

Инструкция развертывания трехузлового DevStack

Версия документа: 1.10

Захарченко Дмитрий, Роман Андреев

March 3, 2025

Contents

1	Развернуть виртуальный кластерный DevStack	1
1.1	Стенд	2
1.1.1	Пользователи и пароли	2
1.1.2	Узлы стенда	2
1.1.3	Конфигурация ВМ	2
1.2	Конфигурация кластера	2
1.2.1	Конфигурация сети	2
1.2.2	Конфигурация пользователя и вход	4
1.2.3	Установка DevStack	4
1.2.4	Конфигурация local.conf	4
1.2.5	Запуск devstack	6
1.2.6	Обнаружение compute узлов контроллером	6
1.2.7	Формирование файлов compute_id	6
1.2.8	Сброс кластера	7
1.3	Настройка Engine	8
1.3.1	Подготовка группы для подключения по iscsi	8
1.3.2	Подготовка группы для подключения по fibre-channel	8
1.3.3	Включение поддержки тонких томов	9
1.4	Настройка драйвера cinder	9
1.4.1	Загрузка драйвера на узлы (tar)	9
1.4.2	Редактирование конфигурации cinder.conf	9
1.5	Дополнительно	12
1.5.1	Патчи	12
1.5.2	logrotate	12
1.6	Иллюстрации к настройке Devstack	13

1 Развернуть виртуальный кластерный DevStack

1.1 Стенд

1.1.1 Пользователи и пароли

Фронтенд	Имя пользователя	Пароль
Hypervisor	admin	password
Guest OS	devstack	devstack

1.1.2 Узлы стенда

№ Ноды	Имя хоста	Адрес гипервизора	Адрес в ВМ	Интерфейс в ВМ
1	devstack1	192.168.5.121	192.168.5.64	ens3
2	devstack2	192.168.5.121	192.168.5.65	ens3
3	devstack3	192.168.5.121	192.168.5.60	ens3

1.1.3 Конфигурация ВМ

Данный раздел добавлен по причине проблем при установке openstack на ВМ. Процесс запуска openstack намертво зависал, при этом никаких внятных логов найдено не было. Причиной оказалась сильная нехватка памяти. Помимо памяти были добавлены ядра процессору и потоки. В конечном итоге виртуальный стенд на котором удалось запустить openstack выглядит следующим образом:

Имя виртуальной машины	RAM	Ядра CPU	Потоки	Диска	Сеть	Имя ОС
DEVSTACK_CNTRL_NODE	32	8	8	50G	nmap5	Ubuntu-22.04.5
DEVSTACK_COMPUTE_NODE_1	32	4	4	60G	nmap5	Ubuntu-22.04.5
DEVSTACK_COMPUTE_NODE_2	32	4	4	100G	nmap5	Ubuntu-22.04.5

ИНФО: образ на шаре: /mnt/qwe/ISO/Ubuntu/ubuntu-22.04.5-live-server-amd64.iso

ИНФО: образ на vair: ISO_UBUNTU_22_04_5_SERVER_ISO

ИНФО: По данной инструкции был установлен и запущен openstack с версией: 6.6.1

ИНФО: Для сборки devstack использовался [официальный репозитрий](#) с основной веткой main

ИНФО: В качестве СХД использовался engine с версией микрокода A-CORE 5.4.1 1724062870

1.2 Конфигурация кластера

Конфигурация выполнялась по [официальной документации](#) с небольшими отклонениями.

1.2.1 Конфигурация сети

Заменить конфигурацию в файле /etc/netplan/50-cloud-init.yaml.

