Выполнил Мешечкин Д. Инфобез-2345

Задание 1/ 2/3/4: На сервере Server1 установить ipvsadm. На сервере установить docker и запустить 2 контейнера с Nginx. Создать интерфейс dummy2 и назначить ему IP-адрес 111.111.111.111/32, проанонсировать этот IP в сеть из трёх серверов.

Начну с dummy на сервере. Создается новый dummy интерфейс 111.111.111.111/32, воспользовавшись ДЗ к вебинару №3:

#ip link add dummy1 type dummy
#ip addr add 111.111.111.111/32 dev dummy1
#ip link set up dev dummy1

И необходимо изменить правило **dummy.conf** добавив второй интерфейс dummy и опцию количество dummy интерфейсов:

cat > /etc/modprobe.d/dummy.conf
install dummy /sbin/modprobe --ignore-install dummy numdummies=2; /sbin/ip link set dev dummy0 name dummy0; /sbin/ip link set dev dummy1 name dummy1

И добавляю конфиг dummy2
cat > /etc/sysconfig/network-scripts/ifcfg-dummy0
NAME=dummy1
DEVICE=dummy1
MACADDR=00:22:33:ff:ff:ff
IPADDR=111.111.111.111
PREFIX=32
ONBOOT=yes
TYPE=dummy
NM_CONTROLLED=no

И анонсирую его через frr
vtysh
conf t
router ospf
network 111.111.111.111/32 area 0
Сохранить, выйти и можно проверять на серверах R2 и R3. Самое забавное, что на сервере R1 второй dummy спокойно живёт и здравствует переживая ребуты. А на R3 через раз-два-три отваливается иногда.

```
[root@server2 ~]# vtysh -c "sh ip ro"
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, A - Babel, F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K>* 0.0.0.0/0 [0/100] via 192.168.1.1, enp0s3, 00:06:35
O>* 1.1.1.1/32 [110/20] via 192.168.12.101, nm-team, weight 1, 00:05:42
O   2.2.2.2/32 [110/10] via 0.0.0.0, dummy0 onlink, weight 1, 00:06:34
C>* 2.2.2.2/32 is directly connected, dummy0, 00:06:34
O>* 3.3.3.3/32 [110/110] via 192.168.23.3, enp0s8, weight 1, 00:06:19
O>* 111.111.111.111/32 [110/20] via 192.168.12.101, nm-team, weight 1, 00:00:26
K>* 169.254.0.0/16 [0/1002] is directly connected, dummy0, 00:06:34
C>* 192.168.1.0/24 is directly connected, enp0s3, 00:06:35
O   192.168.12.0/24 [110/10] is directly connected, nm-team, weight 1, 00:06:33
C>* 192.168.12.0/24 is directly connected, nm-team, 00:06:33
O   192.168.23.0/24 [110/100] is directly connected, enp0s8, weight 1, 00:06:35
C>* 192.168.23.0/24 is directly connected, enp0s8, 00:06:35
```

```
[root@server3 /]# vtysh -c "sh ip ro"
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, A - Babel, F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K>* 0.0.0.0/0 [0/100] via 192.168.1.1, enp0s3, 01:05:10
O>* 1.1.1.1/32 [110/120] via 192.168.23.2, enp0s8, weight 1, 00:08:22
O>* 2.2.2.2/32 [110/110] via 192.168.23.2, enp0s8, weight 1, 00:08:59
O   3.3.3.3/32 [110/10] via 0.0.0.0, dummy0 onlink, weight 1, 01:05:09
C>* 3.3.3.3/32 is directly connected, dummy0, 01:05:09
O>* 111.111.111.111/32 [110/120] via 192.168.23.2, enp0s8, weight 1, 00:03:06
K>* 169.254.0.0/16 [0/1002] is directly connected, dummy0, 01:05:09
C>* 192.168.1.0/24 is directly connected, enp0s3, 01:05:10
O>* 192.168.12.0/24 [110/110] via 192.168.23.2, enp0s8, weight 1, 00:08:59
O   192.168.23.0/24 [110/100] is directly connected, enp0s8, weight 1, 01:05:10
C>* 192.168.23.0/24 is directly connected, enp0s8, 01:05:10
```

Далее установка и подготовка контейнеров и ipvsadm
yum install -y ipvsadm Dependencies Resolved
yum install docker
systemctl start docker
systemctl status docker
echo "This is A" > /srv/A/index.html
docker run --rm -d -v "/srv/A:/usr/share/nginx/html" --name nginx-A nginx
echo "This is B" > /srv/B/index.html
docker run --rm -d -v "/srv/B:/usr/share/nginx/html" --name nginx-B nginx
docker ps

```
[root@server1 ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
6dad1b7a2ec4        nginx               "/docker-entrypoin..."   3 minutes ago       Up 3 minutes        80/tcp              nginx-B
829d10b28dc8        nginx               "/docker-entrypoin..."   10 minutes ago      Up 10 minutes       80/tcp              nginx-A
```

Контейнеры установлены и запущены. Далее согласно методичке:
curl 172.17.0.2 возвращает Forbidden
ls -laZ /srv/*/index.html
semanage fcontext -a -t httpd_sys_content_t /srv/A/index.html
semanage fcontext -a -t httpd_sys_content_t /srv/B/index.html
restorecon -v /srv/A/index.html
restorecon -v /srv/B/index.html
ls -laZ /srv/*/index.html
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 /srv/A/index.html
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 /srv/B/index.html
далее снова проверяю:

```
[root@server1 ~]# curl 172.17.0.2
This is A
[root@server1 ~]# curl 172.17.0.3
This is B
```

Для настройки IPVS сервиса, как я понял, будет использоваться dummy1 с адресом **111.111.111.111/32**

```
[root@server1 ~]# ip a s dummy1
7: dummy1: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group default qlen 1000
    link/ether 00:22:33:ff:ff:ff brd ff:ff:ff:ff:ff:ff
    inet 111.111.111.111/32 brd 111.111.111.111 scope global dummy1
       valid_lft forever preferred_lft forever
    inet6 fe80::222:33ff:feff:ffff/64 scope link
       valid_lft forever preferred_lft forever
```

ipvsadm -A -t 111.111.111.111:80 -s rr
ipvsadm -l -n

```
[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
```

ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.2 -m
ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.3 -m
ipvsadm -l -n

```
[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
```

Задание 5/6: Настроить ipvs для передачи http запросов с сервера server3 в оба контейнера в Nginx используя механизм балансировки round-robin. Удостовериться, что запросы балансируются между контейнерами путем проверки access.log внутри каждого из них.

Настройка произведена, проверяем с сервера R3
for i in `seq 1 1000`; do curl http://192.168.1.41 -s; done | sort | uniq -c

```
[root@server3 /]# for i in `seq 1 1000`; do curl http://111.111.111.111 -s; done | sort | uniq -c
    500 This is A
    500 This is B
[root@server3 /]# for i in `seq 1 999`; do curl http://111.111.111.111 -s; done | sort | uniq -c
    499 This is A
    500 This is B
[root@server3 /]#
```

На сервере R1 проверяю рейт**:**
ipvsadm -l -n --rate

```
[root@server1 ~]# ipvsadm -l -n --rate
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port                 CPS    InPPS   OutPPS    InBPS   OutBPS
  -> RemoteAddress:Port
TCP  111.111.111.111:80                 56      389      285    25079    28914
  -> 172.17.0.2:80                      28      195      140    12551    14336
  -> 172.17.0.3:80                      28      194      145    12528    14578
[root@server1 ~]#
```

Балансировка настроена.