- Compiled by Dinesh Ghemosu

NIST College
Banpea
BScCSIT

Computer Network (CSC258)

<u>LAB 7</u>

Firewall Implementation, Router Access Control List (ACL)

Objective:
1. To understand the router firewall: Access Control Lists (ACLs)
2. To implement basic router security.

Aparatus: Packet Tracer

Theory
<u>Access Control Lists (ACLs)</u>
Background:
Access control lists (ACLs) are an integral part of Cisco's security solution. The proper use and configuration of access lists is a vital part of router configuration because access lists are such versatile networking accessories. Contributing mightily to the efficiency and operation of your network, access lists give network managers a huge amount of control over traffic flow throughout the enterprise. With access lists, we can gather basic statistics on packet flow and security policies can be implemented.

An *access list* is essentially a list of conditions that categories packets that are used to filter unwanted packets when implementing security policies. With the right combination of access lists, network managers arm themselves with the power to enforce nearly any security policy they can invent.

Creating access lists is really a lot like programming a series of if-then statements—if a given condition is met, then a given action is taken. If the specific condition isn't met, nothing happens and the next statement is evaluated. Access-list statements are basically packet filters that packets are compared against, categorized by, and acted upon accordingly. Once the lists are built, they can be applied to either inbound or outbound traffic on any interface. Applying an access list causes the router to analyze every packet crossing that interface in the specified direction and take the appropriate action.

Packet filtering at the network level can be achieved by applying the Access Control Lists (ACLs) at the router called router firewall. ACLs at the router filter the inbound traffic while it permit or deny packets based on source IP/network and destination IP/network, IP, TCP,UDP protocol information. Generally we use the ACLs to provide a basic level of security for accessing our network. Access lists can allow one host to access a part of network and prevent another host from accessing the same area.

Types of Access lists:
There are two types of access lists:

1. **Standard access lists**: These ACLs use only the source IP address in an IP packet as the condition test. All decisions are made based on the source IP address. This means that standard access lists basically permit or deny an entire suite of protocols. They don't distinguish between any of the many types of IP traffic such as Web, Telnet, UDP, and so on. You can create a standard IP access list by using the access-list numbers 1-99 or 1300-1999 (expanded range) or string.
2. **Extended access lists**: Extended access lists can evaluate many of the other fields in the layer 3 and layer 4 headers of an IP packet. They can evaluate source and destination IP addresses, the Protocol field in the Network layer header, and the port number at the Transport layer header. This gives extended access lists the ability to make much more granular decisions when controlling traffic. Valid Extended ACL ID range is: 100 - 199 or a string.

To use an access list as a packet filter, you need to apply it to an interface on the router where you want the traffic filtered. And you've got to specify which direction of traffic you want the access list applied to. That is, when creating an access list, we define criteria that are applied to each packet that is processed by the router; the router decides whether to forward or block each packet on the basis of whether or not the packet matches the criteria.

Some general guidelines for creating and implementing access-lists on a router:

- You can assign only one access list per interface per protocol per direction.
- Organize your access lists so that the more specific tests are at the top.
- Anytime a new entry is added to the list, it will be placed at the bottom of the list, which is why, text editor is recommended to use for access lists.
- You can't remove one line from an access list. If you try to do this, you will remove the entire list.
- Unless your access list ends with a permit any command, all packets will be discarded if they do not meet any of the list's tests. This means every list should have at least one permit statement or it will deny all traffic.
- Create access lists and then apply them to an interface.
- Access lists are designed to filter traffic going through the router. They will not filter traffic that has originated from the router.

To configure Standard Access List

Router(config)#access-list 10 deny host 172.16.30.2
This tell the list to deny any packets from host 172.16.30.2.

Wildcard Masking
Wildcards are used with access lists to specify/filter an individual host, a network, or a specific range of network or networks to filter.
To specify a host, the address would look like this:

    172.16.30.5 0.0.0.0

The four zeros represent each octet of the address. Whenever a zero is present, it indicates that the octet in the address must match the corresponding reference octet exactly.
To specify a range of network of /24 subnet, the wildcard is specify as:

    172.16.30.0 0.0.0.255

This tells the router to match up the first three octets exactly, but the forth octet can be any value.

