Transform iptables into a TCP load balancer

Task requirement

To create two new Caddy LB or two nginx and after creating, it needs to forward the 50% of Load to another LB.

iptables is a user-space utility program that allows a system administrator to configure the IP packet filter rules of the Linux kernel firewall.

Environment details

IPtable rules will be applicable for all types of OS.

UbuntuNAME="Ubuntu"
VERSION="20.04.4

List of tools and technologies

- Caddy:- CADDY_VERSION=v2.7.2

I have configre two vm and install caddy both vm

VM1 >>>> 192.168.122.24

sudo apt install caddy

VM2 >>>> 192.168.122.13
sudo apt install caddy

Vm1 >>>>Change the index.html

jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ cd /usr/share/caddy

jai2@jai2-Standard-PC-Q35-ICH9-2009:/usr/share/caddy$ ls
index.html
jai2@jai2-Standard-PC-Q35-ICH9-2009:/usr/share/caddy$ vim index.html
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Hello jaiprakash!</title>
</head>
<body>

```
        <h1>Hello jaiprakash!</h1>
</body>
</html>
```

Vm2 >>>> Change the index.html

```
jai5@jai5-Standard-PC-Q35-ICH9-2009:/usr/share/caddy$ ls
index.html
jai5@jai5-Standard-PC-Q35-ICH9-2009:/usr/share/caddy$ sudo vim index.html
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Hello jaiprakash!</title>
</head>
<body>
        <h1>Hello jaiprakash!</h1>
</body>
</html>
```

>>>> sudo apt install curl

I am able to curl Page from each other.

jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ curl http://192.168.122.24
```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Hello jaiprakash!</title>
</head>
<body>
        <h1>Hello jaiprakash!</h1>
</body>
```

jai5@jai5-Standard-PC-Q35-ICH9-2009:/etc/nginx$ curl http://192.168.122.13
```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Hello world!</title>
```

```
</head>
<body>
        <h1>Hello world!</h1>
</body>
</html>
```

I am able to curl both from the Base Machine.

```
indianrenters@jai:~$ curl http://192.168.122.24
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Hello jaiprakash!</title>
</head>
<body>
        <h1>Hello jaiprakash!</h1>
</body>
```

```
indianrenters@jai:~$ curl http://192.168.122.13
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Hello world!</title>
</head>
<body>
        <h1>Hello world!</h1>
</body>
</html>
```

==================now work in Vm1 192.168.122.24=================

a) Enable IP Forwarding:

Activate IP forwarding to facilitate traffic routing between interfaces:

```
 >>>>>>>>>>>>>>>>>> >>>>>>vm1 and vm2>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Same process===
jai3@jai3-Standard-PC-Q35-ICH9-2009:~$ sudo vim /etc/sysctl.conf
```

uncomment in the file below line.

```
 net.ipv4.ip_forward=1
```

```
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.accept_source_route = 0

net.ipv4.conf.all.log_martians = 1
```

jai@jai-Standard-PC-Q35-ICH9-2009:~$ sudo sysctl -p

```
net.ipv4.ip_forward = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.secure_redirects = 1
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.all.log_martians = 1
```

Modify source IP for forwarding: this command run vm1 source

jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ sudo iptables -A POSTROUTING -t nat -p tcp -d 192.168.122.13 --dport 80 -j SNAT --to-source 192.168.122.24

Set default connection drop vm1
This implies that by default, any incoming traffic that doesn't match specific predefined rules will be dropped or rejected. In other words, if the firewall doesn't have a rule that explicitly allows the traffic to pass through, it will block it as a security measure. This approach follows the principle of allowing only known and authorised traffic, enhancing the network's security.

jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ sudo iptables -t filter -P FORWARD DROP

Accept Traffic to the Server:
This involves creating rules that explicitly allow certain types of traffic to reach a designated server. For instance, if you have a server at IP address 192.168.122.44 and it's listening on port 80, you can configure the firewall to accept incoming traffic destined for that server's IP and port. This is done using firewall rules that specify the source, destination, protocol, and port of the allowed traffic.
Allow specific traffic from particular sources and to specific destinations (servers) on specific ports (--dport for destination port and --sport for source port). This creates a controlled and secure network environment where only the specified traffic is permitted to traverse the firewall.
This command on run vm1

jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ sudo iptables -t filter -A FORWARD -d 192.168.122.13 -p tcp --dport 80 -j ACCEPT
jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ sudo iptables -t filter -A FORWARD -s 192.168.122.13 -p tcp --sport 80 -j ACCEPT

This command run on vm2

jai5@jai5-Standard-PC-Q35-ICH9-2009:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT

Apply Random Load Balancing and then Round Robin:-

A technique used in network environments to distribute incoming traffic across multiple servers in a random manner. This method is employed to optimize resource utilization, prevent overload on individual servers, and enhance the overall performance and reliability of a system.

