Лабораторная работа № 16

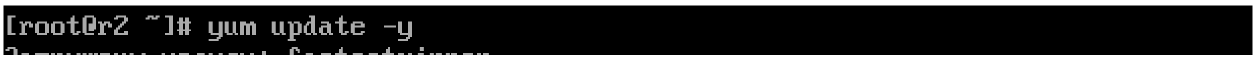
Тема: Сетевые сервисы на Linux. Балансировка нагрузки с помощью ipvsadm.

Цель работы: Изучение основных принципов и практическое применение балансировки нагрузки с использованием ipvsadm в среде Linux.

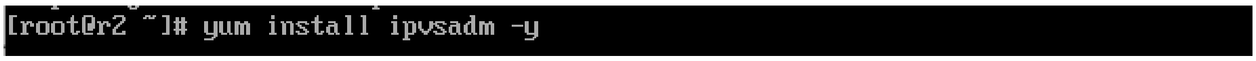
Необходимое оборудование и программное обеспечение:

Виртуальная машина под управлением CentOS 7.

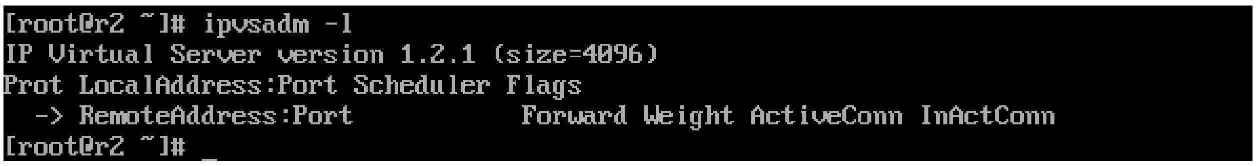
Пример настройки серверов.



```
[root@r2 ~]# yum update -y
```



```
[root@r2 ~]# yum install ipvsadm -y
```



```
[root@r2 ~]# ipvsadm -l
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
[root@r2 ~]# _
```