If you want to block access to the part of the network that ranges from 172.16.8.0 through 172.16.15.0, you would use the block size of 8, and your network number would be 172.16.8.0, and the wildcard mask would be 0.0.7.255. Wildcard is always on number less than the block size.

<u>A few examples</u>:
Router(config)#**access-list 10 deny 172.16.10.0 0.0.0.255**
This tell a router to match the first three octets exactly but that the fourth octet can be anything.

Router(config)#**access-list 10 deny 172.16.0.0 0.0.255.255**
This tell the router to match the first two octets and that the last two octets can be any values.

Router(config)#**access-list 10 deny 172.16.16.0 0.0.3.255**
This configuration tells the router to start at network 172.16.16.0 and use a block size of 4. The range would then be 172.16.16.0 through 172.16.19.255.

Router(config)#**access-list 10 deny 172.16.32.0 0.0.15.255**
This example reveals an access list starting at 172.16.32.0 going up to block size of 16 to 172.16.47.255.

<u>Standard Access List Example</u>

<u>Example 1:</u>

Figure a router has three LAN connections and one WAN connection to the Internet. Users on the Sales LAN should not have access to the Finance LAN, but they should be able to access the Internet and the marketing department files. The Marketing LAN needs to access the Finance LAN for application services.

```
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
```

It’s very important to remember that the any command is the same thing as saying the following using wildcard masking:

```
Lab_A(config)#access-list 10 permit 0.0.0.0 255.255.255.255
```

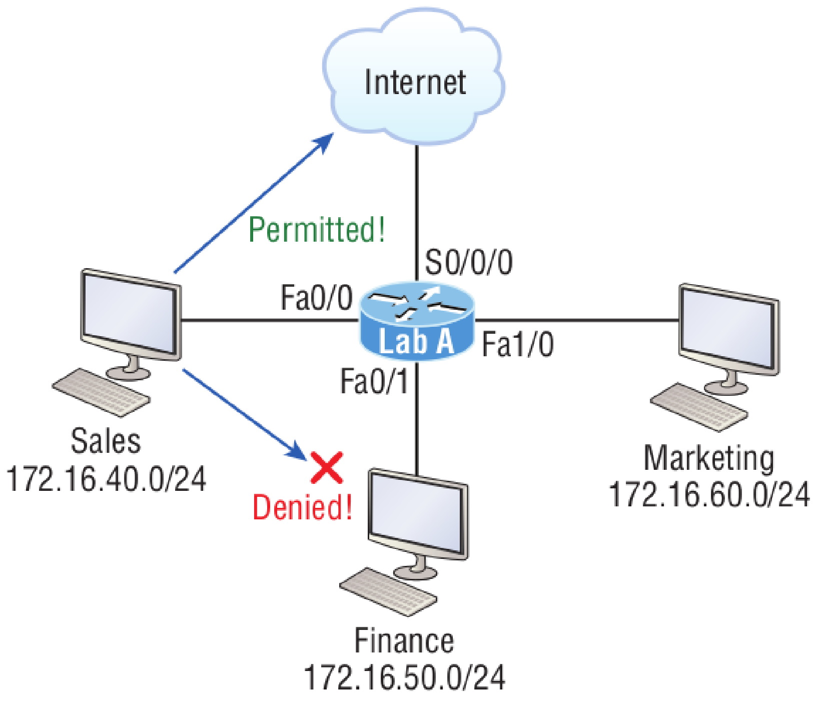


Figure 1: IP access list example with three LANs and one WAN connection.

But where should this access list be placed? If you place it as an incoming access list on Fa0/0, you might as well shut down the FastEthernet interface because all of the Sales LAN devices will be denied access to all networks attached to the router. The best place to apply this access list is on the Fa0/1 interface as an outbound list:

```
Lab_A(config)#int fa0/1
Lab_A(config-if)#ip access-group 10 out
```

Doing this completely stops traffic from 172.16.40.0 from getting out FastEtherent 0/1. Tt has no effect on the hosts from the Sales LAN accessing the Marketing LAN and the Internet because traffic to those destinations doesn’t go through interface Fa0/1. Any packet trying to exit out Fa0/1 will have to go through the access list first.

