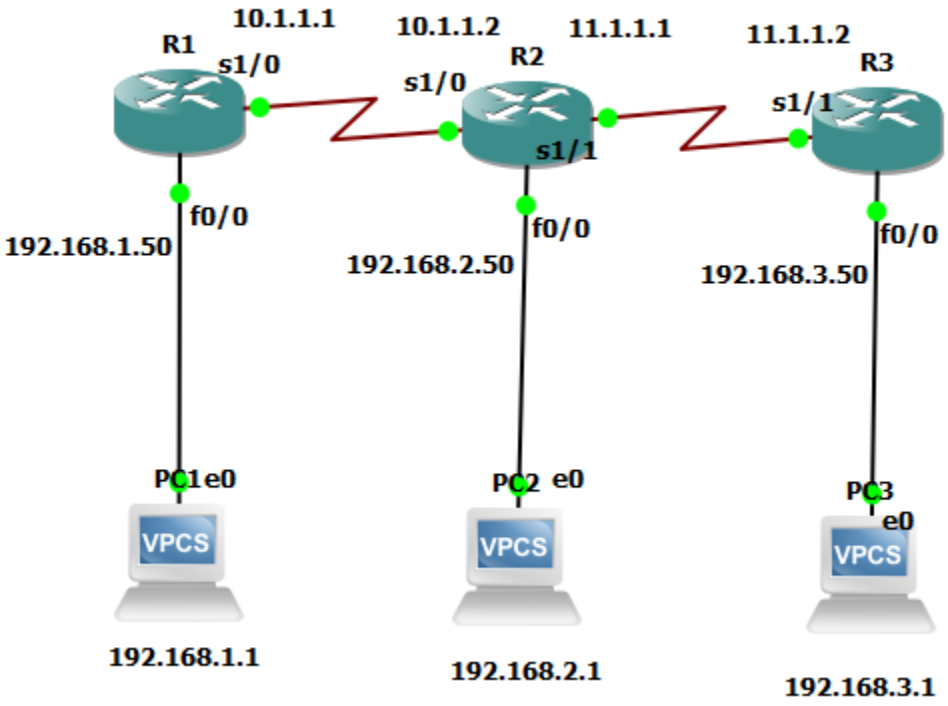
Practical No. 2

Aim: Implement IPv4 ACLs

- **Standard**
- **Extended**

Step 1: Design the Topology



Step 2: Configure the System.

PC 1:

PC 1 can be used to help send data to other PCs on the network via its router. In this case PC 1 is

connected to R1 via the fastEthernet 0/0 connection.

Here we set the Ip-address of the PC to '192.168.1.1/24' with a gateway of '192.168.1.50'.

```
PC1> ip 192.168.1.1/24 192.168.1.50
Checking for duplicate address...
PC1 : 192.168.1.1 255.255.255.0 gateway 192.168.1.50

PC1> show ip

NAME        : PC1[1]
IP/MASK     : 192.168.1.1/24
GATEWAY     : 192.168.1.50
DNS         :
MAC         : 00:50:79:66:68:00
LPORT       : 10024
RHOST:PORT  : 127.0.0.1:10025
MTU:        : 1500
```

Router 1:

Here we configure Router 1 (R1).

We set the ip address for the router and its various connections.

For the PC 1 connection we use the fastEthernet 0/0 interface and set its IP-address to

'192.168.1.50'

Then we configure the serial 1/0 interface and set its IP-address to 10.1.1.1 with 255.255.255.0 as its

subnet mask.

Note: Always remember to set the 'no shutdown' attribute for each interface. This way the protocol state will be changed to up rather than down.

Also after setting up the configuration, it is advised to ping a nearby device to check whether the

connection has been established. Ping 192.168.1.1

```
R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#interface fasthernet 0/0
                         ^
% Invalid input detected at '^' marker.

R1(config)#interface fastethernet 0/0
R1(config-if)#ip address 192.168.1.50 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#
*Mar  1 00:04:37.035: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Mar  1 00:04:38.035: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0,
R1(config-if)#exit
```

```
R1(config)#interface serial 0/0
R1(config-if)#ip address 10.1.1.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#
*Mar  1 00:05:57.203: %LINK-3-UPDOWN: Interface Serial0/0, changed state to up
*Mar  1 00:05:58.203: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0,
R1(config-if)#exit
R1(config)#
```

```
R1#
*Mar  1 00:12:53.267: %SYS-5-CONFIG_I: Configured from console by console
R1#ping 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.1, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 56/59/64 ms
R1#
```

Router 2:

Here we configure Router 2

It is connected to PC 2 via the fastEthernet 0/0 with its Ip-address set to '192.168.2.50' with a

subnet mask of '255.255.255.0'.

