

Exploring and Configuring Linux and Windows Firewalls

I've done this lab to explore Linux firewall iptables and Windows firewall. The objective is to understand the functionality and configuration of both firewalls, aiming to enhance network security.

Task A: Configuring Linux Firewalls

Finding the IP address:

To find the IP address of the eth0 network interface in the Kali VM, you need to start the VM, log in, and open a terminal. Once in the terminal, you can identify and note the IP address associated with the eth0 interface using ifconfig command

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.81.129 netmask 255.255.255.0 broadcast 192.168.81.255
          inet6 fe80::3db2:696b:fece:ac65 prefixlen 64 scopeid 0x20<link>
            ether 00:0c:29:65:ce:0c txqueuelen 1000 (Ethernet)
              RX packets 42 bytes 5640 (5.5 KiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 41 bytes 6004 (5.8 KiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

We used the command iptables -L to view the existing firewall rules

```
(kali㉿kali)-[~]
└─$ sudo iptables -L
[sudo] password for kali:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Adding a new rule to the firewall:

To add a new rule to the INPUT chain of the firewall, execute the following command:

```
(kali㉿kali)-[~]
└─$ sudo iptables -A INPUT -i lo -j ACCEPT
```

This command adds an "ACCEPT" rule to the INPUT chain for all packets coming through the loopback interface ("`-i lo`"). The loopback interface allows communication between different processes on the same machine.

To list the firewall rules with statistics, including the rule added to the INPUT chain, you can use the following command using option `-v`:

```
(kali㉿kali)-[~]
└─$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in      out      source               destination
    0     0 ACCEPT      all   --    lo      any     anywhere            anywhere
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in      out      source               destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in      out      source               destination
```

This command displays the firewall rules for all chains, along with the statistics related to each rule. You can refer to the output to identify the statistics specifically

related to the rule added in the INPUT chain.

Now ping the loopback interface:

```
(kali㉿kali)-[~]
└─$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.015 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.025 ms
^C
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.015/0.036/0.069/0.023 ms
```

Show the statistics related to the rule again:

```
(kali㉿kali)-[~]
└─$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out    source          destination
    6   504 ACCEPT      all  --  lo      any    anywhere
                                             anywhere

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out    source          destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out    source          destination
```

Now add an accept rule for all packets to the OUTPUT chain:

```
(kali㉿kali)-[~]
└─$ sudo iptables -A OUTPUT -o lo -j ACCEPT

(kali㉿kali)-[~]
└─$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out    source          destination
    6   504 ACCEPT      all  --  lo      any    anywhere
                                             anywhere

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out    source          destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out    source          destination
    0     0 ACCEPT      all  --  any    lo      anywhere
                                             anywhere
```

Try to ping the localhost then show the statistics of the chains

Note that the packets are passing through the INPUT and OUTPUT chains so the statistics of

the two chains will increase. As shown in the figure below:

```
(kali㉿kali)-[~]
$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.020 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.029 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.068 ms
^C
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2037ms
rtt min/avg/max/mdev = 0.020/0.039/0.068/0.020 ms

(kali㉿kali)-[~]
$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
    12   1008 ACCEPT      all  --  lo      any    anywhere             anywhere
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
    6    504 ACCEPT      all  --  any    lo      anywhere             anywhere
```

We can notice that the packets increased

- Now from your Host OS, try to ping the Kali VM (IP address in question 1):

Select Command Prompt

```
Microsoft Windows [Version 10.0.19044.2364]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>ping 192.168.81.129

Pinging 192.168.81.129 with 32 bytes of data:
Reply from 192.168.81.129: bytes=32 time<1ms TTL=64
Reply from 192.168.81.129: bytes=32 time<1ms TTL=64
Reply from 192.168.81.129: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.81.129:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
Control-C
^C
```

We can see that there are no packets loss

b) Now configure the firewall to drop all the packets that come to eth0 interface:

```
—(kali㉿kali)-[~]
└─$ sudo iptables -A INPUT -d 192.168.81.129 -j DROP
```

Again, try to ping the Kali VM from your Host OS, you get the following output:

Note that the % of lost packets 100%.

```
C:\Users\USER>ping 192.168.81.129

Pinging 192.168.81.129 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.81.129:
    Packets: Sent = 3, Received = 0, Lost = 3 (100% loss),
Control-C
```

In order to delete the rules added above, enter the following command:

```
└─(kali㉿kali)-[~]
└─$ sudo iptables -F
```

Now try to block the traffic coming from your Host OS to the Kali VM (Run first ipconfig to get the ip address of your HOST OS).

