

Построение кластера

Задание 0. Построение стенда

Схема виртуального лабораторного стенда

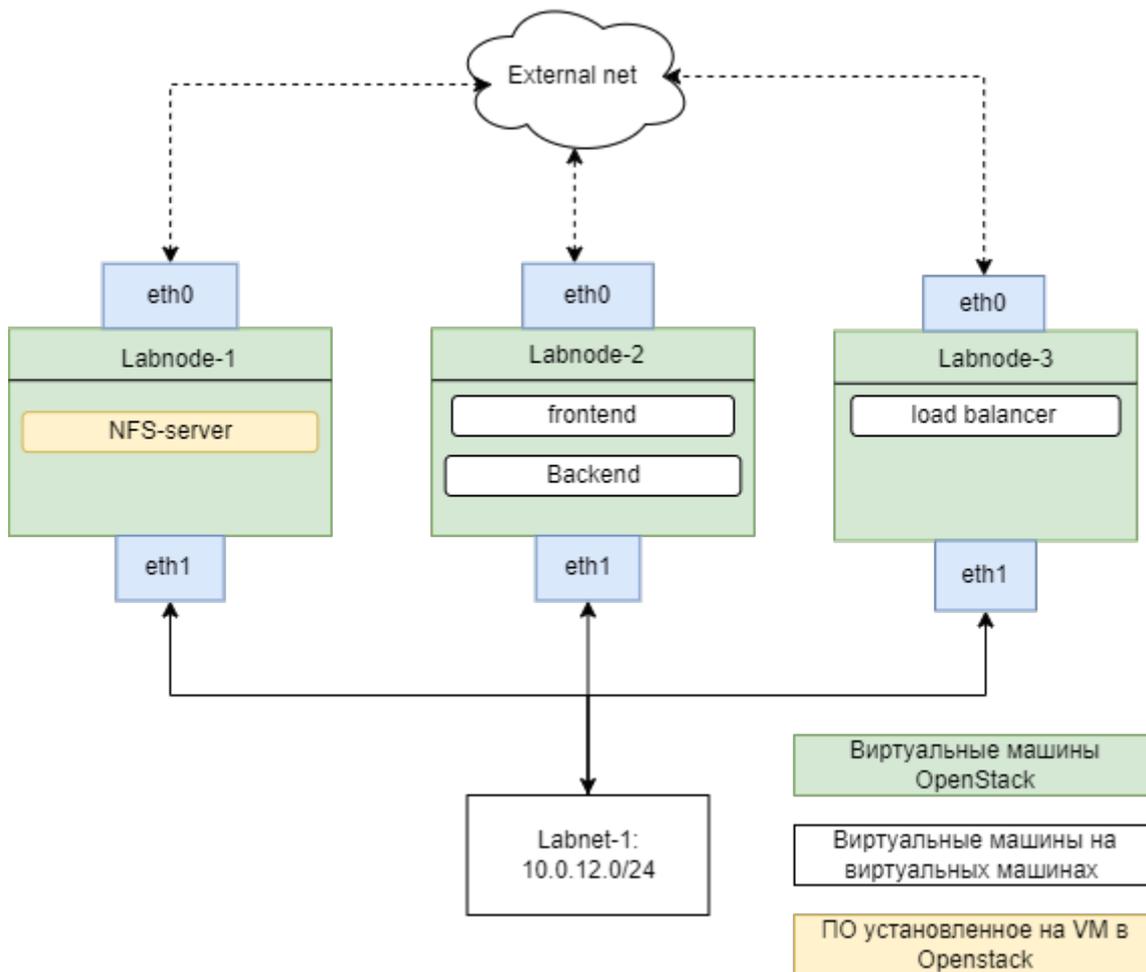


Рисунок 1. Схема стенда

1. Создать виртуальную сеть:

1. labnet

Название сети/подсети	Сетевой адрес	mtu
labnet	10.0.12.0/24	1481

При создании подсети необходимо выбрать пункт "Запретить шлюз"