It is connected to R1 via the interface serial 1/0 with its Ip-address set to '10.1.1.2' with a subnet

mask of '255.255.255.0'.

It is connected to R2 via the interface serial 1/1 with its Ip-address set to '11.1.1.1' with a subnet

mask of '255.255.255.0'. Also, after setting up the configuration, it is advised to ping a nearby device to check whether the

connection has been established

```
R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#interface fastethernet 0/0
R2(config-if)#ip address 192.168.2.50 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
*Mar  1 00:04:39.259: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Mar  1 00:04:40.259: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, c
R2(config-if)#exit
R2(config)#interface serial 0/1
R2(config-if)#ip address 10.1.1.2 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#
*Mar  1 00:05:23.099: %LINK-3-UPDOWN: Interface Serial0/1, changed state to up
*Mar  1 00:05:24.099: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed
R2(config-if)#exit
```

PC 2:

PC 2 can be used to help send data to other PCs on the network via its router. In this case PC 2 is

connected to R2 via the fastEthernet 0/0 connection.

Here we set the Ip-address of the PC to '192.168.2.1/24' with a gateway of '192.168.2.50'.

```
PC2> ip 192.168.2.1/24 192.168.2.50
Checking for duplicate address...

PC1 : 192.168.2.1 255.255.255.0 gateway 192.168.2.50

PC2>
PC2> show ip

NAME        : PC2[1]
IP/MASK     : 192.168.2.1/24
GATEWAY     : 192.168.2.50
DNS         :
MAC         : 00:50:79:66:68:01
LPORT       : 10026
RHOST:PORT  : 127.0.0.1:10027
MTU:        : 1500
```

```
R2#
*Mar  1 00:11:18.139: %SYS-5-CONFIG_I: Configured from console by console
R2#ping 192.168.2.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.1, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 36/56/72 ms
R2#
```

Router 3:

Here we configure Router 3

It is connected to PC 3 via the fastEthernet 0/0 with its Ip-address set to '192.168.3.50' with a

subnet mask of '255.255.255.0'.

Then we configure the serial 1/0 interface and set its IP-address to 11.1.1.2 with 255.255.255.0 as its

subnet mask. Also, after setting up the configuration, it is advised to ping a nearby device to check whether the

connection has been established. Ping 192.168.2.1

```
R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)#interface fastethernet 0/0
R3(config-if)#
R3(config-if)#ip address 192.168.3.50 255.255.255.0
R3(config-if)#no shutdown
R3(config-if)#
*Mar  1 00:06:38.967: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Mar  1 00:06:39.967: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0,
R3(config-if)#exit
R3(config)#interface serial 1/1
R3(config-if)#ip address 11.1.1.1 255.255.255.0
R3(config-if)#no shutdown
R3(config-if)#
*Mar  1 00:07:07.359: %LINK-3-UPDOWN: Interface Serial1/1, changed state to up
*Mar  1 00:07:08.359: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/1, chang
R3(config-if)#exit
```

PC 3:

PC 3 can be used to help send data to other PCs on the network via its router. In this case PC 3 is

connected to R3 via the fastEthernet 0/0 connection.

Here we set the Ip-address of the PC to '192.168.3.1/24' with a gateway of '192.168.3.50'.

```
PC3> ip 192.168.3.1/24 192.168.3.50
Checking for duplicate address...
PC1 : 192.168.3.1 255.255.255.0 gateway 192.168.3.50

PC3> show ipshow ip
Invalid arguments

PC3> show ip

NAME        : PC3[1]
IP/MASK     : 192.168.3.1/24
GATEWAY     : 192.168.3.50
DNS         :
MAC         : 00:50:79:66:68:02
LPORT       : 10028
RHOST:PORT  : 127.0.0.1:10029
MTU:        : 1500
```

```
R3#
*Mar  1 00:10:03.931: %SYS-5-CONFIG_I: Configured from console by console
R3#ping 192.168.3.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.3.1, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 56/63/68 ms
R3#
```

Step 3: Explain what ACL is and how we apply it in the current system.

