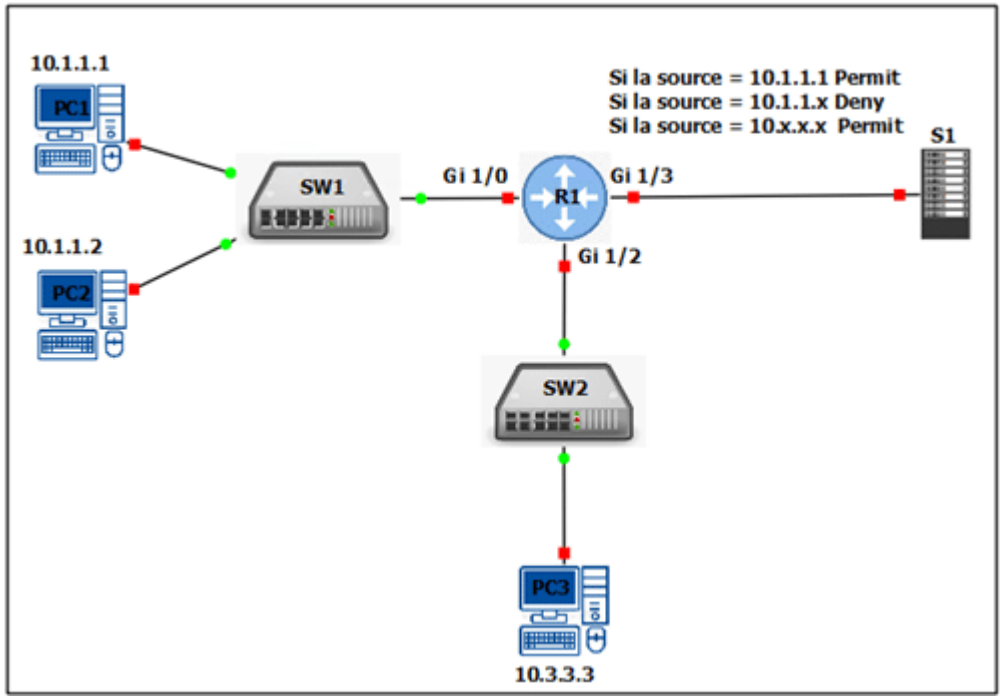
Access Control Lists

Berrehouma N.

Defintions

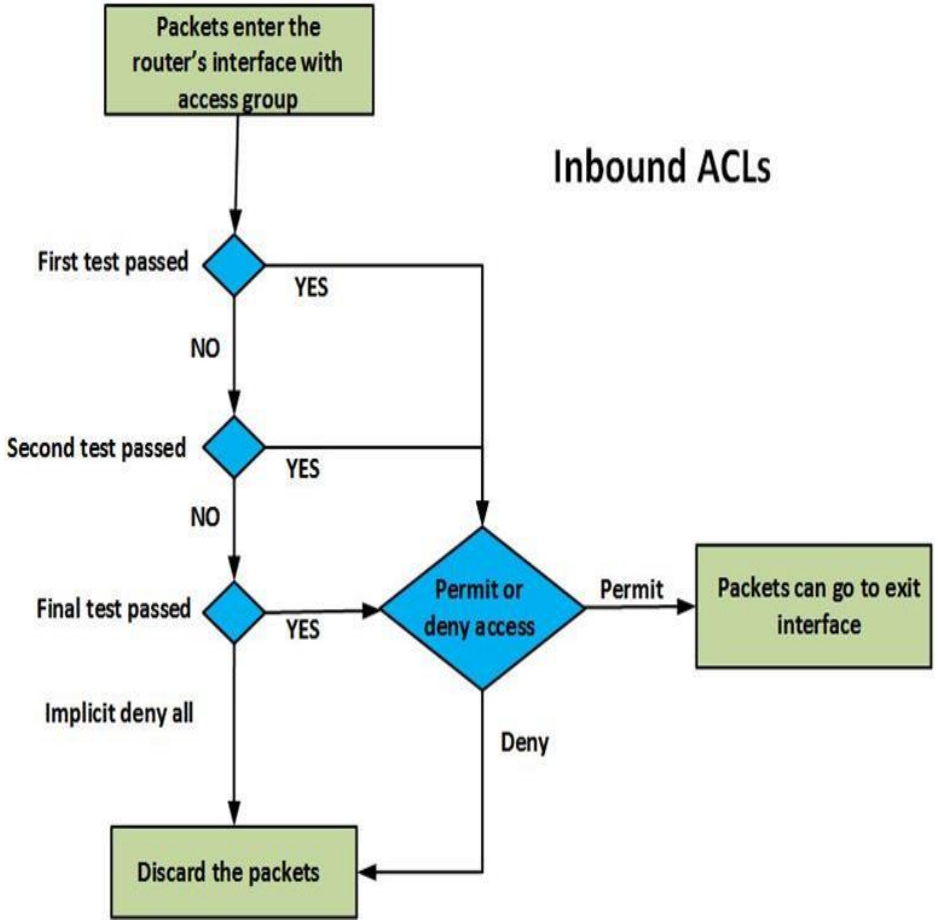
- **Access control lists (ACLs)** can be used to filter trafic on Cisco devices:
- Access lists are a set of rules, organized in a rule table.
- Each rule or line in an access-list provides a condition, either
  - **permit**
  - or **deny**