1. Конфиг сети devstack1

```

network:
  ethernets:
    enp95s0f1:
      addresses:
        - 192.168.5.64/24
      nameservers:
        addresses:
          - 192.168.1.25
      routes:
        - to: default
          via: 192.168.5.1
  version: 2

```

2. Конфиг сети devstack2

```

network:
  ethernets:
    enp95s0f1:
      addresses:
        - 192.168.5.65/24
      nameservers:
        addresses:
          - 192.168.1.25
      routes:
        - to: default
          via: 192.168.5.1
  version: 2

```

3. Конфиг сети devstack3

```

network:
  ethernets:
    enp95s0f1:
      addresses:
        - 192.168.5.60/24
      nameservers:
        addresses:
          - 192.168.1.25
      routes:
        - to: default
          via: 192.168.5.1
  version: 2

```

4. Применяем конфигурацию Выполняем команду на каждом узле.

```

sudo netplan apply

```

1.2.2 Конфигурация пользователя и вход

```
sudo useradd -d /opt/stack -s /bin/bash -m stack
sudo chown stack:stack /opt/stack
sudo chmod 755 /opt/stack

cat <<EOF | sudo tee /etc/sudoers.d/50_stack_sh
stack ALL=(root) NOPASSWD:ALL
Defaults:stack secure_path=\
/sbin:/usr/sbin:/usr/bin:/bin:/usr/local/sbin:/usr/local/bin
Defaults:stack !requiretty
EOF

# Задаем пароль пользователю чтобы иметь
# возможность подключаться им по ssh
sudo passwd stack

# Логинимся созданным пользователем
sudo -u stack -i
```

ИНФО: Пароль пользователю назначается для того чтобы было удобно залить файлы драйвера cinder на ноды. Этот шаг не является обязательным.

ВАЖНО: Далее все действия выполняются из под пользователя stack.

1.2.3 Установка DevStack

```
git clone https://opendev.org/openstack/devstack
cd devstack
```

1.2.4 Конфигурация local.conf

На каждом узле создаем файл /opt/stack/devstack/local.conf и заполняем его:

1. devstack3 (controller)

```
[[local|localrc]]
HOST_IP=192.168.5.60
FIXED_RANGE=192.168.5.0/24
FLOATING_RANGE=192.168.5.0/24

LOGFILE=/opt/stack/logs/stack.sh.log
ADMIN_PASSWORD=devstack
DATABASE_PASSWORD=devstack
RABBIT_PASSWORD=devstack
SERVICE_PASSWORD=devstack

# Запрет запуска инстансов на controller ноде
disable_service n-cpu
```

2. devstack1

```
[[local|localrc]]
HOST_IP=192.168.5.64
FIXED_RANGE=192.168.5.0/24
FLOATING_RANGE=192.168.5.0/24
LOGFILE=/opt/stack/logs/stack.sh.log

ADMIN_PASSWORD=devstack
DATABASE_PASSWORD=devstack
RABBIT_PASSWORD=devstack
SERVICE_PASSWORD=devstack

DATABASE_TYPE=mysql
SERVICE_HOST=192.168.5.60
MYSQL_HOST=$SERVICE_HOST
RABBIT_HOST=$SERVICE_HOST
GLANCE_HOSTPORT=$SERVICE_HOST:9292
ENABLED_SERVICES=n-cpu,c-vol,placement-client,ovn-controller,
ovs-vswitchd,ovsdb-server,q-ovn-metadata-agent

NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://$SERVICE_HOST:6080/vnc_lite.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN
```

ИНФО: SERVICE_HOST - Адрес узла-контроллера

ВАЖНО: Значение ENABLED_SERVICES должно быть записано в одну строку.

3. devstack2

```
[[local|localrc]]
HOST_IP=192.168.5.65
FIXED_RANGE=192.168.5.0/24
FLOATING_RANGE=192.168.5.0/24
LOGFILE=/opt/stack/logs/stack.sh.log

ADMIN_PASSWORD=devstack
DATABASE_PASSWORD=devstack
RABBIT_PASSWORD=devstack
SERVICE_PASSWORD=devstack

DATABASE_TYPE=mysql
SERVICE_HOST=192.168.5.60
MYSQL_HOST=$SERVICE_HOST
RABBIT_HOST=$SERVICE_HOST
GLANCE_HOSTPORT=$SERVICE_HOST:9292
ENABLED_SERVICES=n-cpu,c-vol,placement-client,ovn-controller,
ovs-vswitchd,ovsdb-server,q-ovn-metadata-agent
```

```
NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://$SERVICE_HOST:6080/vnc_lite.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN
```

ИНФО: `SERVICE_HOST` - Адрес узла-контроллера

ВАЖНО: Значение `ENABLED_SERVICES` должно быть записано в одну строку.

