Configuration and Management of Networks

Pedro Amaral
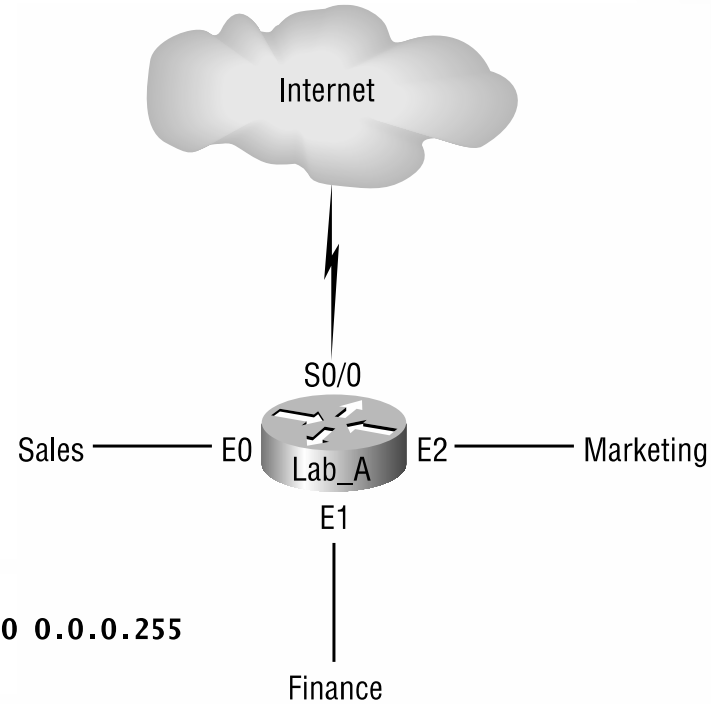
Access Control Lists

Sales 176.16.40.0/24

```
Lab_A#config t
Lab_A(config)#access-list 10 deny 172.16.40.0 0.0.0.255
Lab_A(config)#access-list 10 permit any


Lab_A(config)#int e1
Lab_A(config-if)#ip access-group 10 out
```
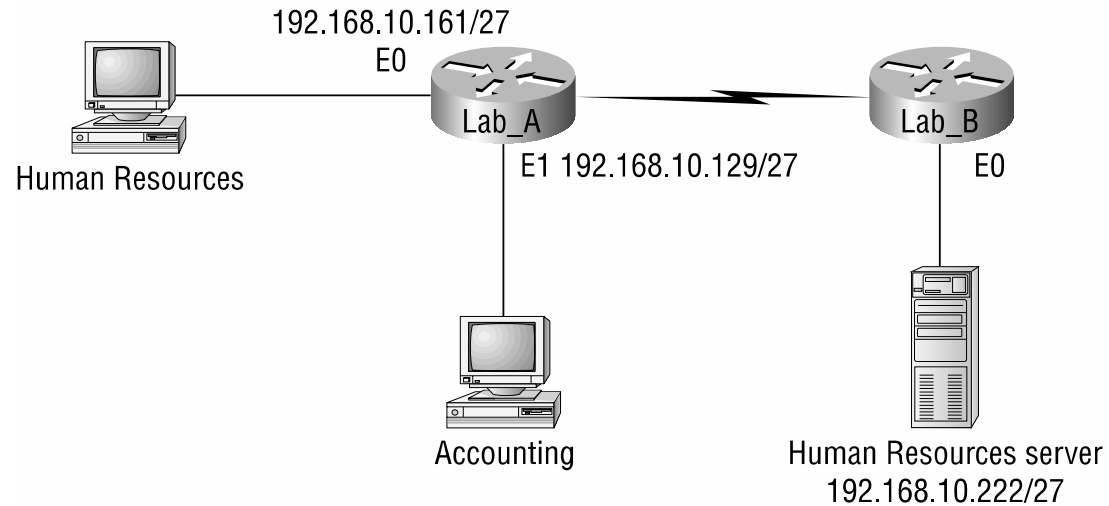
Sales cannot have access to finance

Access Control Lists

192.168.10.161/27
E0

Lab_A

Lab_B

Human Resources

E1 192.168.10.129/27

E0

Accounting

Human Resources server
192.168.10.222/27

```
Lab_B#config t
Lab_B(config)#access-list 10 deny 192.168.10.128 0.0.0.31
Lab_B(config)#access-list 10 permit any
Lab_B(config)#interface Ethernet 0
Lab_B(config-if)#ip access-group 10 out
```