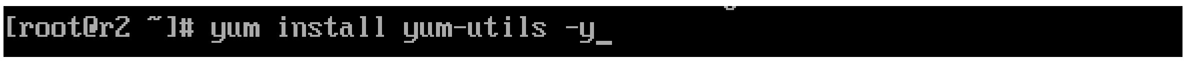
yum install -y yum-utils

yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

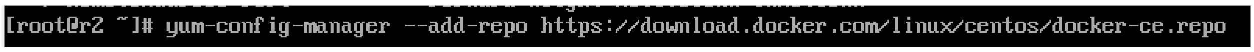
yum install docker-ce -y

systemctl start docker

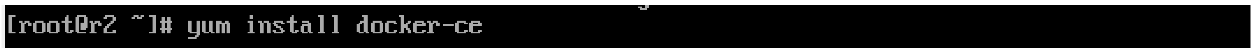
docker run hello-world



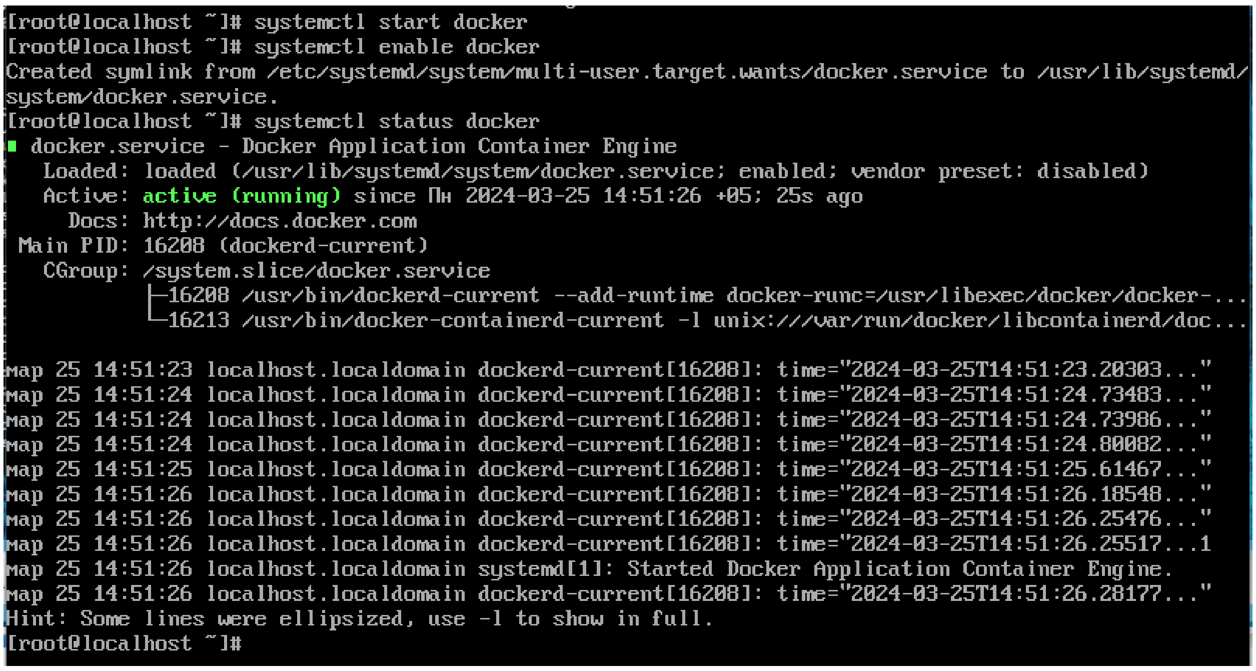
```
[root@r2 ~]# yum install yum-utils -y_
```



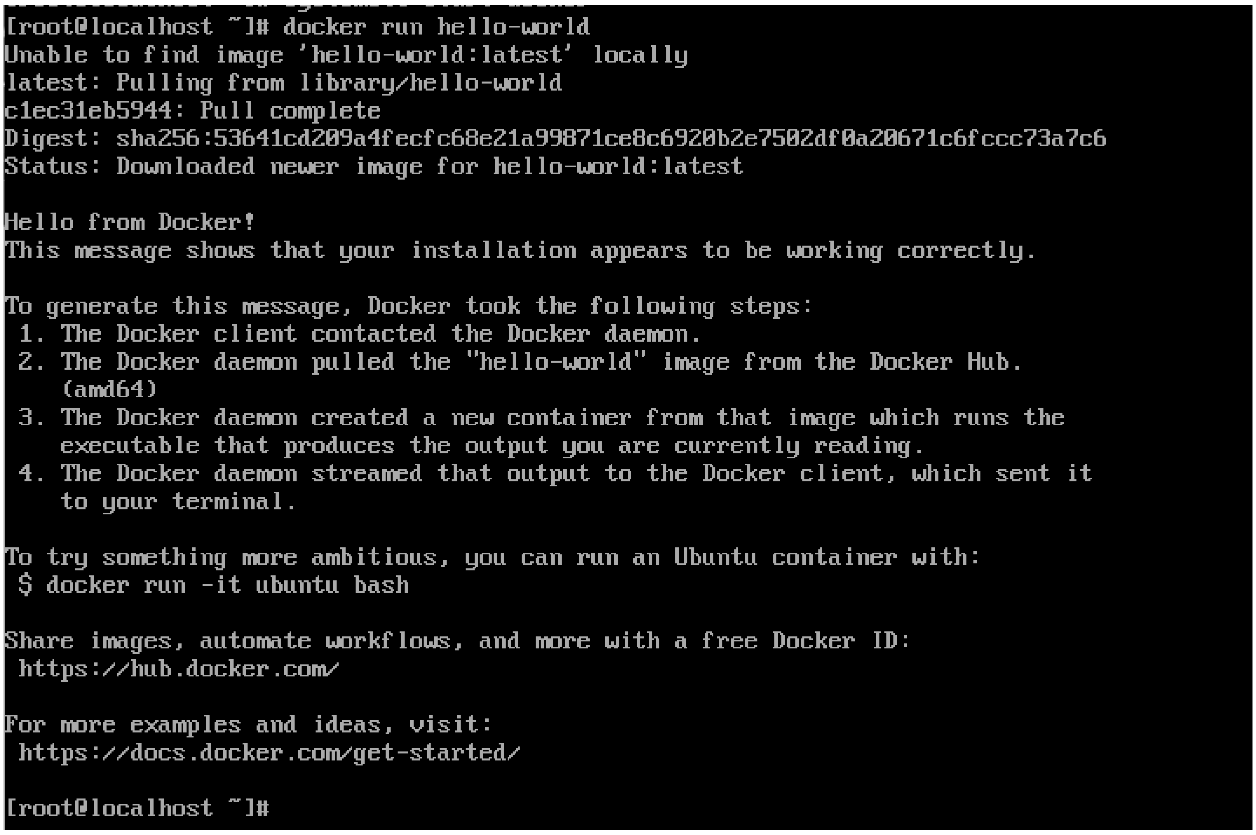
```
[root@r2 ~]# yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
```