2. Создать виртуальные машины для работы

Название виртуальной машины	Источник	Тип инстанса	Сети для внешнего подключения	Размер диска	Размер доп диска	ip в сети labnet
Labnode-1	Образ-Ubuntu-server20.04	small	external-net	15GB	75gb	10.0.12.21
Labnode-2	Образ-Ubuntu-server20.04	small	external-net	15GB	-	10.0.12.22
Labnode-3	Образ-Ubuntu-server20.04	small	external-net	15GB	-	10.0.12.23

Задание 1. Настройка NFS

На labnode-1 необходимо настроить NFS сервер и использовать его на всех vm с точкой монтирования nfs `/media/nfs_share/`

Обратите внимание на fstab на узле labnode-1

Задание 2. Установка Pacemaker и Corosync

Установка очень проста. На **всех** узлах нужно выполнить команду:

```
sudo apt install -y pacemaker corosync pcs resource-agents qemu-kvm qemu-system qemu-utils qemu-kvm virt-manager libvirt-daemon-system virtinst libvirt-clients bridge-utils
```

Далее поднять pcs. Тоже, на всех узлах:

```
sudo systemctl start pcsd
sudo systemctl enable pcsd
```

Для обращения к узлам по имени, а не по адресу удобнее прописать на **всех узлах** сопоставление ip адреса и его имени. В таком случае, для сетевого взаимодействия между узлами можно будет обращаться по его имени. Для того чтобы прописать это соответствие, необходимо открыть файл **/etc/hosts**:

```
sudo vi /etc/hosts
```

Прописать в нем следующее:

```
10.0.12.21 labnode-1 labnode-1.novalocal
10.0.12.22 labnode-2 labnode-2.novalocal
10.0.12.23 labnode-3 labnode-3.novalocal
```

Задайте пользователю **hacluster** пароль на всех узлах(сам пользователь был автоматически создан в процессе установки pacemaker).

```
echo password | sudo passwd --stdin hacluster
```

И, с помощью pcs создать кластер (на одном из узлов):

```
sudo pcs cluster auth labnode-1 labnode-2 labnode-3 -u hacluster -p password --force
sudo pcs cluster setup --force --name labcluster labnode-1 labnode-2 labnode-3
sudo pcs cluster start --all
```

Отключить fencing (в рамках работы он не рассматривается)

```
sudo pcs property set stonith-enabled=false
```

Включить автозапуск сервисов на всех трех машинах:

```
sudo systemctl enable pacemaker corosync --now
sudo systemctl status pacemaker corosync
```

Просмотреть информацию о кластере и кворуме:

```
sudo pcs status
sudo corosync-quorumtool
```

Задание 3. Настройка моста.

На **labnode-1**, **labnode-2** и **labnode-3**. Необходимо создать bridge интерфейсы с названием br0

без NAT

Задание 4. Создание ресурса

Необходимо создать VM на labnode-1 и разместить ее на общем nfs хранилище.

Для создания VM, необходимо использовать образ ubuntu22.04 и бридж br0

Его можно получить с s3:

```
curl -L https://s3.resds.ru/itt/ubuntu22.04.iso -o /tmp/ubuntu22.04.iso
```

После этого сделать дамп конфигурации VM и сохранить с названием `ubuntu.xml`

должны быть подключены к `br0`

все `vm` должны оказаться в одном `I2` домене

Скопировать `ubuntu.xml` с **labnode-1** на **labnode-2** и **labnode-3**:

```
scp ubuntu.xml labnode-2:~  
scp ubuntu.xml labnode-3:~
```

На `labnode-1`, `labnode-2` и `labnode-3` также переместить файл в `/etc/pacemaker/`

```
sudo mv ubuntu.xml /etc/pacemaker/  
sudo chown hacluster:haclient /etc/pacemaker/ubuntu.xml
```

Теперь добавить сам ресурс:

```
sudo pcs resource create ubuntu VirtualDomain \  
config="/etc/pacemaker/ubuntu.xml" \  
migration_transport=tcp meta allow-migrate=true
```

Просмотреть список добавленных ресурсов

```
sudo pcs status  
sudo pcs resource show ubuntu
```