```
Connection-specific DNS Suffix . : Home
Link-local IPv6 Address . . . . . : fe80::bf3d:86cf:f44:93e%20
IPv4 Address. . . . . : 192.168.1.112
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.1.1
```

```
└─(kali㉿kali)-[~]
└─$ sudo iptables -A INPUT -s 192.168.1.112 -j DROP
```

Again, display the list of rules using the following command:

```
└─(kali㉿kali)-[~]
└─$ sudo iptables -A INPUT -s 192.168.1.112 -j DROP

└─(kali㉿kali)-[~]
└─$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
DROP       all   --  192.168.1.112      anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
```

Try to ping the Kali VM. What do you see as output?

```
C:\Users\USER>ping 192.168.81.129

Pinging 192.168.81.129 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.81.129:
    Packets: Sent = 3, Received = 0, Lost = 3 (100% loss),
Control-C
^C
```

Now try to ping the following address: www.uj.edu.sa. You get an output similar to the following:

```
└─(kali㉿kali)-[~]
└─$ ping www.uj.edu.sa
PING www.uj.edu.sa (82.147.204.98) 56(84) bytes of data.
^C
— www.uj.edu.sa ping statistics —
5 packets transmitted, 0 received, 100% packet loss, time 4087ms

└─(kali㉿kali)-[~]
└─$ ping 80.147.204.98
PING 80.147.204.98 (80.147.204.98) 56(84) bytes of data.
64 bytes from 80.147.204.98: icmp_seq=1 ttl=128 time=116 ms
64 bytes from 80.147.204.98: icmp_seq=2 ttl=128 time=112 ms
64 bytes from 80.147.204.98: icmp_seq=3 ttl=128 time=110 ms
64 bytes from 80.147.204.98: icmp_seq=4 ttl=128 time=113 ms
^C
— 80.147.204.98 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 110.373/112.768/115.806/2.054 ms
```

Now let's block access to this site, but first enter the following command to list all the firewall rules: iptables –L.

```
└$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP      all  --  192.168.1.112          anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Let's try to block the ping to this site, enter the following command:

```
└(kali㉿kali)-[~]
└$ sudo iptables -A OUTPUT -s 0/0 -d 80.147.204.98 -j DROP
```

Use iptables –L again.

```
└(kali㉿kali)-[~]
└$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP      all  --  192.168.1.112          anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP      all  --  anywhere             mail.scsbit.de
```

Now try to ping www.uj.edu.sa again.

```
└(kali㉿kali)-[~]
└$ ping www.uj.edu.sa
PING www.uj.edu.sa (82.147.204.98) 56(84) bytes of data.
^C
--- www.uj.edu.sa ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5105ms
```

To remove all the previously added rules enter: iptables –F.

(use iptables -L to make sure that the rules are removed)

```
└─(kali㉿kali)-[~]
└─$ sudo iptables -F

└─(kali㉿kali)-[~]
└─$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination

Chain FORWARD (policy ACCEPT)
target     prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
```

Task B: Configuring Windows Firewall

- Block access to this URL: <https://pastebin.com>



The screenshot shows the Windows Defender Firewall with Advanced Security interface. The left navigation pane includes 'Inbound Rules', 'Outbound Rules' (selected), 'Connection Security Rules', and 'Monitoring'. The main area displays a table titled 'Outbound Rules' with columns: Name, Group, Profile, and Enabled. The table lists various rules such as Cortana, ShellExperienceHost, StartMenuExperienceHost, FirewallAPI.dll, Skype, 3D Viewer, AllJoyn Router (TCP-Out), AllJoyn Router (UDP-Out), and App Installer. The right pane contains an 'Actions' menu with options like 'New Rule...', 'Filter by Profile', 'Filter by State', 'Filter by Group', 'View', 'Refresh', 'Export List...', and 'Help'.

New Outbound Rule Wizard

Rule Type

Select the type of firewall rule to create.

The screenshot shows the 'Rule Type' step of the New Outbound Rule Wizard. On the left, a 'Steps:' sidebar lists: Rule Type (selected), Program, Protocol and Ports, Scope, Action, Profile, and Name. The main panel asks 'What type of rule would you like to create?'. It lists four options: 'Program' (radio button), 'Port' (radio button), 'Predefined:' (radio button), and 'Custom' (radio button, selected). The 'Predefined:' option has a dropdown menu showing '@FirewallAPI.dll,-80200'. The 'Custom' option has a placeholder 'Custom rule.'.

```
C:\WINDOWS\system32>nslookup www.pastebin.com
Server: static-86-51-35-24.mobily.com.sa
Address: 86.51.35.24

Non-authoritative answer:
Name: www.pastebin.com
Addresses: 2606:4700:10::ac43:22aa
           2606:4700:10::6814:448f
           2606:4700:10::6814:438f
           104.20.68.143
           172.67.34.170
           104.20.67.143
```

New Outbound Rule Wizard

X

Scope

Specify the local and remote IP addresses to which this rule applies.

