Using Linux IPTables

Vidhan
2022568
CSE 232 - Computer Networks - PA03

Q1 - Setting up the VMs and Port Forwarding

Deliverables

Set up four VMs as shown in the figure. Use the same setup for the entire assignment.
a) Configure the IP addresses and routes for all VMs, as shown in the figure
b) Configure VM2 as the gateway such that it can forward the incoming traffic to one of the servers – add forwarding functionality

Methodology

I set up four VMs using VirtualBox, each running Ubuntu 22.04.
To configure the IP addresses and routes for each VM according to the provided figure, I used VirtualBox's network settings to add a NAT network (as shown in the image above). Then, I configured the IP addresses and routes using the following steps:

sudo nano /etc/netplan/01-netcfg.yaml

then added the following configuration:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    [port name]:
      match:
        macaddress: [mac address]
      set-name: [new port name]
      addresses:
      - [IP address]/[subnet mask]
      gateway4: [gateway IP]
      dhcp4: false
      routes:
      - to: [destination network addr]/[subnet mask]
```

via: [gateway IP]

then applied the changes using the following command:

sudo netplan apply

Using the above method, I have configured the IP addresses and routes for all the VMs and interfaces as shown in the figure. This way the configurations and routes persist even after a reboot.

- The following are the results:

VM1 - Client:
port ens34 has IP address 20.1.1.1/24 and default gateway 20.1.1.2, with a route to 40.1.1.0/24 via 20.1.1.2

VM2 - Gateway:
port ens34 has IP address 20.1.1.2/24 and port ens36 has IP address 40.1.1.2 -- acting as a gateway for the networks 20.1.1.0/24 and 40.1.1.0/24 respectively.

VM3 - Server 1:
port ens34 has IP address 40.1.1.1/24 and default gateway 40.1.1.2, with a route to 20.1.1.0/24 via 40.1.1.2

VM4 - Server 2:
port ens34 has IP address 40.1.1.3/24 and default gateway 40.1.1.2, with a route to 20.1.1.0/24 via 40.1.1.2

Q2 - Traffic filtering at the gateway VM

Deliverables

a) The gateway must block all traffic (except for ping) destined to the server 40.1.1.1/24.
b) The gateway must block only TCP traffic initiated by 20.1.1.1/24.

sudo nano /etc/netplan/01-netcfg.yaml

Then, added the following configuration:

```
network:
    version: 2
    renderer: networkd
    ethernets:
```

```
[port name]:
  match:
    macaddress: [mac address]
  set-name: [new port name]
  addresses:
    - [IP address]/[subnet mask]
  gateway4: [gateway IP]
  dhcp4: false
  routes:
    - to: [destination network addr]/[subnet mask] via: [gateway IP]
```

Then applied the changes using:

sudo netplan apply

Using the above method, I have configured the IP addresses and routes for all the VMs and interfaces as shown in the figure. This way, the configurations and routes persist even after a reboot.

Results

VM1 - Client:
port ens34 has IP address 20.1.1.1/24 and default gateway 20.1.1.2, with a route to 40.1.1.0/24 via 20.1.1.2

VM2 - Gateway:
port ens34 has IP address 20.1.1.2/24 and port ens36 has IP address 40.1.1.2 -- acting as a gateway for the networks 20.1.1.0/24 and 40.1.1.0/24 respectively.

VM3 - Server 1:
port ens34 has IP address 40.1.1.1/24 and default gateway 40.1.1.2, with a route to 20.1.1.0/24 via 40.1.1.2

VM4 - Server 2:
port ens34 has IP address 40.1.1.3/24 and default gateway 40.1.1.2, with a route to 20.1.1.0/24 via 40.1.1.2

Each VM has a port enp0s8 which is a Bridged Adapter connected to the host network for internet access. I have configured VM2 as the gateway such that it can forward the incoming traffic to one of the servers using the following commands:

Q2 - Traffic filtering at the gateway VM

Deliverables

a) The gateway must block all traffic (except for ping) destined to the server 40.1.1.1/24. Show that this works; attach the screenshot.

b) The gateway must block only TCP traffic initiated by 20.1.1.1/24. Show that this works; attach the screenshot.

Methodology

Using IPTables, I have configured the gateway VM to block all traffic (except for ping) destined to the server 40.1.1.1/24 using OUTPUT and FORWARD chains. The following commands were used to configure the IPTables:

[Commands for OUTPUT and FORWARD chain here]

Results from ping and nc commands confirmed that packets are being transmitted and received as requested. However, TCP connections using nc are not established as expected.

Q3 - Using the configuration from Q2 to measure the performance of the gateway

Deliverables

a) Use 'iperf2' tool to test the TCP and UDP bandwidth between 20.1.1.1/24 and 40.1.1.3/24. Attach the screenshot.
b) Calculate the minimum, average, and maximum RTT. Attach the screenshot.

Methodology

I used the iperf2 tool to test the TCP and UDP bandwidth between the client 20.1.1.1/24 and the server 40.1.1.3 using appropriate commands. The following results were obtained for UDP traffic and RTT calculations using the ping command.

Q4 - Network address translation at the gateway VM

Deliverables

a) Change the source IP address of every packet from 20.1.1.1/24 to 40.1.1.2/24.
b) When the response packet arrives at the gateway, revert the destination IP address to the original.
c) Validate this by observing packets at each VM using Wireshark/tcpdump. Attach the screenshot.

Methodology

Using IPTables, I configured the gateway to modify the source IP address for packets from 20.1.1.1/24 using the POSTROUTING chain. Responses are reverted using the PREROUTING chain.

Q5 - Load balancing at the gateway VM

Deliverables

a) Balance traffic from 20.1.1.1/24 to servers 40.1.1.1/24 and 40.1.1.3/24 with probabilities of 0.8 and 0.2, respectively.
b) Test the configuration using ping packets.

Methodology

Load balancing was configured using IP tables with PREROUTING chain and DNAT target for probability-based routing. Testing verified the correct distribution of packets between servers.