Проверить список виртуальных машин на узле, на котором запустился ресурс:

```
sudo virsh list --all
```

Задание 5. Настройка динамической миграции

Необходимо перейти в файл `/etc/libvirt/libvirtd.conf`

```
sudo vi /etc/libvirt/libvirtd.conf
```

Добавить туда три параметра:

```
listen_tls = 0
listen_tcp = 1
auth_tcp = "none"
```

Сохранить файл. После этого необходимо изменить файл **/etc/default/libvirt**

```
sudo vi /etc/default/libvirt
```

Добавить параметр:

```
LIBVIRT_ARGS="--listen --config /etc/libvirt/libvirt.conf"
```

Перезагрузить **libvirt**.

```
sudo systemctl mask libvirtd.socket libvirtd-ro.socket libvirtd-admin.socket libvirtd-tls.socket libvirtd-tcp.socket
sudo systemctl restart libvirtd
```

Проделать эти операции на всех узлах.

Задание 6. Миграция ресурса

Нужно переместить ресурс на **labnode-2**:

```
sudo pcs resource move ubuntu labnode-2
```

На labnode-2 посмотреть статус кластера, и проверить список запущенных гостевых машин можно следующими командами:

```
sudo pcs status
sudo virsh list --all
```

Команда move добавляет ресурсу правило, заставляющее его запускаться только на указанном узле. Для того чтобы очистить все добавленные ограничения - clear:

```
sudo pcs resource clear ubuntu
```

Необходимо дождаться загрузки VM. Переместить ресурс на labnode-1:

```
sudo pcs resource move ubuntu labnode-1
```

Посмотреть на результат:

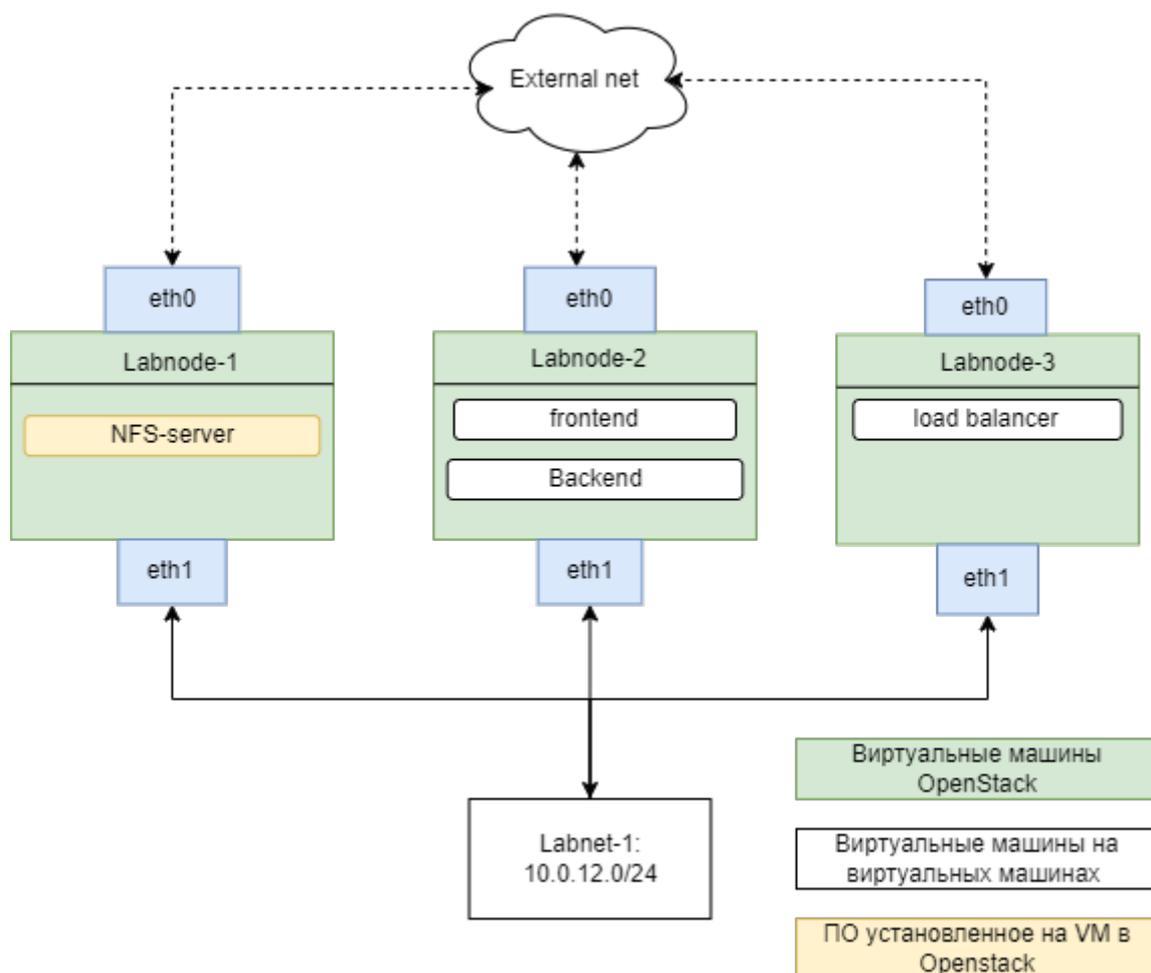
```
sudo pcs status
sudo virsh list --all
```