Steps:

- Rule Type
- Program
- Protocol and Ports
- Scope**
- Action
- Profile
- Name

Which local IP addresses does this rule apply to?

- Any IP address
 These IP addresses:

Add...
Edit...
Remove

Customize the interface types to which this rule applies:

Customize...

Which remote IP addresses does this rule apply to?

- Any IP address
 These IP addresses:

2606:4700:10::ac43:22aa
2606:4700:10::6814:448f
2606:4700:10::6814:438f
104.20.68.143
172.67.34.170
104.20.67.143

Add...
Edit...
Remove

< Back

Next >

Cancel

New Outbound Rule Wizard

X

Action

Specify the action to be taken when a connection matches the conditions specified in the rule.

Steps:

- Rule Type
- Program
- Protocol and Ports
- Scope
- Action**
- Profile
- Name

What action should be taken when a connection matches the specified conditions?

- Allow the connection**

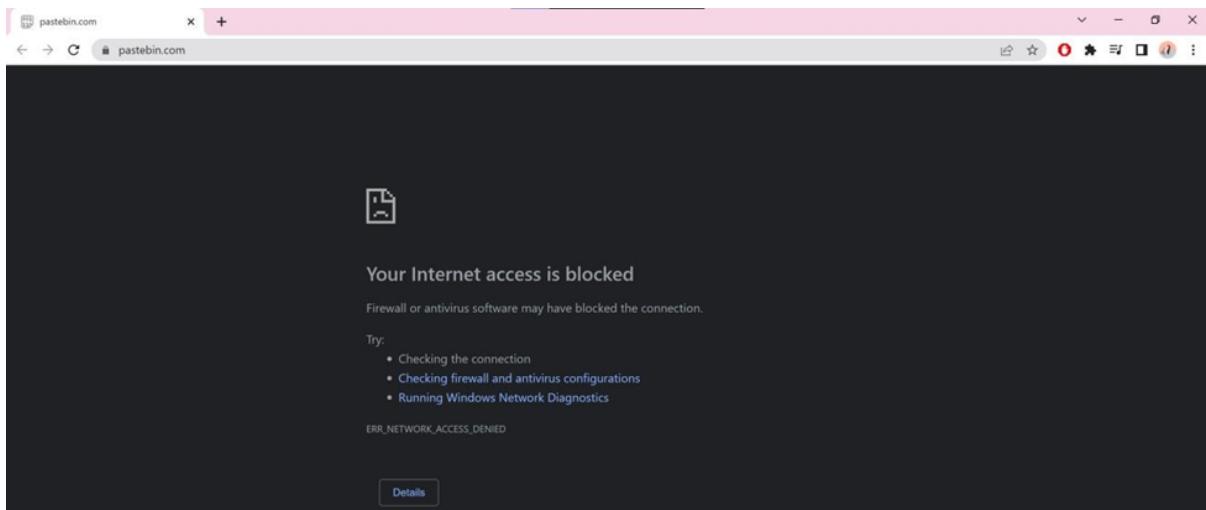
This includes connections that are protected with IPsec as well as those are not.

- Allow the connection if it is secure**

This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.

Customize...

- Block the connection**



- Prevent any application inside windows from accessing the Internet.

Name	Group	Profile	Enabled
@[Microsoft.Windows.Cortana_1.13.0.1836_...	@[Microsoft.Windows.Corta...	All	Yes
@[Microsoft.Windows.ShellExperienceHos...	@[Microsoft.Windows.ShellE...	All	Yes
@[Microsoft.Windows.StartMenuExperi...	@[Microsoft.Windows.Start...	All	Yes
@FirewallAPI.dll.-80204	@FirewallAPI.dll.-80200	All	Yes
Skype	(78E1CD88-49E3-476E-B926...	All	Yes
Skype	(78E1CD88-49E3-476E-B926...	All	Yes
3D Viewer	3D Viewer	All	Yes
AllJoyn Router (TCP-Out)	AllJoyn Router	Domai...	Yes
AllJoyn Router (UDP-Out)	AllJoyn Router	Domai...	Yes
App Installer	App Installer	All	Yes

Program

Specify the full program path and executable name of the program that this rule matches.

Steps:

- Rule Type
- Program**
- Action
- Profile
- Name

Does this rule apply to all programs or a specific program?

All programs
Rule applies to all connections on the computer that match other rule properties.

This program path:
 Browse...
 Example: c:\path\program.exe
 %ProgramFiles%\browser\browser.exe

Action

Specify the action to be taken when a connection matches the conditions specified in the rule.