Example 2:

Stop the Accounting users from accessing the Human Resources server attached to the Lab_B router but allow all other users access to that LAN using a standard ACL. In this example, Ethernet 0 is the outbound interface on the Lab_B router and here's the access list that should be placed on it:

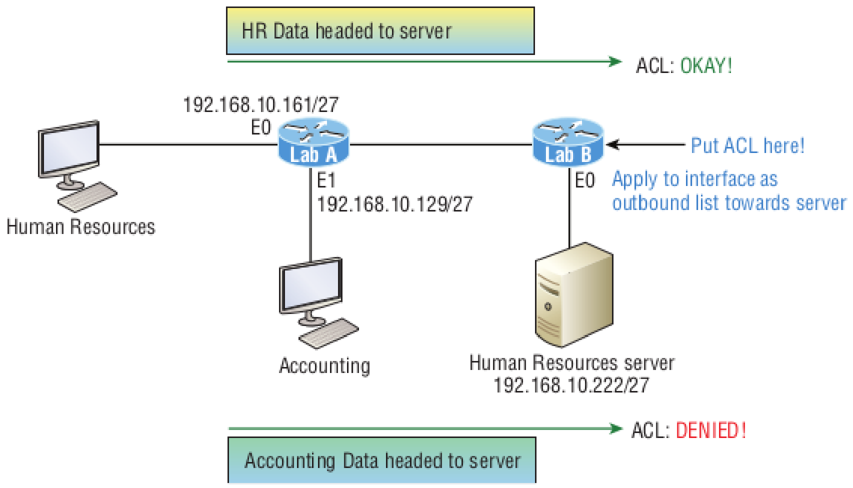


Figure 2: IP standard access list example 2

```
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
```

The accounting subnet is the 192.168.10.128/27, which is a 255.255.255.224, with a block size of 32 in the fourth octet.

Example 3:

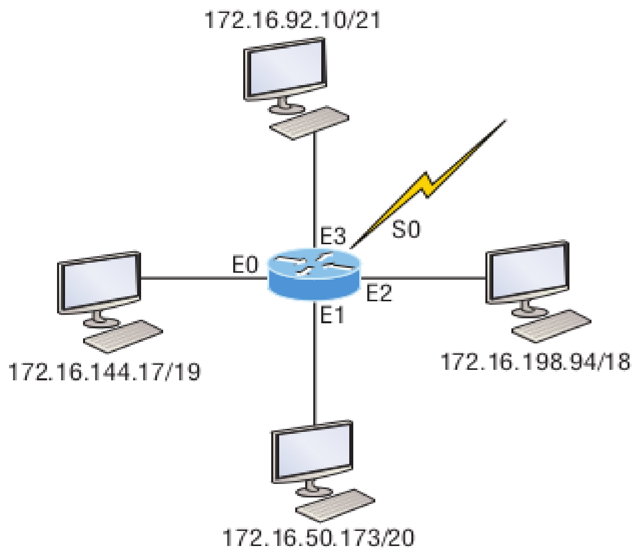


Figure 3: IP standard access list example 3.

You need to write an access list that will stop access from each of the four LANs shown in the diagram to the Internet. Each of the LANs reveals a single host's IP address, which you need to use to determine the subnet and wildcards of each LAN to configure the access list. Here is an example of what your answer should look like, beginning with the network on E0 and working through to E3:

```
Router(config)#access-list 1 deny 172.16.128.0 0.0.31.255
Router(config)#access-list 1 deny 172.16.48.0 0.0.15.255
Router(config)#access-list 1 deny 172.16.192.0 0.0.63.255
Router(config)#access-list 1 deny 172.16.88.0 0.0.7.255
Router(config)#access-list 1 permit any
Router(config)#interface serial 0
Router(config-if)#ip access-group 1 out
```

Controlling VTY (Telnet/SSH) Access
You need to do these two things to make this happen:

1. Create a standard IP access list that permits only the host or hosts you want to be able to telnet into the routers.
2. Apply the access list to the VTY line with the access-class in command.