In this case, incoming traffic destined for the IP address 192.168.122.123 on port 80 is subject to random distribution between one destinations ([192.168.122.13:80](192.168.122.13:80) The --mode random option ensures that the traffic is distributed based on a randomized algorithm, with different probabilities assigned to each destination (--probability 0.33 and --probability 0.5 in this case).

The goal of applying random load balancing is to distribute traffic unpredictably, achieving a fair distribution of incoming requests among the specified destinations. This way, the servers can collectively handle the load more efficiently, minimizing the risk of overloading any single server and contributing to improved performance and fault tolerance.

This command run Vm1 (source)
jai2@jai2-Standard-PC-Q35-ICH9-2009:~$ sudo iptables -A PREROUTING -t nat -p tcp -d 192.168.122.24 --dport 80 \
        -m statistic --mode random --probability 0.33 \
        -j DNAT --to-destination 192.168.122.13:80

Capture Specific Port Traffic:
Use tcpdump to monitor network traffic on a specific port:-
tcpdump -i enp1s0 port 80 -n

Output of TCP Dump

1st curl hit
13:16:56.040352 IP 192.168.122.1.51940 > 192.168.122.24.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 483302308 ecr 531586825], length 0
13:16:56.040412 IP 192.168.122.1.51940 > 192.168.122.24.80: Flags [P.], seq 1:79, ack 1, win 502, options [nop,nop,TS val 483302308 ecr 531586825], length 78: HTTP: GET / HTTP/1.1
13:16:56.040425 IP 192.168.122.24.80 > 192.168.122.1.51940: Flags [.], ack 79, win 509, options [nop,nop,TS val 531586825 ecr 483302308], length 0

2nd curl Hit

13:16:57.280164 IP 192.168.122.1.51948 > 192.168.122.24.80: Flags [S], seq 628312303, win 64240, options [mss 1460,sackOK,TS val 483303548 ecr 0,nop,wscale 7], length 0
13:16:57.280227 IP 192.168.122.24.51948 > 192.168.122.13.80: Flags [S], seq 628312303, win 64240, options [mss 1460,sackOK,TS val 483303548 ecr 0,nop,wscale 7], length 0

I have also done Load Testing:-
sudo apt install apache2-utils
Install ab command at local for load testing
install apache2-utils for ab command for Load Testing

run load test using ab command :-
ab -n 1000 -c 100 http://192.168.122.24:80/
jai@jai-Standard-PC-Q35-ICH9-2009:~$ ab -n 1000 -c 100 http://192.168.122.24:80/
indianrenters@jai:~$ sudo ab -n 1000 -c 100 http://192.168.122.24:80/
[sudo] password for indianrenters:
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.122.24 (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests

Server Software: Caddy
Server Hostname: 192.168.122.24
Server Port: 80

Document Path: /
Document Length: 231 bytes

Concurrency Level: 100
Time taken for tests: 0.312 seconds
Complete requests: 1000
Failed requests: 522
   (Connect: 0, Receive: 0, Length: 522, Exceptions: 0)
Total transferred: 443780 bytes
HTML transferred: 225780 bytes
Requests per second:3203.70 [#/sec] (mean)
Time per request: 31.214 [ms] (mean)
Time per request: 0.312 [ms] (mean, across all concurrent requests)
Transfer rate: 1388.42 [Kbytes/sec] received

Connection Times (ms)
          min mean[+/-sd] median   max
Connect: 0   3   3.1   2   13
Processing: 1   25 24.8   13   109
Waiting: 0   24 24.3   12   93
Total: 1   28 24.6   20   111

Percentage of the requests served within a certain time (ms)
  50% 20
  66% 38
  75% 49
  80% 53
  90% 61
  95% 76
  98% 87
  99% 90
 100% 111 (longest request)

Output curl

curl http://192.168.122.24
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hello jaiprakash!</title>
</head>
<body>
    <h1>Hello jaiprakash!</h1>
</body>
</html>

curl http://192.168.122.24
i<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hello world!</title>
</head>
<body>
    <h1>Hello world!</h1>
</body>
</html>