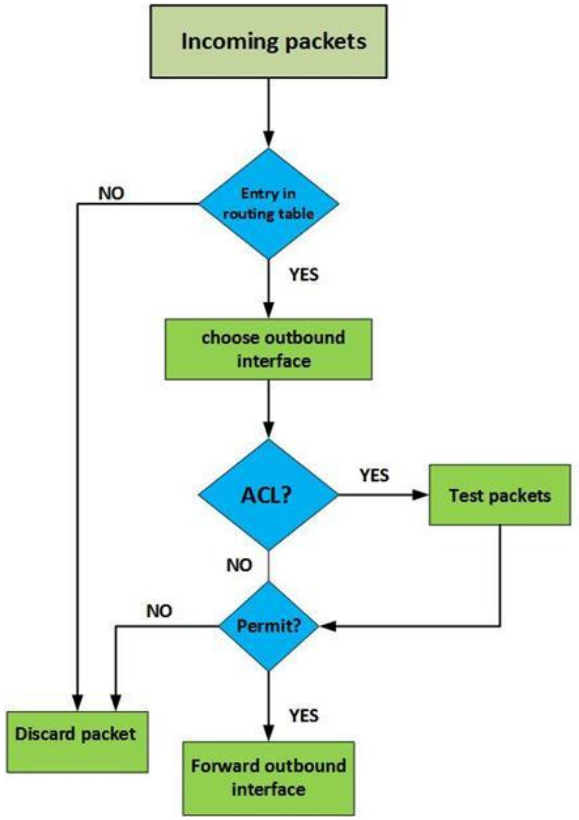


ACL Processing

- When filtering traffic, access lists are applied on interfaces.
- As a packet passes through a router, the top line of the rule list is checked first, and the router continues to go down the list until a match is made.
- Once a match is made, the packet is either permitted or denied.
- There is an implicit ‘deny all’ at the end of all access lists.
- You don’t create it, and you can’t delete it. Thus, access lists that contain **only deny statements** will **prevent all traffic**.

ACL Processing



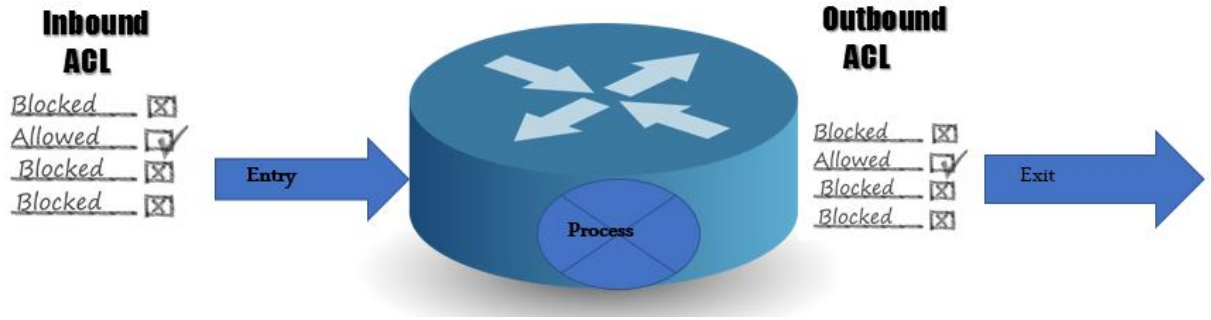


Outbound ACL

ACL Processing

- Access lists are applied either
  - inbound (packets received on an interface, before routing),
  - or outbound (packets leaving an interface, *after* routing).

- Only one access list **per interface**, **per protocol**, **per direction** is allowed.



Types of Access Lists

There are two categories of access lists: **numbered** and **named**.