Here, I'm allowing only host 172.16.10.3 to telnet into a router:

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

Because of the implied deny any at the end of the list, the ACL stops any host from telnetting into the router except the host 172.16.10.3, regardless of the individual IP address on the router being used as a target.

Configuring Extended Access lists

Using an extended access list allow us to specify source and destination addresses as well as the protocol and port number that identify the upper-layer protocol or application. An extended ACL is just what we need to affectively allow users access to a physical LAN while denying them access to specific hosts- - even specific services on those hosts!
The extended access-list range from 100 to 199. The 2000-2699 range is also available for extended IP access lists.

Router(config)#**access-list 110 deny tcp any host 172.16.30.2 eq 23 log**
This line says to deny any source host trying to telnet to destination host 172.16.30.2. The log command is used to log messages every time the access list entry is hit; it is used to monitor inappropriate access attempts.

Router(config)# access-list 110 permit ip any any
It will permit the IP stack. If TCP was used instead of IP in this line, then UDP, etc. would all be denied.

The command 0.0.0.0 255.255.255.255 is the same command as any. So above command also looks like:
Router(config)# access-list 110 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255

As always, once our access list is created, we must apply it to an interface with the same command used for the IP standard list:
Router(config-if)#**ip access-group 110 in**
Or this:
Router(config-if)#**ip access-group 110 out**

Example 1: Deny access to a host at 172.16.50.5 on the finance department LAN for both Telnet and FTP services. All other services on this and all other hosts are acceptable for the sales and marketing departments to access.

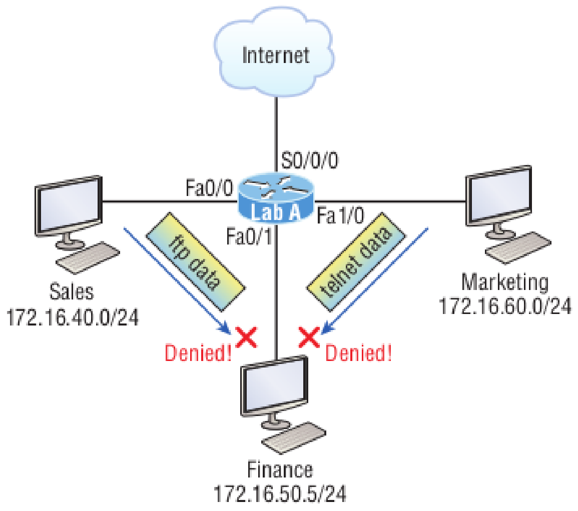


Figure 4: Extended ACL example 1

Here's the ACL we must create:
Lab_A#**config t**
Lab_A(config)#**access-list 110 deny tcp any host 172.16.50.5 eq 21**
Lab_A(config)#**access-list 110 deny tcp any host 172.16.50.5 eq 23**
Lab_A(config)#**access-list 110 permit ip any any**

The access-list 110 tells the router we're creating an extended IP ACL. The tcp is the protocol field in the Network layer header. If the list doesn't say tcp here, you cannot filter by TCP port numbers 21 and 23 as shown in the example. Remember that these values indicate FTP and Telnet, which both use TCP for connection-oriented services. The any command is the source, which means any source IP address, and the host is the destination IP address. This ACL says that all IP traffic will be permitted from any host except FTP and Telnet to host 172.16.50.5 from any source.

