using a number in the range 100 to 199 (for an extended IP ACL) and 2000 to 2699 (expanded IP ACLs).
<i>deny</i>	Denies access if the conditions are matched.
<i>permit</i>	Permits access if the conditions are matched.
<i>remark</i>	Indicates whether this entry allows or blocks the specified address. Could also be used to enter a remark.
<i>protocol</i>	Name or number of an Internet protocol. Common keywords include <i>icmp</i> , <i>ip</i> , <i>tcp</i> , or <i>udp</i> . To match any Internet protocol (including ICMP, TCP, and UDP) use the <i>ip</i> keyword.
<i>source</i>	Number of the network or host from which the packet is being sent.
<i>source-wildcard</i>	Wildcard bits to be applied to source.
<i>destination</i>	Number of the network or host to which the packet is being sent.
<i>destination-wildcard</i>	Wildcard bits to be applied to the destination.
<i>operator</i>	(Optional) Compares source or destination ports. Possible operands include <i>lt</i> (less than), <i>gt</i> (greater than), <i>eq</i> (equal), <i>neq</i> (not equal), and <i>range</i> (inclusive range).
<i>port</i>	(Optional) The decimal number or name of a TCP or UDP port.
<i>established</i>	(Optional) For the TCP protocol only: Indicates an established connection.

## 协议选项

扩展ACL可以针对不同类型的互联网协议和端口进行过滤。在输入复杂的ACE时,可以使用?获取帮助。突出显示的四个协议是最常使用的选项。

```
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
```



# 配置扩展 IPv4 ACL 协议和端口(续)

协议的选择会影响端口选项。这里有很多TCP端口选项可用，如下例输出所示。

## Using port numbers

```
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 22
```

## Using keywords

```
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
```

## Generating Port Numbers

```
R1(config)#access-list 101 permit tcp any eq ?
```

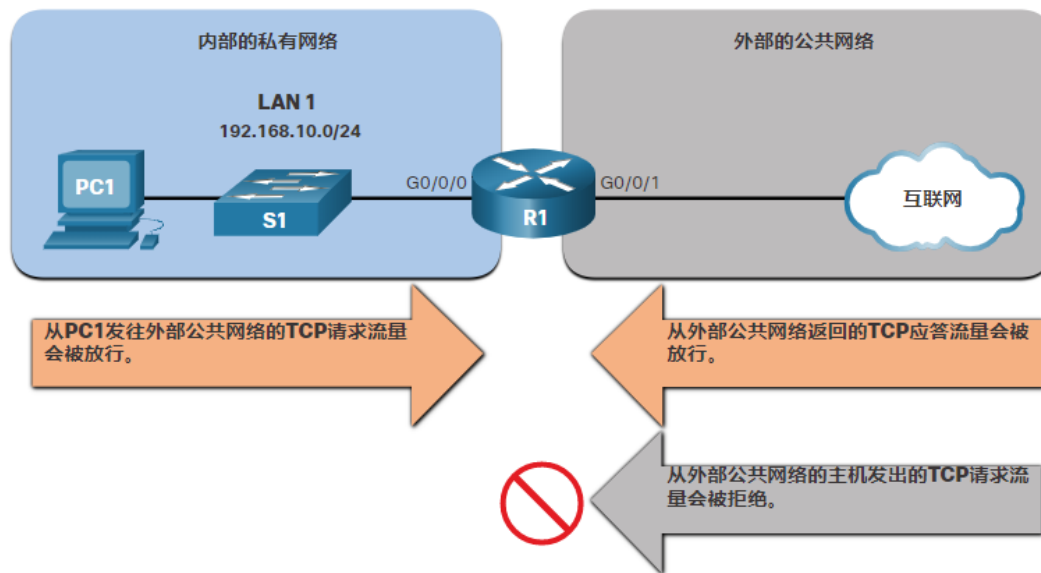
```
<0-65535> Port number
bgp Border Gateway Protocol (179)
chargen Character generator (19)
cmd Remote commands (rcmd, 514)
daytime Daytime (13)
discard Discard (9)
domain Domain Name Service (53)
drip Dynamic Routing Information Protocol (3949)
echo Echo (7)
exec Exec (rsh, 512)
finger Finger (79)
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)
nntp Network News
Transport Protocol (119)
pim-auto-rp PIM Auto-RP (496)
pop2 Post Office Protocol v2 (109)
```