Blocks accounting in access to HR server
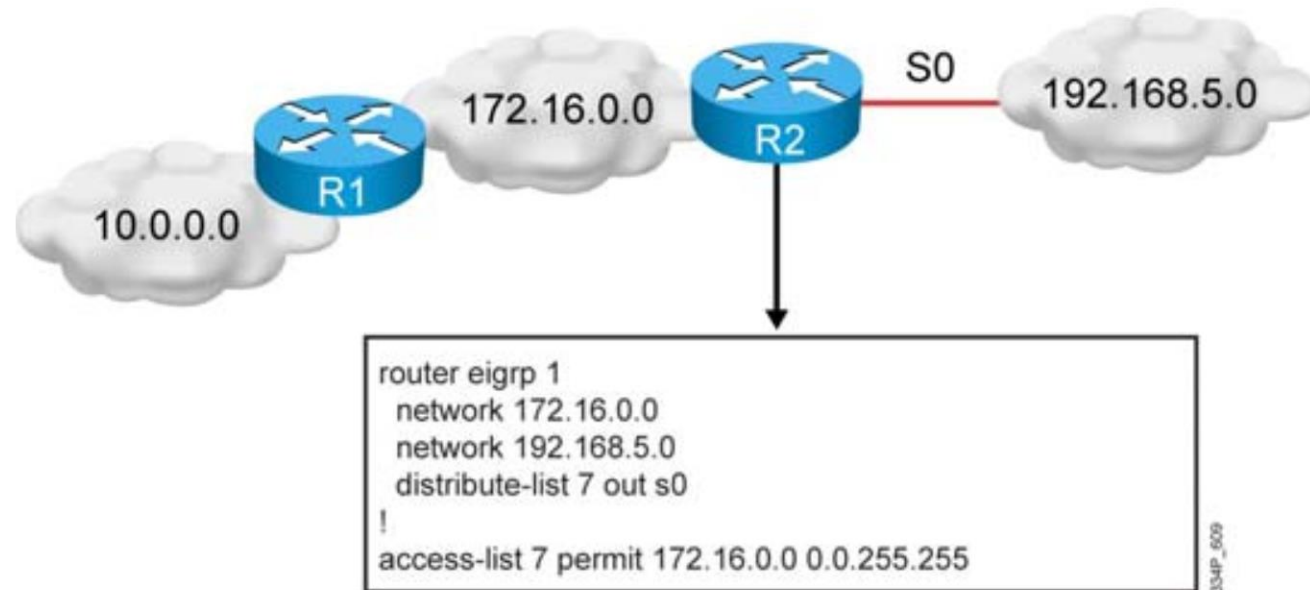
Access Control Lists

```
Lab_A(config)#access-list 50 permit 172.16.10.3
Lab_A(config)#line vty 0 4
Lab_A(config-line)#access-class 50 in
```

Only host 172.16.10.3 can telnet to router

Filtering Distribute List Filters

■ Hides network 10.0.0.0 using interface filtering

Filtering IP Prefix Filters

- Traditionally, IP prefix filters were implemented with IP access lists configured with the **distribute-list** command.

- Prefix lists:

    - Better performance than access lists

    - User-friendly command-line interface

    - Match routes in part of an address space with a subnet mask longer or shorter than a set number

Filtering IP Prefix Filters

- Filter by exact prefix length
  - mask filtering "/"
- Filter within a range
  - using **ge**
  - using **le**
  - using **ge** and **le**
- The matching process also considers the subnet mask

Filtering IP Prefix Filters without ge or le

- Similar to IP access lists with no wildcard bits

`R2(config)#`

```
ip prefix-list MyMatchList permit 192.168.0.0/16
```

- Which prefixes are matched?
  - 192.168.0.0/16: Match
  - 192.168.0.0/20: No Match
  - 192.168.2.0/24: No Match

Filtering IP Prefix Filters with ge or le

- A prefix list entry with the **ge** or **le** keyword matches any prefix within a specified address space for which the subnet mask falls within the specified limits.

`R2(config)#`

```
ip prefix-list List1 permit 192.168.0.0/16 le 20
ip prefix-list List2 permit 192.168.0.0/16 ge 18
```

- Which prefixes are matched?
  - 192.168.0.0/16, List1: Match
  - 192.168.0.0/16, List2: No Match

Filtering – Route Maps

- Route maps are similar to a scripting language for these reasons:
  - They work like access lists, but are more sophisticated.
    - They offer top-down processing.
    - When one of them finds a match, it stops searching.
  - Lines are sequence-numbered for easier editing.
    - Insertion of lines
    - Deletion of lines
  - Route maps are named, rather than numbered, for easier documentation.
  - Match criteria and set criteria can be used; similar to the if-then logic in in scripting languages.

Filtering – Route Maps

- A list of statements constitutes a route map.
- The list is processed in a top-down manner, like access lists are.
- The first match found for a route is applied.
- The sequence number is used for inserting or deleting specific route map statements.

```
route-map MyRouteMap permit 10
      { match statements }
      { match statements }
      { set statements }
      { set statements }
route-map MyRouteMap deny 20
      ::      ::      ::
      ::      ::      ::
route-map MyRouteMap permit 30
      ::      ::      ::
      ::      ::      ::
```

Filtering – Route Maps

- The match statement may contain multiple references.
- Multiple match criteria in the same line: logical OR.
- At least one reference must permit the route for it to be a candidate for redistribution.

```
route-map MyRouteMap permit 10
  match ip address ACL1 ACL2 ALC3
                    ───────────────►
                    Logical OR
```

```
route-map MyRouteMap deny 20
  match ACL1
  match interface fastethernet0/0
  match metric 3
Logical AND
```

- Multiple match statements on separate lines: logical AND.
- All match statements must permit the route for it to remain a candidate for redistribution.
- Route map permit or deny statements determine if the candidate will be redistributed.

Filtering – Route Maps

`R1(config)#`

```
route-map MyRouteMap permit 10
```

- Defines the route map with the name MyRouteMap

`R1(config-route-map)#`

```
match ip address prefix-list MyList
```

- Matches based on the prefix list "MyList" when defining the conditions to match.

`R1(config-route-map)#`

```
set interface ethernet 0
```

- Defines that interface ethernet 0 be used to forward packets that pass a match clause.

Configuration and Management of Networks

Route Maps – BGP example

```
R1#

router bgp 65010
neighbor 2.2.2.2 remote-as 65010
neighbor 3.3.3.3 remote-as 65010
neighbor 2.2.2.2 update-source loopback0
neighbor 3.3.3.3 update-source loopback0
neighbor 192.168.28.1 remote-as 65020
neighbor 192.168.28.1 route-map med_65020 out
!
access-list 66 permit 192.168.25.0.0 0.0.0.255
access-list 66 permit 192.168.26.0.0 0.0.0.255
!
route-map med_65020 permit 10
match ip address 66
set metric 100
!
route-map med_65020 permit 100
set metric 200
```