- Access-list (ACL) is a set of rules defined for controlling network traffic and reducing network attacks. ACLs are used to filter traffic based on the set of rules defined for the incoming or outgoing of the network.
- Layer 3 Security – Router blocks the IP address, which means that ACL is applied.
- ACL is also called as packet filtering firewall

There are two main types of ACL:

- Standard:
  - Standard access-list is implemented using source IP address only.
  - Standard Access-list is generally applied close to destination (but not always).
  - Standard access-list uses the range 1-99 and extended range 1300-1999.
  - In a standard access list, the whole network or sub-network is denied.
- Extended:
  - In the Extended access list, packet filtering takes place on the basis of source IP address, destination IP address, port numbers.
  - Extended access-list is generally applied close to the source but not always.

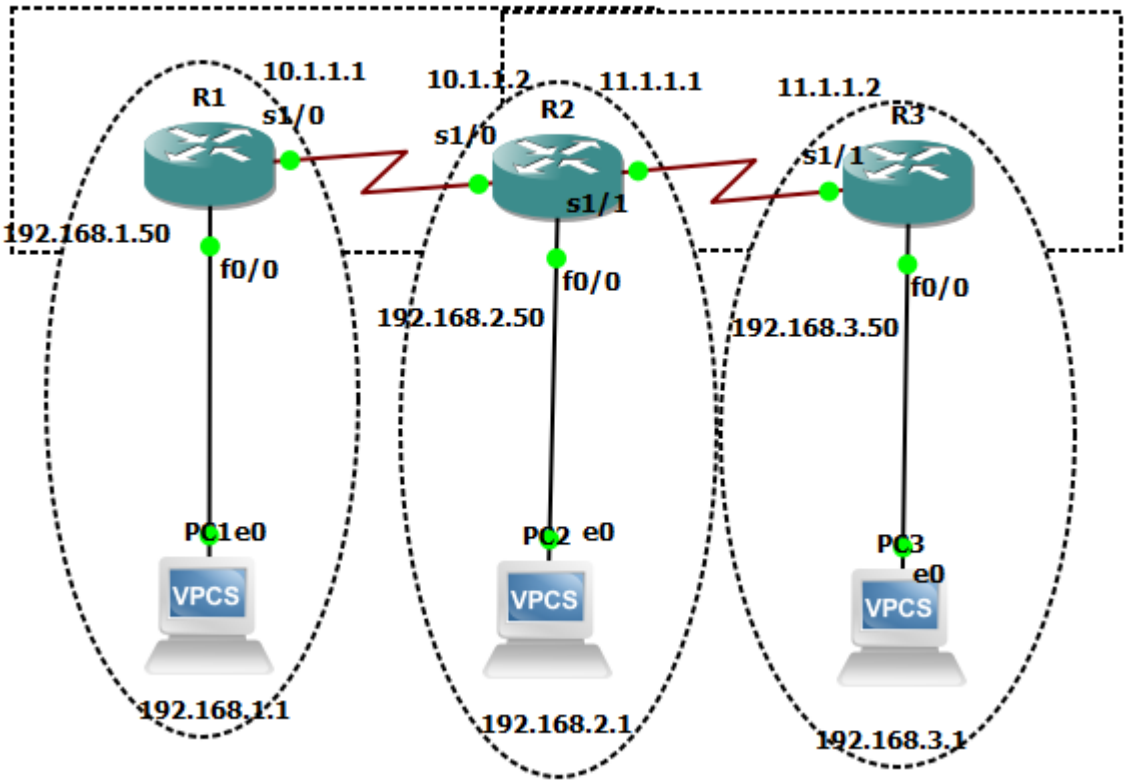
- Extended ACL is created from 100 – 199 & extended range 2000 – 2699.
- In an extended access list, particular services will be permitted or denied.

To enable ACL on our network we use Routing protocols.

Note: As the destination is not reachable from PC 1 to Router 2 (means via connection is not reachable) so we apply a routing protocol.

There are many routing protocols:

- Static
- Dynamic:
  - RIP, OSPF, EIGRP.
  - Here we apply RIP. After applying RIP all the Routers and PCs are able to communicate and ping each other.



Extended:

On the LAN_A router

access-list 110 deny ip 192.168.11.0 0.0.0.255 192.168.12.0 0.0.0.255

access-list 110 permit ip any any

int e0

ip access-group 110 in

OR

Standard:

On the LAN B router

access-list 10 deny 192.168.11.0 0.0.0.255

access-list 10 permit any

int e0

ip access-group 10 out

Step 4: Configure and test the ACL on the network system.