- **Numbered** access lists are broken down into several ranges, each dedicated to a specific protocol:

1–99	IP standard access list
100-199	IP extended access list
200-299	Protocol type-code access list
300-399	DECnet access list
400-499	XNS standard access list
500-599	XNS extended access list
600-699	Appletalk access list
700-799	48-bit MAC address access list
800-899	IPX standard access list
900-999	IPX extended access list
1000-1099	IPX SAP access list
1100-1199	Extended 48-bit MAC address access list
1200-1299	IPX summary address access list
1300-1999	IP standard access list (expanded range)
2000-2699	IP extended access list (expanded range

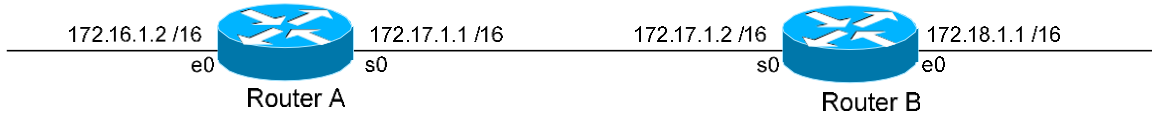
- **Named** access lists provide a bit more flexibility. Descriptive names can be used to identify your access-lists.

Types of Access Lists

- individual lines *cannot* be removed from a **numbered access list**. The entire access list must be deleted and recreated.
- All new entries to a numbered access list are added to the bottom.
- individual lines *can* be removed from a **named access-list**.
- However, like numbered lists, all new entries are still added to the bottom of the access list.
- There are two common types of named access lists:
  - IP standard named access lists
  - IP extended named access lists

Standard IP Access List

- access-list *[1-99] [permit | deny] [source address] [wildcard mask] [log]*
- Standard IP access-lists are based upon the source host or network IP address, and should be placed closest to the destination network.
- Consider the following example:



- In order to block network 172.18.0.0 from accessing the 172.16.0.0 network, we would create the following access-list on Router A:

Router(config)# *access-list 10 deny 172.18.0.0 0.0.255.255*
Router(config)# *access-list 10 permit any*

Standard IP Access List

- Notice the wildcard mask of 0.0.255.255 on the first line.
- This will match (*deny*) all hosts on the 172.18.x.x network.
- The second line uses a keyword of *any*, which will match (*permit)* any other address.
- Remember that you must have at least one permit statement in your access list.
- To apply this access list, we would configure the following on Router A:

```
Router(config)# int s0
Router(config-if)# ip access-group 10 in
```

Standard IP Access List

- To view all IP access lists configured on the router:

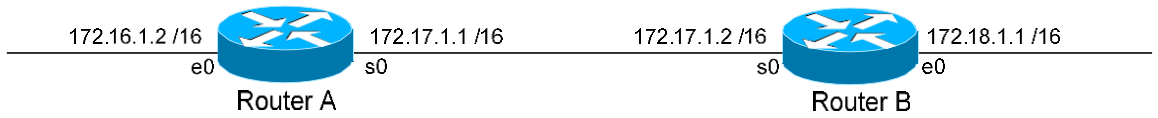
Router# *show ip access-list*

- To view what interface an access-list is configured on:

Router# *show ip interface*
Router# *show running-config*

Extended IP Access List

- access-list *[wildcard mask] [destination address] [wildc[100-199] [permit | deny] [protocol] [source addressard mask] [operator [port]] [log]*
- Extended IP access-lists block based upon the source IP address, destination IP address, and TCP or UDP port number.
- Extended access-lists should be placed closest to the source network.
- Consider the following example:



Extended IP Access List

- Assume there is a webserver on the 172.16.x.x network with an IP address of 172.16.10.10.
- In order to block network 172.18.0.0 from accessing anything on the 172.16.0.0 network, EXCEPT for the HTTP port on the web server, we would create the following access-list on Router B:

Router(config)# *access-list 101 permit tcp 172.18.0.0 0.0.255.255 host 172.16.10.10 eq 80*

Router(config)# *access-list 101 deny ip 172.18.0.0 0.0.255.255 172.16.0.0 0.0.255.255*

Router(config)# *access-list 101 permit ip any any*

Extended IP Access List

- The first line allows the 172.18.x.x network access only to port 80 on the web server.
- The second line blocks 172.18.x.x from accessing anything else on the 172.16.x.x network.
- The third line allows 172.18.x.x access to anything else.
- To apply this access list, we would configure the following on Router B:

```
Router(config)# int e0
Router(config-if)# ip access-group 101 in
```

Named Access Lists

Named access lists provide us with two advantages over numbered access lists.

- we can apply an identifiable name to an access list, for documentation purposes.
- we can remove individual lines in a named access-list, which is not possible with numbered access lists.

- Though we can *remove* individual lines in a named access list, we cannot *insert* individual lines into that named access list.
- New entries are always placed at the bottom of a named access list.