Steps:

- Rule Type
- Program
- **Action**
- Profile
- Name

What action should be taken when a connection matches the specified conditions?

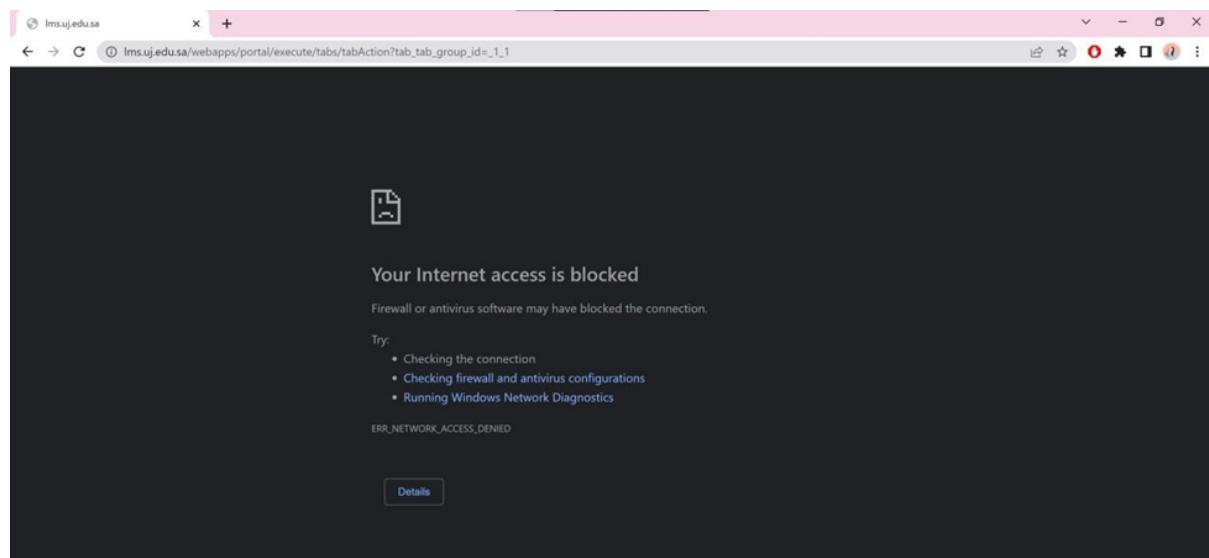
Allow the connection
This includes connections that are protected with IPsec as well as those are not.

Allow the connection if it is secure
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.

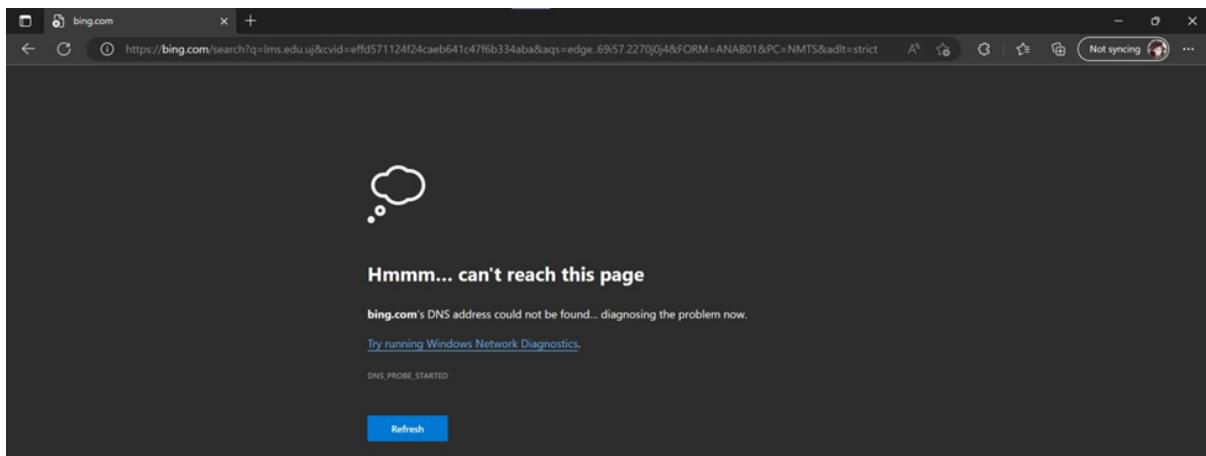
Block the connection

[Customize...](#)

Google Chrome



Microsoft edge



Conclusion

Through this lab, I obtained practical knowledge in configuring and managing Linux and Windows firewalls. I learned how to allow or block specific traffic, secure network interfaces, and restrict access to websites and applications. This experience will enable me to strengthen network defense and protect against unauthorized access.