1.2.5 Запуск devstack

На всех нодах, начиная с controller ноды выполняем.

Параллельный запуск compute нод с контроллером не проверялся, поэтому рекомендую дождаться запуска controller ноды и после запустить обе compute ноды.

```
/opt/stack/devstack/stack.sh
```

1.2.6 Обнаружение compute узлов контроллером

После того как обе compute ноды запустились, убеждаемся что они выводятся на вебке на вкладке “[Compute Host](#)”. Если все ноды отображаются можно запускать `discover` на controller ноды. Ниже приведены скриншоты того что должно отображаться на веб-интерфейсе перед запуском `discover_hosts.sh`. Смотри иллюстрации рис.1, рис.2 и рис.3.

```
/opt/stack/devstack/tools/discover_hosts.sh
```

ВАЖНО: Если вы выполняете повторную сборку compute узлов без пересборки controller ноды, выполнение запуска `discover_hosts.sh` необходимо пропустить и выполнить следующий шаг по формированию файлов идентификаторов для узлов.

1.2.7 Формирование файлов compute_id

После выполнения дискавери хосты будут добавлены в БД что можно увидеть на рис.4, но этого не достаточно, идентификаторы этих хостов необходимо сохранить в файл на каждом из compute узлов. То есть на каждом compute узле должен быть файл `/etc/nova/compute_id` который содержит идентификатор этого узла.

Ниже приведен листинг как сформировать файлы `compute_id` на compute узлах. Код выполняется на controller узле. Перед выполнением замените значения переменных `compute_host` и `compute_addr` согласно конфигурации вашего кластера. Иллюстрация рис.4.

```
# Формирование файла для первой compute ноды
compute_host='devstack1'
compute_addr='192.168.5.64'
query=$(echo "SELECT uuid FROM compute_nodes WHERE host =
↪  '${compute_host}';")
compute_id=$(mysql nova_cell1 -e "${query}" --batch --skip-column-names)
ssh $compute_addr "echo $compute_id | sudo tee /etc/nova/compute_id"
```

```
# Формирование файла для второй compute ноды
compute_host='devstack2'
compute_addr='192.168.5.65'
query=$(echo "SELECT uuid FROM compute_nodes WHERE host =
↳  '${compute_host}';")
compute_id=$(mysql nova_cell1 -e "${query}" --batch --skip-column-names)
ssh $compute_addr "echo $compute_id | sudo tee /etc/nova/compute_id"
```

После создания файлов `compute_id` на compute нодах уже можно создавать инстансы виртуальных машин. В качестве примера удачного создания инстанса. Иллюстрации рис.5, рис.6 и рис.7.

1.2.8 Сброс кластера

Кластер предусматривает два метода очистки `unstack.sh` и `clean.sh`. Если вы намерены выполнить `clean.sh` то выполнять `unstack.sh` не требуется, так как она вызывается внутри `clean.sh`.

```
/opt/stack/devstack/unstack.sh
# или же
/opt/stack/devstack/clean.sh
# или если уже создавались ВМ
sudo rm -rf /etc/libvirt/qemu/inst*
sudo virsh list | grep inst | awk '{print $1}' | xargs -n1 virsh destroy
```

1.3 Настройка Engine

1.3.1 Подготовка группы для подключения по iscsi

- Создать DDP группу. См. Рис.11
- Создать ip ресурс для созданной группы. См. Рис.12
- Создать target для созданного ip ресурса. См. Рис.13
- Создать группы доступа iscsi. См. Рис.16