Задание 7. Развертывание вложенной инфраструктуры

Аналогичным образом как было описано ранее, необходимо дополнительно развернуть виртуальные машины и привести к виду

Параметры VM, аналогичны созданной ранее VM(ubuntu)

Все созданные виртуальные машины должны находится в кластере



Задание 8. Развертывание frontend

Подключаемся на созданную ранее виртуальную машину на узле *Labnode-2*

Устанавливаем утилиту для работы с системой контроля версий `GIT`

```
sudo apt install -y git
```

Склонируем с помощью Git публичный репозиторий и перейдем в нее

```
git clone https://gitlab.resds.ru/itt/sample-front.git
cd sample-front
```

Установим NodeJS-18

```
# Добавляем репозиторий
curl -s https://deb.nodesource.com/setup_18.x | sudo bash
# Устанавливаем пакет
sudo apt install nodejs -y
# Проверяем установку
node -v
```

Устанавливаем пакеты node-js

```
npm ci
```

Запускаем проект

```
npm start
```

Проверьте запуск веб приложения используя браузер

Задание 9. Развертывание backend

1. Необходимо на виртуальной машине backend скопировать репозиторий `https://gitlab.resds.ru/itt/sample-back.git`
2. Установить Nodejs 18
3. Установить все используемые пакеты
4. Запустить проект с помощью команды

```
npm run
```

5. Проверить работоспособность backend Для выполнения проверки работоспособности можно выполнить команду, для добавления заметки и просмотра списка заметок
6. Добавление заметки

```
curl --location --request POST 'http://127.0.0.1:5000/api/notes' \
--header 'Content-Type: application/json' \
--data-raw '{
  "index": "1",
  "note": "test message"
}
```

```
}'
```

2. Проверка списка заметок

```
curl --location --request GET 'http://127.0.0.1:5000/api/notes'
```

Задание 10. Развертывание NGINX

1. Установить веб-сервер NGINX

```
sudo apt install nginx
```

2. Изменяем конфигурацию NGINX `/etc/nginx/conf.d/defaultl.conf`

```
upstream front {
    server notes-app:3000;
}

upstream back {
    server notes-back:5000;
}

server {
    listen 80;
    location / {
        proxy_pass http://front;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Real-IP $remote_addr;
    }

    location /api {
        proxy_pass http://back/api;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

Можно произвести проверку синтаксиса с помощью команды

```
nginx -c /etc/nginx/conf.d/default.conf
```

в конфигурации необходимо изменить `notes-app` и `notes-back` на адреса VM

3. Проверьте работы приложения при миграции виртуальных машин и выключение хостовой виртуальной машины

Версия #16

Тарабанов Илья Федорович создал 10 января 2024 13:15:20

Тарабанов Илья Федорович обновил 16 мая 2024 19:14:49