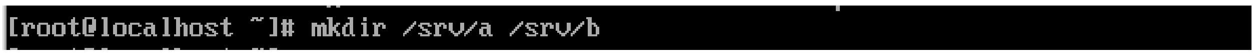
```
[root@r2 ~]# yum install docker-ce
```



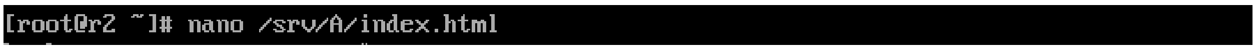
```
[root@localhost ~]# systemctl start docker
[root@localhost ~]# systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/
system/docker.service.
[root@localhost ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
   Active: active (running) since Пн 2024-03-25 14:51:26 +05; 25s ago
     Docs: http://docs.docker.com
 Main PID: 16208 (dockerd-current)
   CGroup: /system.slice/docker.service
           ├─16208 /usr/bin/dockerd-current --add-runtime docker-runc=/usr/libexec/docker/docker-...
           └─16213 /usr/bin/docker-containerd-current -l unix:///var/run/docker/libcontainerd/doc...

мар 25 14:51:23 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:23.20303..."
мар 25 14:51:24 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:24.73483..."
мар 25 14:51:24 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:24.73986..."
мар 25 14:51:24 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:24.80082..."
мар 25 14:51:25 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:25.61467..."
мар 25 14:51:26 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:26.18548..."
мар 25 14:51:26 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:26.25476..."
мар 25 14:51:26 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:26.25517...1
мар 25 14:51:26 localhost.localdomain systemd[1]: Started Docker Application Container Engine.
мар 25 14:51:26 localhost.localdomain dockerd-current[16208]: time="2024-03-25T14:51:26.28177..."
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost ~]#
```



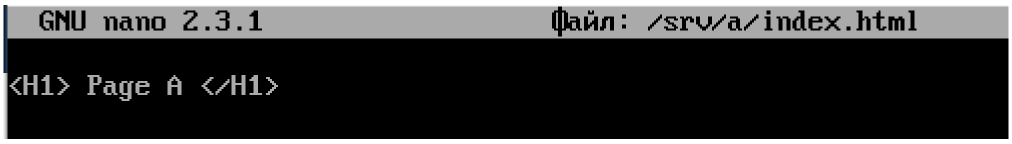
```
[root@localhost ~]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:53641cd209a4fecfc68e21a99871ce8c6920b2e7502df0a20671c6fccc73a7c6
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

[root@localhost ~]#
```



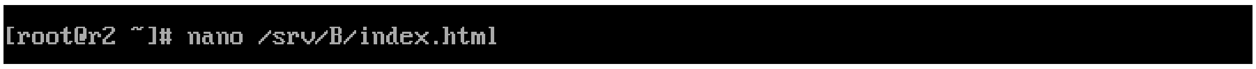
```
[root@localhost ~]# mkdir /srv/a /srv/b
```



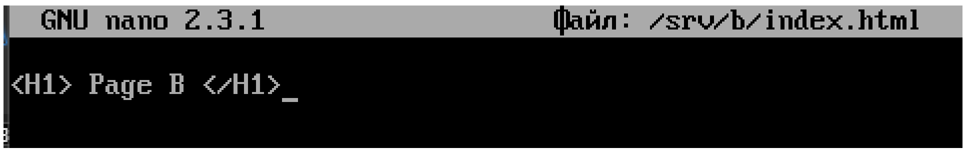
```
[root@r2 ~]# nano /srv/A/index.html
```



```
  GNU nano 2.3.1                    Файл: /srv/a/index.html

<H1> Page A </H1>
```