1. Получение iqn таргета

```
cat /etc/iscsi/initiatorname.iscsi
```

1.3.2 Подготовка группы для подключения по fibre-channel

- Создать DDP группу. См. Рис.14
- Создать группы доступа fc. См. Рис.15

1. Получение wwn fc портов

```
awk '{print $0} : "substr(FILENAME, $0)}'  
↪ /sys/class/fc_host/host*/port_name
```

1.3.3 Включение поддержки тонких томов

Этот шаг необходим если планируете использовать cinder с тонкими томами. Для того чтобы конфигурация cinder с параметром `aero_thin_provisioning_support=True` необходимо создать тонкое хранилище. См. рис.17, рис.18 и рис.19.

1.4 Настройка драйвера cinder

1.4.1 Загрузка драйвера на узлы (tar)

Загрузите архив с файлами драйвера на каждый сервер в домашний каталог пользователя stack. После загрузки архива драйвера, на каждом сервере необходимо выполнить следующие команды:

```
cd / && tar xzf ~/aero-cinder.tar.gz
```

После распаковки архива в системе должен появиться каталог с файлами драйвера:

```
/opt/stack/cinder/cinder/volume/drivers/aero
```

1.4.2 Редактирование конфигурации cinder.conf

Редактируем на каждой ноде `/etc/cinder/cinder.conf`. Ниже приведен пример конфигурации который необходимо изменить согласно вашей текущей конфигурации хранилища. Описание какой параметр за что отвечает расположено под листингом конфига.

```
[DEFAULT]
...
enabled_backends = lvmdriver-1,aero-iscsi,aero-fc

[aero-iscsi]
volume_driver = cinder.volume.drivers.aero.iscsi.AeroDriverISCSI
volume_backend_name = aero-iscsi
aero_iscsi_port = 3260
aero_pool_name = cinder_iscsi
aero_pool_type = DDP
aero_hosts = 192.168.5.25,192.168.5.26
aero_protocol = https
aero_ssl_verify = False
aero_ddp_pool_raid = RAID-1
aero_ddp_pool_disk_qty = 2
aero_username = admin
aero_password = Bac12345
aero_thin_provisioning_support = True
aero_target_iqn = iqn.2014-07.ru.aerodisk:cinder
aero_force_volume_delete = True

[aero-fc]
volume_driver = cinder.volume.drivers.aero.fc.AeroDriverFC
volume_backend_name = aero-fc
aero_pool_name = cinder_fc
aero_pool_type = DDP
```

```

aero_hosts = 192.168.5.25,192.168.5.26
aero_protocol = https
aero_ssl_verify = False
aero_ddp_pool_raid = RAID-1
aero_ddp_pool_disk_qty = 2
aero_username = admin
aero_password = Bac12345
aero_thin_provisioning_support = True
aero_target_wwns =
↳ 100000109b570100,100000109b5700ff,100000109b223de7,100000109b223de8
aero_force_volume_delete = True

```

ВАЖНО: Значение `aero_target_wwns` должно быть записано в одну строку.

ИНФО: Данный файл будет удален при выполнении `unstack.sh` и `clean.sh` поэтому рекомендуется сделать его резервную копию в каталоге отличном от `/etc/cinder`.

Описание параметров для изменения `aero-iscsi`:

- `volume_backend_name`: Имя которое вы хотите использовать для определения типов через веб-интерфейс, может быть задано произвольно (без символов подчеркивания).
- `aero_pool_name`: Имя пула на СХД Engine, пул должен существовать и быть активным.
- `aero_pool_type`: Тип пула на СХД Engine
- `aero_iscsi_port`: Порт по которому доступно iscsi на хранилище Engine
- `aero_hosts`: Адреса узлов СХД Engine, разделенных запятыми
- `aero_ssl_verify`: Проверять сертификат при использовании https
- `aero_protocol`: Протокол доступа к API Engine [http/https]
- `aero_ddp_pool_raid`: Тип RAID использованный при создании пула на Engine
- `aero_ddp_pool_disk_qty`: кол-во дисков использованных при создании пула на Engine
- `aero_username`: Пользователь с доступом к API Engine
- `aero_password`: Пароль для доступа к API Engine
- `aero_thin_provisioning_support`: Поддержка хранилищем работы с тонкими томами (по умолчанию False) Для работы тонких томов необходимо сконфигурировать Engine как показано тут.
- `aero_target_iqn`: Таргет для подключения созданный на СХД Engine
- `aero_force_volume_delete`: Если установлено в True удаляет маппинг вместе с томом (по умолчанию True).

