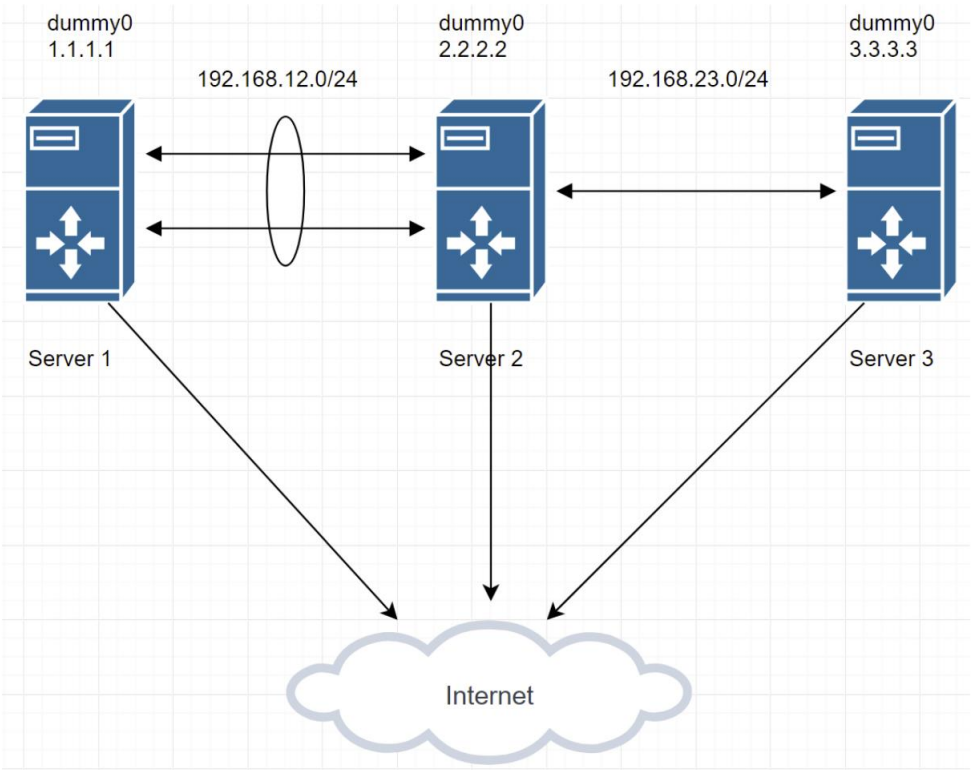
ЗАДАНИЕ №5

Топология:



1) На сервере Server1 установить ipvsadm.
2) На сервере Server1 установить docker и запустить 2 контейнера с Nginx.
3) Создать интерфейс dummy2 и назначить ему IP-адрес 111.111.111.111/32.
4) Проанонсировать этот IP в сеть из трёх серверов.
5) Настроить ipvs для передачи http запросов с сервера server3 в оба контейнера в Nginx используя механизм балансировки round-robin.
6) Удостовериться, что запросы балансируются между контейнерами путем проверки access.log внутри каждого из них.

РЕШЕНИЕ:

1)

```
2. server1
[root@server1 ~]# ipvsadm -l
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
[root@server1 ~]#
```

2)

```
[root@server1 ~]# docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS     NAMES
91c3719b8a8a   nginx     "/docker-entrypoint.…"   About a minute ago   Up About a minute   80/tcp    nginx-B
8c501ca69328   nginx     "/docker-entrypoint.…"   4 minutes ago        Up 4 minutes        80/tcp    nginx-A
[root@server1 ~]#
```

3)

```
[root@server1 ~]# ip addr show dummy2
7: dummy2: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueu
    link/ether 5a:d9:71:1b:37:9b brd ff:ff:ff:ff:ff:ff
    inet 111.111.111.111/32 brd 111.111.111.111 scope global d
       valid_lft forever preferred_lft forever
    inet6 fe80::58d9:71ff:fe1b:379b/64 scope link
       valid_lft forever preferred_lft forever
```

4) Переписываем анонсируемые сети:
network 1.1.1.1/32 area 0
network 111.111.111.111/32 area 0
network 192.168.12.0/24 area 0

С server2 проверим прилетел ли маршрут:

```
[root@server2 ~]# ip r get 111.111.111.111
111.111.111.111 via 192.168.12.1 dev nm-team src 192.168.12.2
    cache
[root@server2 ~]#
```

5)

```
[root@server1 ~]# ipvsadm -A -t 111.111.111.111:80 -s rr
[root@server1 ~]# ipvsadm -l -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  111.111.111.111:80 rr
[root@server1 ~]# ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.2 -m
[root@server1 ~]# ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.3 -m
[root@server1 ~]# ipvsadm -l -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  111.111.111.111:80 rr
  -> 172.17.0.2:80                Masq    1      0          0
  -> 172.17.0.3:80                Masq    1      0          0
```

6)

```
[root@server1 nginx]# docker logs -n 3 nginx-A
192.168.23.2 - - [16/Jul/2021:20:06:37 +0000] "GET / HTTP/1.1" 200 10 "-" "curl/7.29.0" "-"
192.168.23.2 - - [16/Jul/2021:20:09:34 +0000] "GET / HTTP/1.1" 200 10 "-" "curl/7.29.0" "-"
192.168.23.2 - - [16/Jul/2021:20:10:18 +0000] "GET / HTTP/1.1" 200 10 "-" "curl/7.29.0" "-"
[root@server1 nginx]# docker logs -n 3 nginx-B
192.168.23.2 - - [16/Jul/2021:20:09:07 +0000] "GET / HTTP/1.1" 200 10 "-" "curl/7.29.0" "-"
192.168.23.2 - - [16/Jul/2021:20:10:00 +0000] "GET / HTTP/1.1" 200 10 "-" "curl/7.29.0" "-"
192.168.23.2 - - [16/Jul/2021:20:10:23 +0000] "GET / HTTP/1.1" 200 10 "-" "curl/7.29.0" "-"
[root@server1 nginx]#
```

Из лога можно увидеть, что RR отрабатывает корректно

Также можно отправить с server3 HTTP-запрос к балансировщику:

```
[root@server3 ~]# for i in `seq 2 10`; do curl 111.111.111.111 -s;done
This is A
This is B
This is A
This is B
This is A
This is B
This is A
This is B
This is A
```

P.s. чтобы настройки ipvsadm сохранились после ребута можно сделать:
ipvsadm-save > ipvsadm.sav
Добавить в rc.local: ipvsadm-restore < ipvsadm.sav