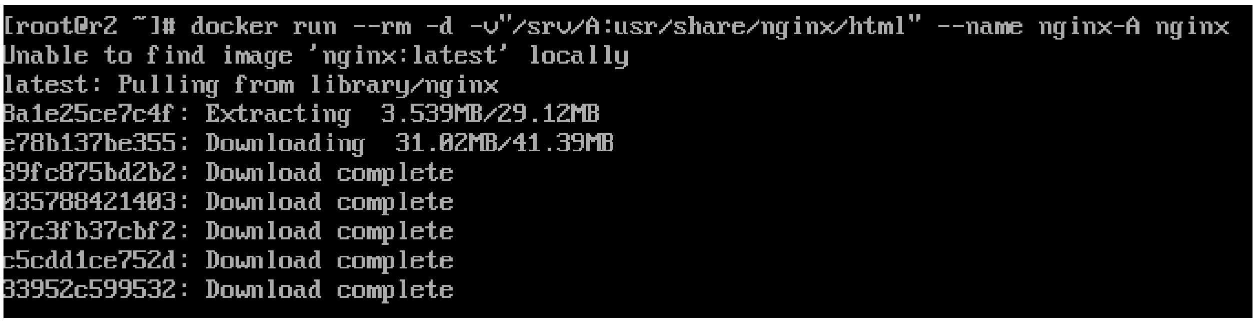


```
[root@r2 ~]# nano /srv/B/index.html
```



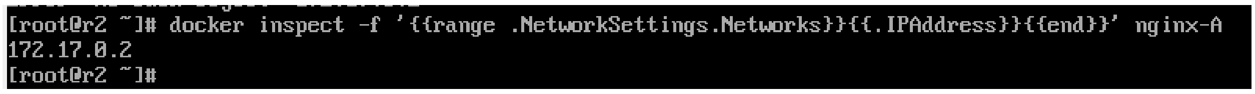
```
  GNU nano 2.3.1                    Файл: /srv/b/index.html

<H1> Page B </H1>_
```

docker run --rm -d -v "/srv/A:/usr/share/nginx/html" --name nginx-A nginx

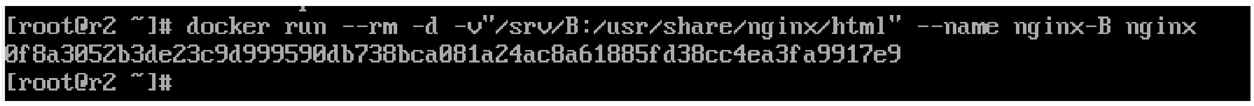


```
[root@r2 ~]# docker run --rm -d -v"/srv/A:usr/share/nginx/html" --name nginx-A nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
8a1e25ce7c4f: Extracting  3.539MB/29.12MB
e78b137be355: Downloading  31.02MB/41.39MB
39fc875bd2b2: Download complete
035788421403: Download complete
87c3fb37cbf2: Download complete
c5cdd1ce752d: Download complete
33952c599532: Download complete
```

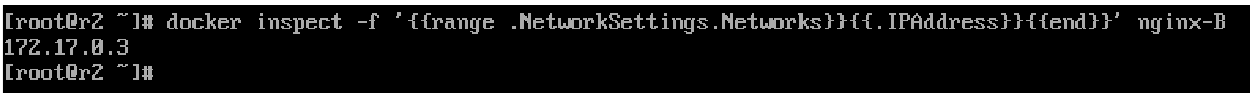
docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' nginx-A



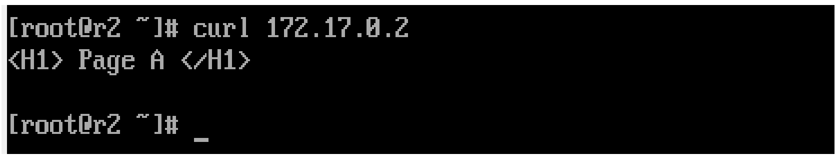
```
[root@r2 ~]# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' nginx-A
172.17.0.2
[root@r2 ~]#
```