After the list is created, it must be applied to the FastEthernet 0/1 interface outbound because we want to block all traffic from getting to host 172.16.50.5 and performing FTP and Telnet. If this list was created to block access only from the Sales LAN to host 172.16.50.5, then we'd have put this list closer to the source, or on FastEthernet 0/0. In that situation, we'd apply the list to inbound traffic. This highlights the fact that you really need to analyze each situation carefully before creating and applying ACLs!
Okay—now let's go ahead and apply the list to interface Fa0/1 to block all outside FTP and Telnet access to the host 172.16.50.5:

```
Lab_A(config)#int fa0/1
Lab_A(config-if)#ip access-group 110 out
```

Example 2: Figure below has four LANs and a serial connection. We need to prevent Telnet access to the networks attached to the E1 and E2 interfaces.
The configuration on the router would look something like this, although the answer can vary:

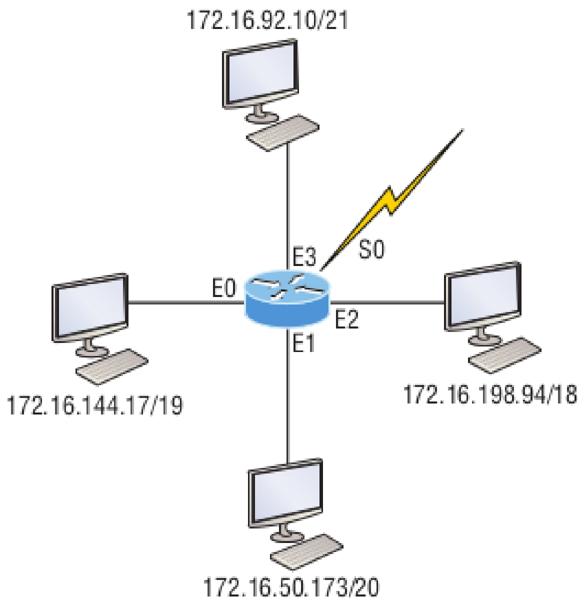


Figure 5: Extended ACL example 2.

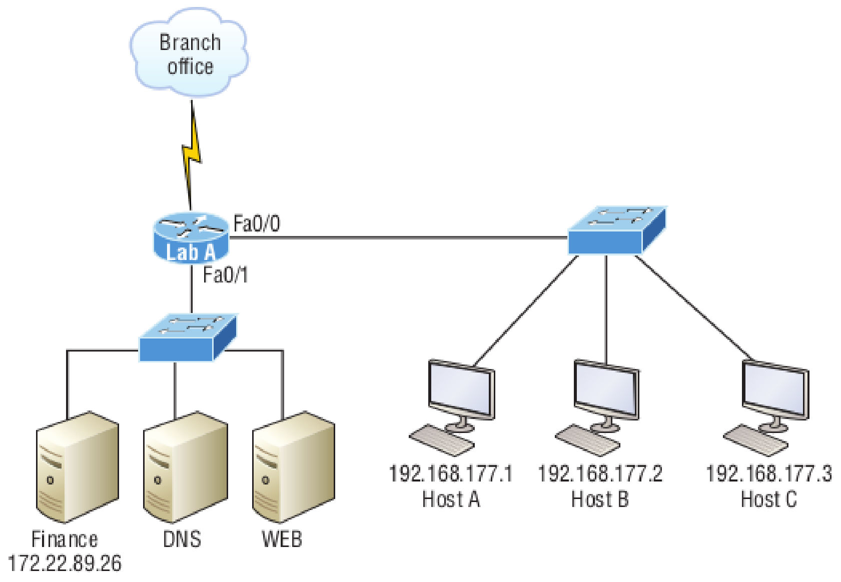
```
Router(config)#access-list 110 deny tcp any 172.16.48.0 0.0.15.255 eq 23
Router(config)#access-list 110 deny tcp any 172.16.192.0 0.0.63.255 eq 23
Router(config)#access-list 110 permit ip any any
Router(config)#interface Ethernet 1
Router(config-if)#ip access-group 110 out
Router(config-if)#interface Ethernet 2
Router(config-if)#ip access-group 110 out
```

Here are the key factors to understand from this list:

- First, you need to verify that the number range is correct for the type of access list you are creating. In this example, it's extended, so the range must be 100–199.
- Second, you must verify that the protocol field matches the upper-layer process or application, which in this case, is TCP port 23 (Telnet).
- Third, verify that the destination port number matches the application you're filtering for. In this case, port 23 matches Telnet, which is correct, but know that you can also type **telnet** at the end of the line instead of 23.
- Finally, the test statement permit ip any any is important to have there at the end of the list because it means to enable all packets other than Telnet packets destined for the LANs connected to Ethernet 1 and Ethernet 2.