- To create a standard named access list, the syntax would be as follows:

```
Router(config)# ip access-list standard NAME
Router(config-std-nacl)# deny 172.18.0.0 0.0.255.255
Router(config-std-nacl)# permit any
```

- To create an extended named access list, the syntax would be as follows:

```
Router(config)# ip access-list extended NAME
Router(config-ext-nacl)# permit tcp 172.18.0.0 0.0.255.255 host 172.16.10.10 eq 80
Router(config-ext-nacl)# deny ip 172.18.0.0 0.0.255.255 172.16.0.0 0.0.255.255
Router(config-ext-nacl)# permit ip any any
```

- Notice that the actual configuration of the named access-list is performed in a separate router “mode”:

```
Router(config-std-nacl)#

Router(config-ext-nacl)#
```

Tutorial Works

- You need to block all traffic from the network 192.168.5.0/24 while allowing all other traffic.
- Task:
  - Write a standard numbered ACL to achieve this.
  - Apply the ACL to the inbound direction of interface FastEthernet0/0.

A web server (10.0.0.10) must only allow HTTP (port 80) traffic from the 172.16.0.0/16 network.
All other traffic to the server should be blocked.

- Write an extended numbered ACL to permit HTTP traffic from 172.16.0.0/16 to the server and deny all other traffic to it.
- Apply the ACL to the outbound direction of interface GigabitEthernet0/1.

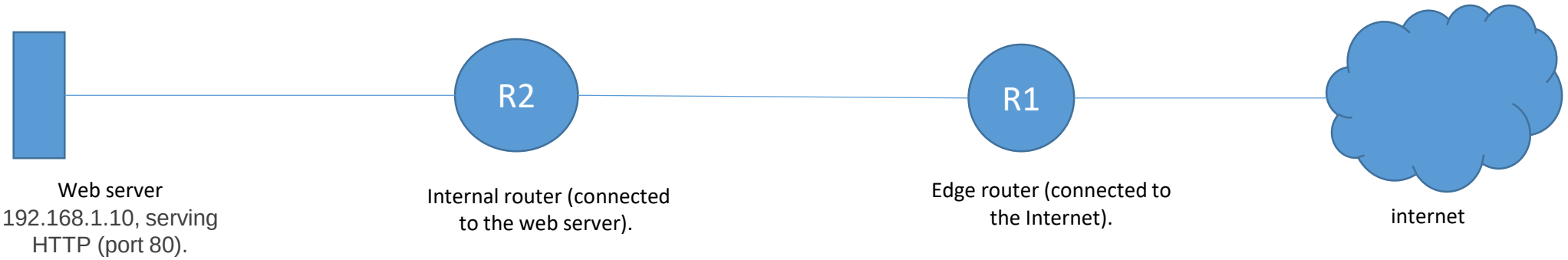
You need to deny the following IP addresses in a single ACL line:

- 192.168.10.1
- 192.168.10.2
- 192.168.10.3

- Calculate the correct wildcard mask to match these addresses.
- Write the ACL line using the calculated wildcard mask.

Lab: Configuring ACLs on Routers to Protect a Web Server

- **Objective**:
- Configure two routers to filter traffic between the Internet and an internal network hosting a web server.
- Allow HTTP traffic to the web server but block ICMP echo requests (pings) from the Internet while permitting other ICMP messages.



Assign IP addresses to interfaces and Configure Routing

- Basic Router Configurations
- Assign IP addresses to interfaces:
- R1:
  - Gig0/0 (Internet-facing): 203.0.113.1/24
  - Gig0/1 (to R2): 10.0.0.1/30

- R2:
  - Gig0/0 (to R1): 10.0.0.2/30
  - Gig0/1 (to Web Server): 192.168.1.1/24

- Enable dynamic or static routing between R1 and R2.

Configure ACLs on R1 (Edge Router)

- **Goal**: Block ICMP echo requests (pings) from the Internet but allow other ICMP messages (e.g., unreachable, TTL exceeded).

```
R1(config)# access-list 101 deny icmp any any echo
R1(config)# access-list 101 permit icmp any any
R1(config)# access-list 101 permit tcp any host 192.168.1.10 eq 80
R1(config)# access-list 101 permit ip any any
```

Apply ACL inbound on Internet-facing interface:

```
R1(config)# interface Gig0/0
R1(config-if)# ip access-group 101 in
```

Verify the Configuration

- Test ICMP from the Internet:
  - Ping to 203.0.113.1 (R1’s Internet interface) → Should fail.
  - Test other ICMP (e.g., traceroute) → Should work.
- Test HTTP Access:
  - From the Internet, try accessing http://192.168.1.10 → Should succeed.

Challenges

- **Add NAT on R1** to translate the web server’s private IP (192.168.1.10) to a public IP.
- Add **Timed ACL rules**
- **Modify the ACL** to log denied packets for monitoring.