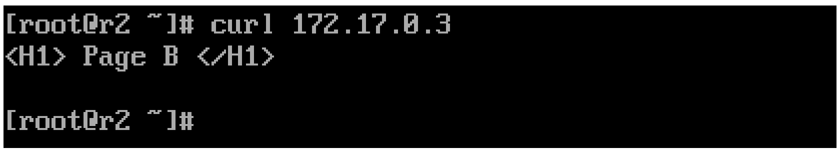
```
[root@r2 ~]# docker run --rm -d -v"/srv/B:/usr/share/nginx/html" --name nginx-B nginx
0f8a3052b3de23c9d999590db738bca081a24ac8a61885fd38cc4ea3fa9917e9
[root@r2 ~]#
```



```
[root@r2 ~]# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' nginx-B
172.17.0.3
[root@r2 ~]#
```



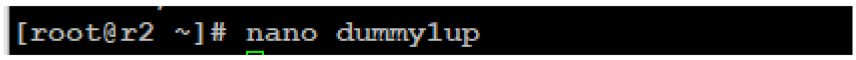
```
[root@r2 ~]# curl 172.17.0.2
<H1> Page A </H1>

[root@r2 ~]# _
```



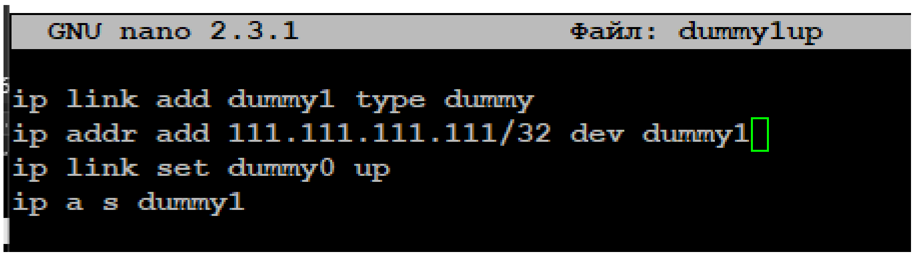
```
[root@r2 ~]# curl 172.17.0.3
<H1> Page B </H1>

[root@r2 ~]#
```

Если не работает, то нужно настроить selinux или отключить.

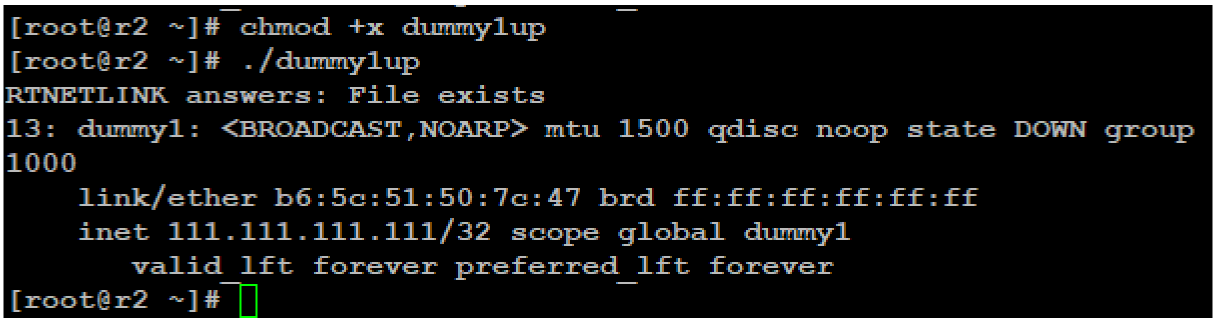
Создадим еще один dummy-интерфейс:



```
[root@r2 ~]# nano dummy1up
```