Описание параметров для изменения `aero-fc`:

- `volume_backend_name`: Имя которое вы хотите использовать для определения типов через веб-интерфейс, может быть задано произвольно без символов подчеркивания.
- `aero_pool_name`: Имя пула на СХД Engine, пул должен существовать и быть активным.

- **aero_pool_type**: Тип пула на СХД Engine
- **aero_hosts**: Адреса узлов СХД Engine, разделенных запятыми
- **aero_ssl_verify**: Проверять сертификат при использовании https
- **aero_protocol**: Протокол доступа к API Engine [http/https]
- **aero_ddp_pool_raid**: Тип RAID использованный при создании пула на Engine
- **aero_ddp_pool_disk_qty**: кол-во дисков использованных при создании пула на Engine
- **aero_username**: Пользователь с доступом к API Engine
- **aero_password**: Пароль для доступа к API Engine
- **aero_thin_provisioning_support**: Поддержка хранилищем работы с тонкими томами (по умолчанию False) Для работы тонких томов необходимо сконфигурировать Engine как показано тут.
- **aero_target_wwns**: Список wwn портов fibre-channel на СХД Engine. Порты записываются в одну строку без пробелов, разделяя запятыми.
- **aero_force_volume_delete**: Если установлено в True удаляет маппинг вместе с томом (по умолчанию True).

После того как конфигурация сформирована необходимо перезагрузить сервисы.

```
# Команда для controller узла
sudo systemctl restart devstack@c-api devstack@c-vol devstack@c-sch

# Команда для compute узлов
sudo systemctl restart devstack@c-vol
```

Для проверки включения и работы драйвера можно отслеживать логи командой:

```
sudo journalctl -fu devstack@c-vol | grep .aero.
```

Для работы с драйвером его необходимо донстроить в веб-интерфейсе. Для этого необходимо выполнить несколько шагов:

- Создать типы дисков поддерживаемые драйвером Поддерживаемые драйвером типы это **aero_iscsi** и **aero_fc** (На этом моменте название можно задавать произвольно). Иллюстрация рис.7.
- Связать с драйвером через доп. параметры типа. В качестве ключа и значения необходимо указывать значения строго из конфигурации **/etc/cinder/cinder.conf**. В нашем случае это **volume_backend_name** в качестве ключа и **aero-iscsi** или **aero-fc** в качестве значения. Иллюстрация рис.8 и рис.9

После создания и настройки типов дисков можно приступить к созданию виртуального диска. В окне создания необходимо выбрать один из созданных на предыдущем шаге типов. Иллюстрация рис.10

1.5 Дополнительно

1.5.1 Патчи

1. Патч-01

- Исправление в работе https
- Добавлена возможность указывать два контроллера в конфигурации cinder.

Изменения параметров в конфигурации cinder.conf

- Изменен `aero_host` на `aero_hosts`
- Удален `aero_port`
- Добавлен `aero_ssl_verify`

1.5.2 logrotate

В процессе эксплуатации в лог `libvirt` пишется коллосальное количество информации. Это часто приводит к переполнению диска и невозможности работать с системой. Для того чтобы этого не происходило необходимо настроить ротацию лога как показано ниже.

```
cat <<EOF | tee /etc/logrotate.d/libvirtd
/var/log/libvirt/libvirtd.log {
    daily
    missingok
    rotate 4
    compress
    copytruncate
    notifempty
    maxsize 100M
}
EOF

systemctl restart logrotate.service
```

1.6 Иллюстрации к настройке Devstack

В данном разделе собраны иллюстрации к инструкции.