• To initially apply the routing protocol for the system:

To configure the network route for the Router R1, we use the RIP protocol. Using

router rip – version 2 we set the network route PC 1 and Router 1.

```
R1#config
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#router rip
R1(config-router)#version 2
R1(config-router)#network 192.168.1.0
R1(config-router)#network 10.0.0.0
R1(config-router)#exit
R1(config)#exit
R1#
*Mar  1 00:23:02.435: %SYS-5-CONFIG_I: Configured from console by console
R1#
```

To configure the network route for the Router 2

we use the RIP protocol. Using router rip – version 2 we set the network route PC 2 and Router 2. This includes the two serial connections (serial 1/0, serial 1/1) and the fastEthernet 0/0 connection.

```
R2#config
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#router rip
R2(config-router)#version 2version 2
 Wrong Rip version, only version 1 or version 2 is valid
R2(config-router)#version 2
R2(config-router)#network 192.168.2.0
R2(config-router)#network 10.0.0.0
R2(config-router)#network 11.0.0.0
R2(config-router)#exit
R2(config)#
R2(config)#exit
R2#
```

To configure the network route for the Router 3

we use the RIP protocol. Using router rip – version 2 we set the network route PC 3 and Router 3. This includes the serial connection serial 1/0 and the fast Ethernet 0/0 connection.

To check whether the routing has been done and the connection has been made, we use a simple ping. If it returns a successful message, we can assume that the routing of the network has been done for all the routers.

```
R1#config
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#network 192.168.3.0
                  ^
% Invalid input detected at '^' marker.

R1(config)#router rip
R1(config-router)#version 2
R1(config-router)#network 192.168.3.0
R1(config-router)#network 11.0.0.0
R1(config-router)#do ping 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 40/57/92 ms
R1(config-router)#
```

- **To apply Standard ACL:**

Here the scenario we are going to use is that whatever data that is being sent by PC 1 should not be received by PC 3, so based on the features of a standard ASL, we know that it needs to be place near the destination. So, by checking the network topology we can say that the best place to create a standard ASL is in Router 3. We should also note that the number of the standard ASL should be in the range of 1-99, in our case we have chosen 10. So, by configuring the access list to 'deny', we can deny all communication from PC 1 to

PC 2.

```
R1(config)#access-list 10 deny host 192.168.1.1
R1(config)#exit
R1#
*Mar  1 00:35:13.859: %SYS-5-CONFIG_I: Configured from console by console
R1#show access list
% Ambiguous command:  "show access list"
R1#show access-list
Standard IP access list 10
    10 deny   192.168.1.1
```

Note: The 'deny' keyword tends to deny access to the entire network, so we have to configure the

access list for the serial interface 1/0 to allow data to be received from other devices other than PC

1. To achieve this, we use the 'permit' keyword in the access-list configuration.

```
R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#access-list 10 permit any
R1(config)#interface se1/0
R1(config-if)#ip access-group 10 in
R1(config-if)#exit
R1(config)#exit
R1#
*Mar  1 00:39:35.543: %SYS-5-CONFIG_I: Configured from console by console
R1#wr
Building configuration...
[OK]
R1#
```

Here we can observe that when PC 1 tries to send data to PC 3, its access is denied.

```
PC1> ping 192.168.3.1
*192.168.1.50 icmp_seq=1 ttl=255 time=40.450 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=2 ttl=255 time=15.823 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=3 ttl=255 time=15.552 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=4 ttl=255 time=16.240 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=5 ttl=255 time=15.719 ms (ICMP type:3, code:1, Destination host unreachable)
```

But when PC 3 tries to send data to PC 1, its access is denied.

```
PC1> ping 192.168.2.1
*192.168.1.50 icmp_seq=1 ttl=255 time=15.886 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=2 ttl=255 time=15.828 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=3 ttl=255 time=16.072 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=4 ttl=255 time=15.622 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.1.50 icmp_seq=5 ttl=255 time=15.900 ms (ICMP type:3, code:1, Destination host unreachable)
```

But when PC 1 tries to send data to PC 2, its access is not denied.

```
PC1> ping 192.168.1.1
192.168.1.1 icmp_seq=1 ttl=64 time=0.001 ms
192.168.1.1 icmp_seq=2 ttl=64 time=0.001 ms
192.168.1.1 icmp_seq=3 ttl=64 time=0.001 ms
192.168.1.1 icmp_seq=4 ttl=64 time=0.001 ms
192.168.1.1 icmp_seq=5 ttl=64 time=0.001 ms
```

Based on this observation we can assume that we have successfully used a standard access list.