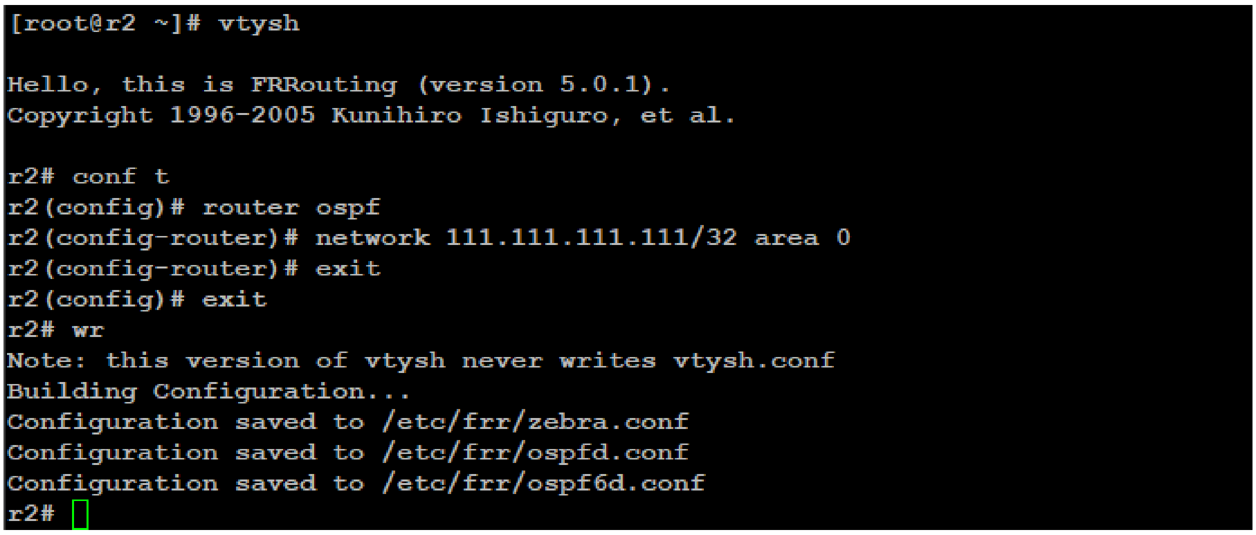


```
  GNU nano 2.3.1                    Файл: dummy1up

ip link add dummy1 type dummy
ip addr add 111.111.111.111/32 dev dummy1
ip link set dummy0 up
ip a s dummy1
```



```
[root@r2 ~]# chmod +x dummy1up
[root@r2 ~]# ./dummy1up
RTNETLINK answers: File exists
13: dummy1: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN group
1000
    link/ether b6:5c:51:50:7c:47 brd ff:ff:ff:ff:ff:ff
    inet 111.111.111.111/32 scope global dummy1
       valid_lft forever preferred_lft forever
[root@r2 ~]#
```

Анонсирум сеть 111.111.111.111 с помощью ospf:

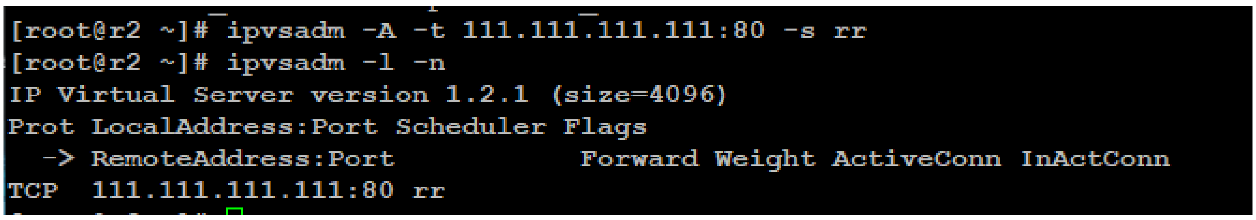


```
[root@r2 ~]# vtysh

Hello, this is FRRouting (version 5.0.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

r2# conf t
r2(config)# router ospf
r2(config-router)# network 111.111.111.111/32 area 0
r2(config-router)# exit
r2(config)# exit
r2# wr
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Configuration saved to /etc/frr/zebra.conf
Configuration saved to /etc/frr/ospfd.conf
Configuration saved to /etc/frr/ospf6d.conf
r2#
```