Note:
The protocol parameter must be TCP since Telnet uses TCP. If it were TFTP instead, then the protocol parameter would have to be UDP because TFTP uses UDP at the Transport layer.

Example 3: allow HTTP access to the Finance server from source Host B only. All other traffic will be permitted.



```
Lab_A#config t
Lab_A(config)#access-list 110 permit tcp host 192.168.177.2 host 172.22.89.26 eq 80
Lab_A(config)#access-list 110 deny tcp any host 172.22.89.26 eq 80
Lab_A(config)#access-list 110 permit ip any any
Lab_A(config)#interface fastethernet 0/1
Lab_A(config-if)#ip access-group 110 out
```

First we need to permit Host B HTTP access to the Finance server. But since all other traffic must be allowed, we must detail who cannot HTTP to the Finance server, so the second test statement is there to deny anyone else from using HTTP on the Finance server. Finally, now that Host B can HTTP to the Finance server and everyone else can't, we'll permit all other traffic with our third test statement.
Our challenge required us to allow only HTTP traffic to the Finance server from Host B. If we apply the ACL inbound on Fa0/0, then the branch office would be able to access the Finance server and perform HTTP. So in this example, we need to place the ACL closest to the destination:

Named ACLs

Names access lists allow you to use names to both create and apply either standard or extended access lists.

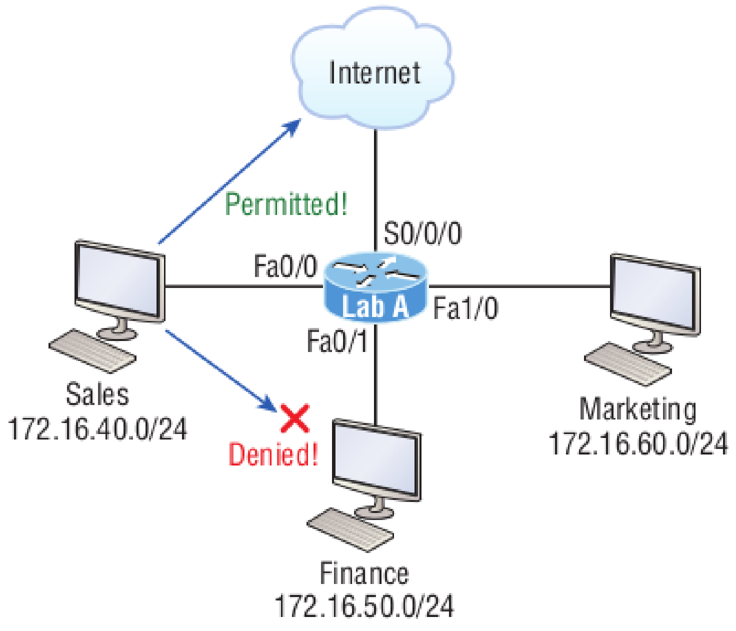


Figure 6: Names access list example.

```
Lab_A#config t
Lab_A(config)#ip access-list standard BlockSales
Lab_A(config-std-nacl)#
```

Here BlockSales is standard access list name.

```
Lab_A(config-std-nacl)#deny 172.16.40.0 0.0.0.255
Lab_A(config-std-nacl)#permit any
Lab_A(config-std-nacl)#exit

Lab_A(config)#^Z                    //ctrl+z
Lab_A#

Lab_A#sh running-config | begin ip access
ip access-list standard BlockSales
deny
172.16.40.0 0.0.0.255
permit any
!
```

And there it is: the BlockSales access list has truly been created and is in the running-config of the router. Next, apply the access list to the correct interface:

```
Lab_A#config t
Lab_A(config)#int fa0/1
Lab_A(config-if)#ip access-group BlockSales out
```

Commands used for verifying access-list configuration

```
show access list
show access-list 110
show ip access-list
show ip interface f0/0
show running-config | begin ip access
```