## 配置扩展 IPv4 ACL

# TCP Established扩展 ACL

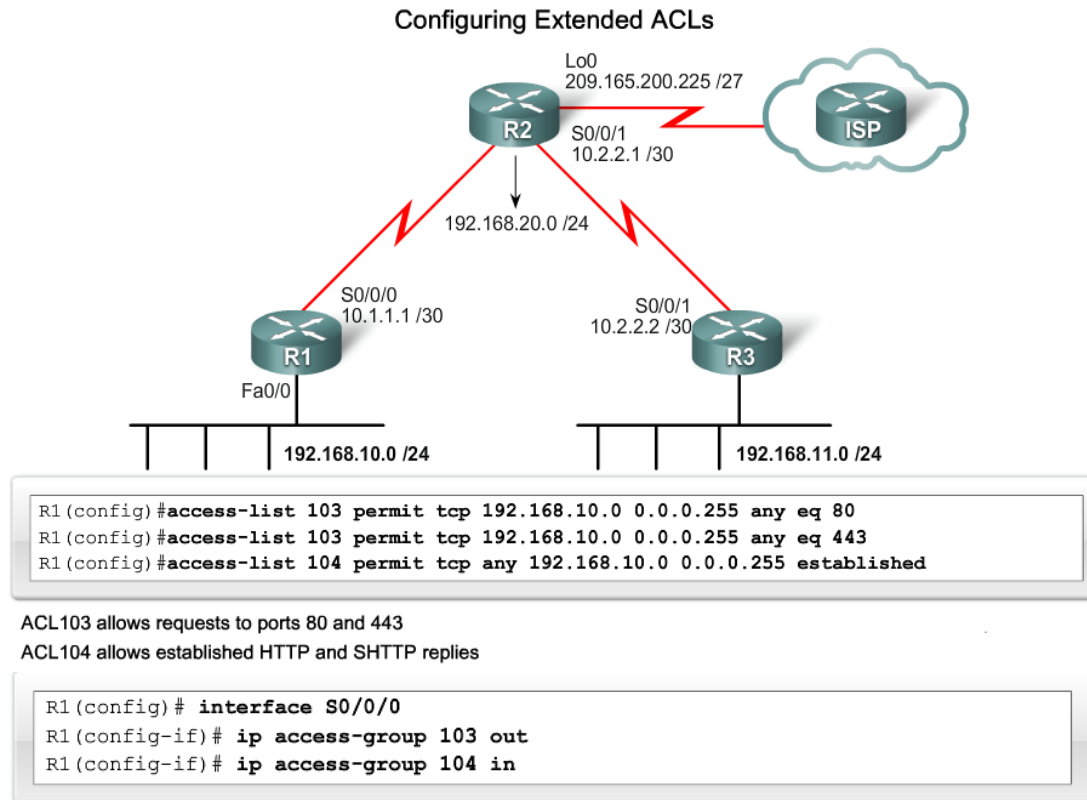
keyword. 我们可以使用 **TCP established** 关键字, 通过TCP来执行基本的状态化防火墙服务。

- **established** 关键字可以让内部流量离开内部的私有网络, 同时允许返回的应答流量进入这个内部的私有网络。
- 外部主机生成TCP流量并尝试与内部主机进行通信, 这个流量会被拒绝。



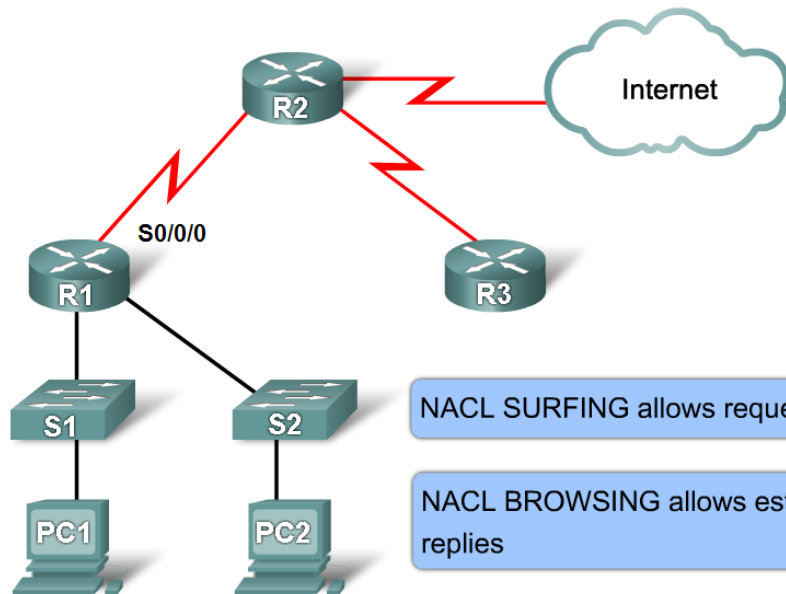
# Configuring Extended ACLs

- Extended ACL example:



# Creating named extended ACLs

## Configuring Named Extended ACLs



NACL SURFING allows requests to ports 80 and 443

NACL BROWSING allows established HTTP and SHTTP replies

```
R1(config)# access-list extended SURFING
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 0.0.0.225 any eq 443
R1(config)# access-list extended BROWSING
R1(config-ext-nacl)# permit tcp any 192.168.10.0 0.0.0.255 established
```

# ACL summary

- 一个协议在接口的一个方向上只能有一个ACL (in,out)
- 缺省deny ( implicit deny)
- 注意精确匹配.
- 先检查是否匹配, 再决定是permit还是deny
- 不要在ACL在端口激活的情况下修改编辑语句 (注意配置顺序) !!!
- Numbered ACL不能插入和删除一行 (最新版本的IOS 12.2(14)S支持插入和删除)
- 使用 no access-list x 将删除整个ACL列表
- 如果数据包被拒绝, 将返回给发送者一个 ICMP host unreachable的信息
- ACL不能对路由器本身产生的包向外发送时进行过滤

# Packet Tracer – 配置扩展 IPv4 ACL - 场景 1

在此 Packet Tracer 中，您将完成以下目标：

- 配置、应用并验证编号的扩展IPv4 ACL
- 配置、应用并验证命名的扩展IPv4 ACL

## Packet Tracer — 配置扩展 IPv4 ACL - 场景 2

在此 Packet Tracer 中，您将完成以下目标：

- 配置命名的扩展 IPv4 ACL
- 应用并验证扩展 IPv4 ACL

# 21.7 单元练习与测验



# Packet Tracer — IPv4 ACL 实施挑战

在此 Packet Tracer 中，您将完成以下目标：

- 在路由器上配置命名的标准ACL
- 在路由器上配置命名的扩展ACL
- 在路由器上配置扩展ACL来满足特定的通信需求
- 配置ACL来控制对网络设备终端线路的访问
- 在适当的路由器接口上，在适当的方向上配置ACL。
- 验证已配置 ACL 的运行工作

# 单元练习与测验

## 实验 - 配置和验证扩展 IPv4 ACL

在本实验中，您将完成以下目标：

- 建立网络并配置设备的基本设置
- 配置并验证扩展IPv4 ACL

## 我在这个模块中学到了什么？

- 要创建编号的标准 ACL, 我们可以使用 **ip access-list standard access-list-name** 全局配置命令。
- 我们可以使用 **no access-list access-list-number** 全局配置命令来删除编号的标准 ACL。
- 我们可以使用 **show ip interface** 命令来验证接口是否应用了 ACL。
- 要创建编号的标准 ACL, 我们可以使用 **ip access-list standard access-list-name** 全局配置命令。
- 我们可以使用 **no ip access-list standard access-list-name** 全局配置命令来删除命名的标准 ACL。
- 要想在接口上应用编号或命名的标准IPv4 ACL, 我们可以使用 **ip access-group {access-list-number | access-list-name} { in | out }** 全局配置命令。
- 要想从接口删除ACL, 首先要使用 **no ip access-group** 接口配置命令。
- 要想从路由器配置中删除ACL, 需要使用**no access-list**全局配置命令。

## 我在这个模块中学到了什么？

- 扩展ACL可以过滤源地址、目的地址、协议(即 IP、TCP、UDP、ICMP)和端口号。
- 要想创建编号的扩展 ACL, 我们可以使用 Router(config)# **access-list access-list-number {deny | permit | remark text} protocol source source-wildcard [operator [port]] destination destination-wildcard [operator [port]] [established] [log]** 全局配置命令。
- 我们可以使用 TCP **established** 关键字, 通过TCP来执行基本的状态化防火墙服务。
- **show ip interface** 命令用于验证接口上的 ACL 及其应用方向。
- 要想修改 ACL, 我们可以使用文本编辑器或序列号。
- 我们还可以使用ACL序列号来删除或添加ACE。
- 在输入一条ACE时, 路由器会自动为它分配一个序列号。