Создаем новый IPVS-сервис. Для теста используем ip-адрес созданного dummy-интерфейса:

ipvsadm -A -t 111.111.111.111:80 -s rr

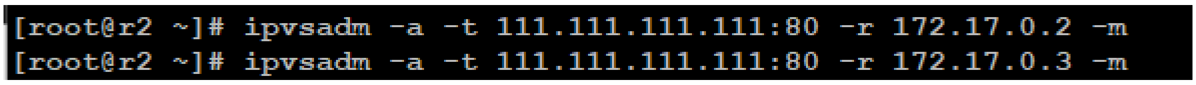
ipvsadm -l -n



```
[root@r2 ~]# ipvsadm -A -t 111.111.111.111:80 -s rr
[root@r2 ~]# 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
```

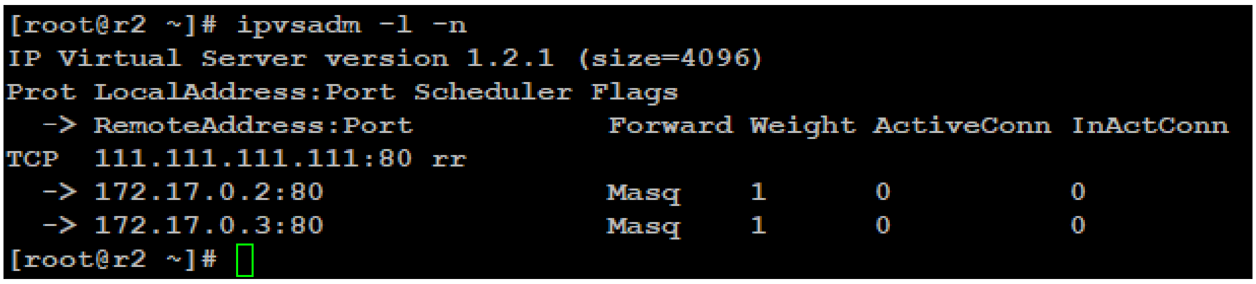
Добавляем реалы:

ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.2 -m



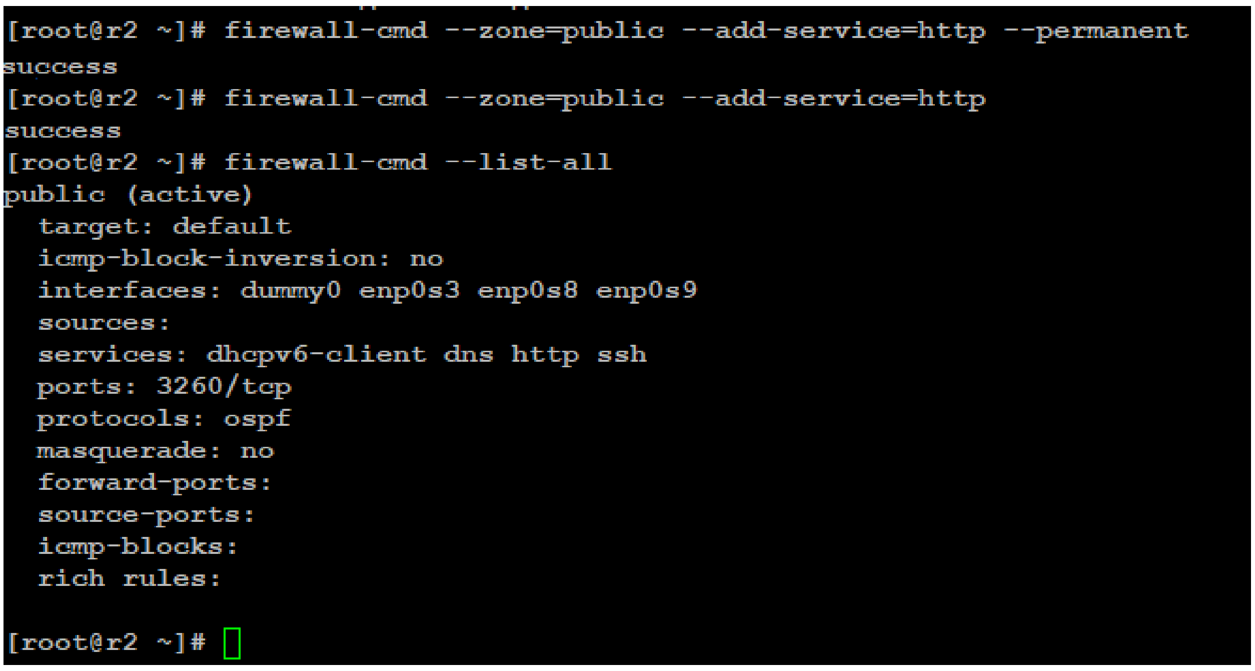
```
[root@r2 ~]# ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.2 -m
[root@r2 ~]# ipvsadm -a -t 111.111.111.111:80 -r 172.17.0.3 -m
```