To remove any acl configuration = In config mode no access-list <access-list-number>

- **To apply External ACL:**

Here the scenario we are going to use is that PC 2 should not be able to send data to PC 3.

To achieve this, we are going to use External ASL. The concept of external ASL states that we need to place it close to its source. So, in terms of the network topology and the position of PC 2, the closest place that is near the source would be Router 2. We should also note that the number of the external ASL should be in the range of 100-199, in our case we have chosen 141. So, by configuring the access list to 'deny', we can deny all communication from PC 2 to PC 3. The difference between the standard ASL and the external ASL in this regard is that in external ASL we have to mention both the source and the destination, whereas in standard ASL we have to only mention the destination.

```
R3(config)#access-list 141 deny icmp host 192.168.2.1 host 192.168.3.1
R3(config)#do show access-list
Extended IP access list 141
    10 deny icmp host 192.168.2.1 host 192.168.3.1
```

Note: The 'deny' keyword tends to deny access to the entire network, so here we have to initially give permission for other devices to send data to PC 2 and then group the output that is provided by PC 2 via the serial interface 1/1 in order to prevent it from sending any data to PC 3. To achieve this, we use the 'permit' keyword in the access list configuration

```
R3(config)#access-list 141 permit icmp any any
R3(config)#do show access-list
Extended IP access list 141
    10 deny icmp host 192.168.2.1 host 192.168.3.1
    20 permit icmp any any
```

```
R3(config)#interface se1/1
R3(config-if)#ip access-gr
% Incomplete command.

R3(config-if)#ip access-group 141 out
R3(config-if)#do access list 141
access list 141
% Incomplete command.

R3(config-if)#do access-list 141
access-list 141
          ^
% Invalid input detected at '^' marker.

R3(config-if)#do show access-list 141
Extended IP access list 141
    10 deny icmp host 192.168.2.1 host 192.168.3.1
    20 permit icmp any any
R3(config-if)#exit
R3(config)#exit
R3#
*Mar  1 00:43:03.411: %SYS-5-CONFIG_I: Configured from console by console
```

Here we can observe that when PC 2 tries to send data to PC 3, its access is denied.

```
PC2> ping 192.168.3.1
*192.168.2.50 icmp_seq=1 ttl=255 time=15.550 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.2.50 icmp_seq=2 ttl=255 time=15.902 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.2.50 icmp_seq=3 ttl=255 time=16.180 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.2.50 icmp_seq=4 ttl=255 time=15.628 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.2.50 icmp_seq=5 ttl=255 time=16.155 ms (ICMP type:3, code:1, Destination host unreachable)
```

Similarly, when PC 3 tries to send data to PC 2, nothing gets through since all communication has

been blocked between them.

```
PC3> ping 192.168.2.1
*192.168.3.50 icmp_seq=1 ttl=255 time=14.997 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.3.50 icmp_seq=2 ttl=255 time=15.599 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.3.50 icmp_seq=3 ttl=255 time=15.828 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.3.50 icmp_seq=4 ttl=255 time=15.355 ms (ICMP type:3, code:1, Destination host unreachable)
*192.168.3.50 icmp_seq=5 ttl=255 time=15.665 ms (ICMP type:3, code:1, Destination host unreachable)
```

But when PC 1 tries to send data to PC 2, its access is not denied.

```
PC3> ping 192.168.3.1
192.168.3.1 icmp_seq=1 ttl=64 time=0.001 ms
192.168.3.1 icmp_seq=2 ttl=64 time=0.001 ms
192.168.3.1 icmp_seq=3 ttl=64 time=0.001 ms
192.168.3.1 icmp_seq=4 ttl=64 time=0.001 ms
192.168.3.1 icmp_seq=5 ttl=64 time=0.001 ms
```

But when PC 1 tries to send data to PC 2, its access is not denied.

```
PC1> ping 192.168.2.1
192.168.2.1 icmp_seq=1 timeout
192.168.2.1 icmp_seq=2 timeout
84 bytes from 192.168.2.1 icmp_seq=3 ttl=62 time=36.971 ms
84 bytes from 192.168.2.1 icmp_seq=4 ttl=62 time=40.185 ms
84 bytes from 192.168.2.1 icmp_seq=5 ttl=62 time=41.151 ms
```

Based on this observation we can assume that we have successfully used an external access list.