И проверяем:



```
[root@r2 ~]# 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
[root@r2 ~]#
```

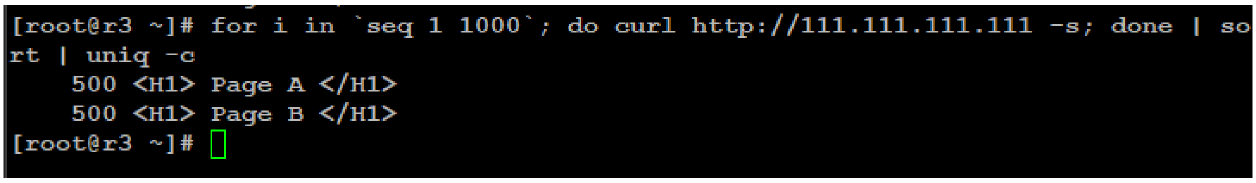
Открываем порт 80 на файрволле:



```
[root@r2 ~]# firewall-cmd --zone=public --add-service=http --permanent
success
[root@r2 ~]# firewall-cmd --zone=public --add-service=http
success
[root@r2 ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: dummy0 enp0s3 enp0s8 enp0s9
  sources:
  services: dhcpv6-client dns http ssh
  ports: 3260/tcp
  protocols: ospf
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:

[root@r2 ~]#
```

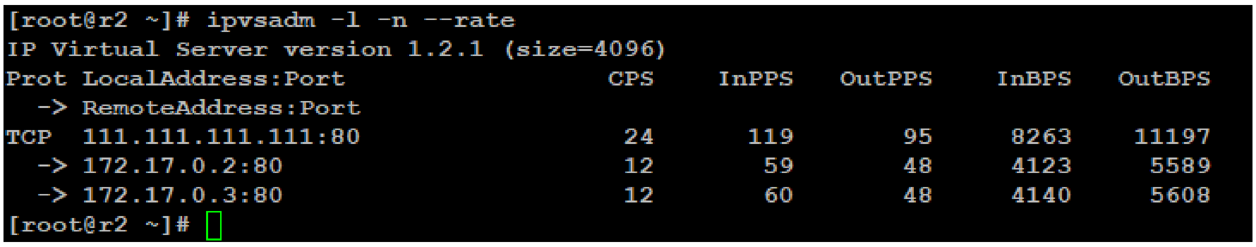
На хосте **r3** проверяем балансировку нагрузки отправив 1000 запросов:

for i in `seq 1 1000`; do curl http://111.111.111.111 -s; done | sort | uniq -c



```
[root@r3 ~]# for i in `seq 1 1000`; do curl http://111.111.111.111 -s; done | so
rt | uniq -c
    500 <H1> Page A </H1>
    500 <H1> Page B </H1>
[root@r3 ~]#
```

Теперь посмотрим статистику на сервере:



```
[root@r2 ~]# 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                 24      119       95     8263    11197
  -> 172.17.0.2:80                      12       59       48     4123     5589
  -> 172.17.0.3:80                      12       60       48     4140     5608
[root@r2 ~]#
```

Балансировка работает.

Задание:

1.

2. В браузере откройте админ. панель.

3. *Создаите домен вида: catec<номер варианта>.kz

4. *Создате аккаунт вида: <ваше имя на латинице>@<домен>

5. *Проверьте работоспособность почтового сервера.