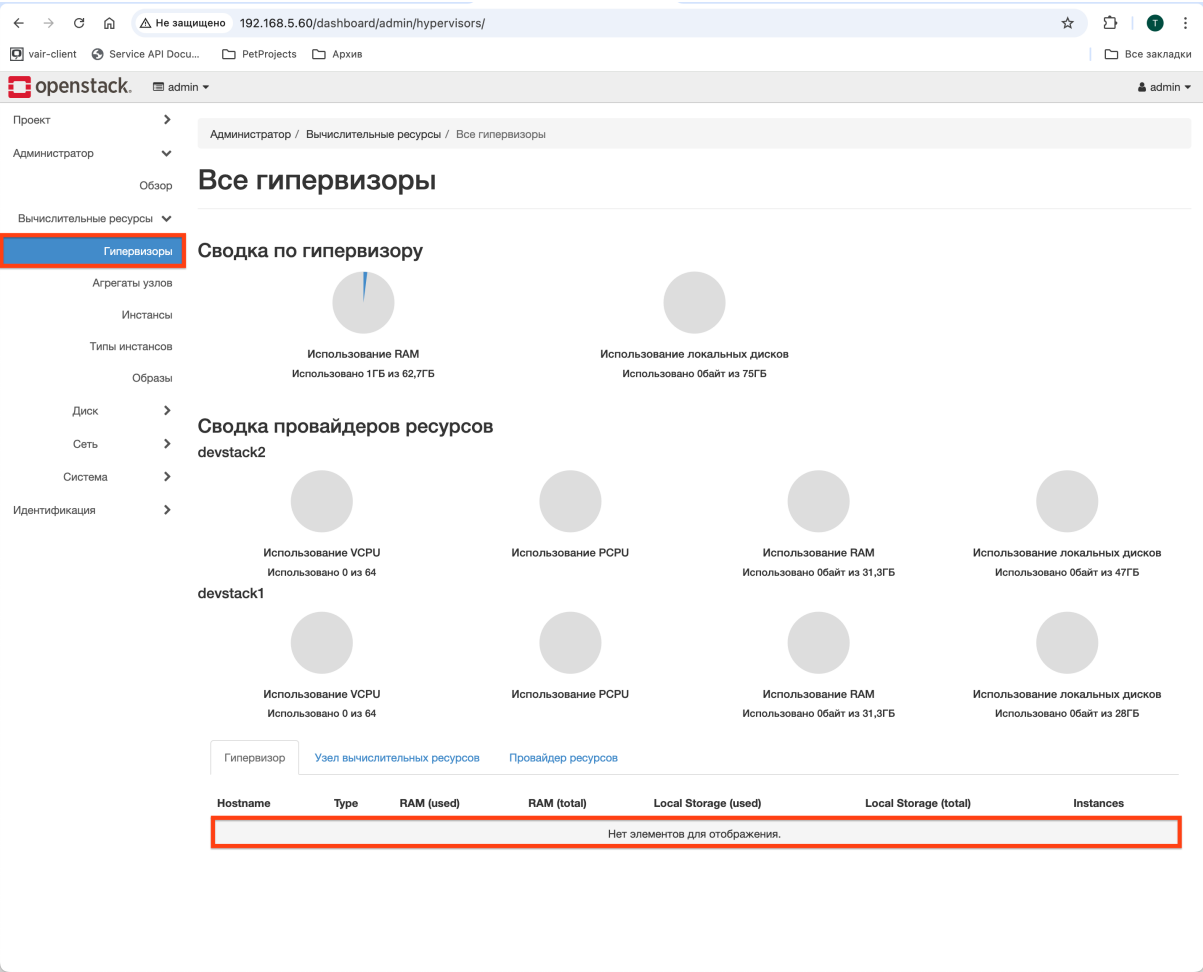


Рис.1: Гипервизоры отсутствуют

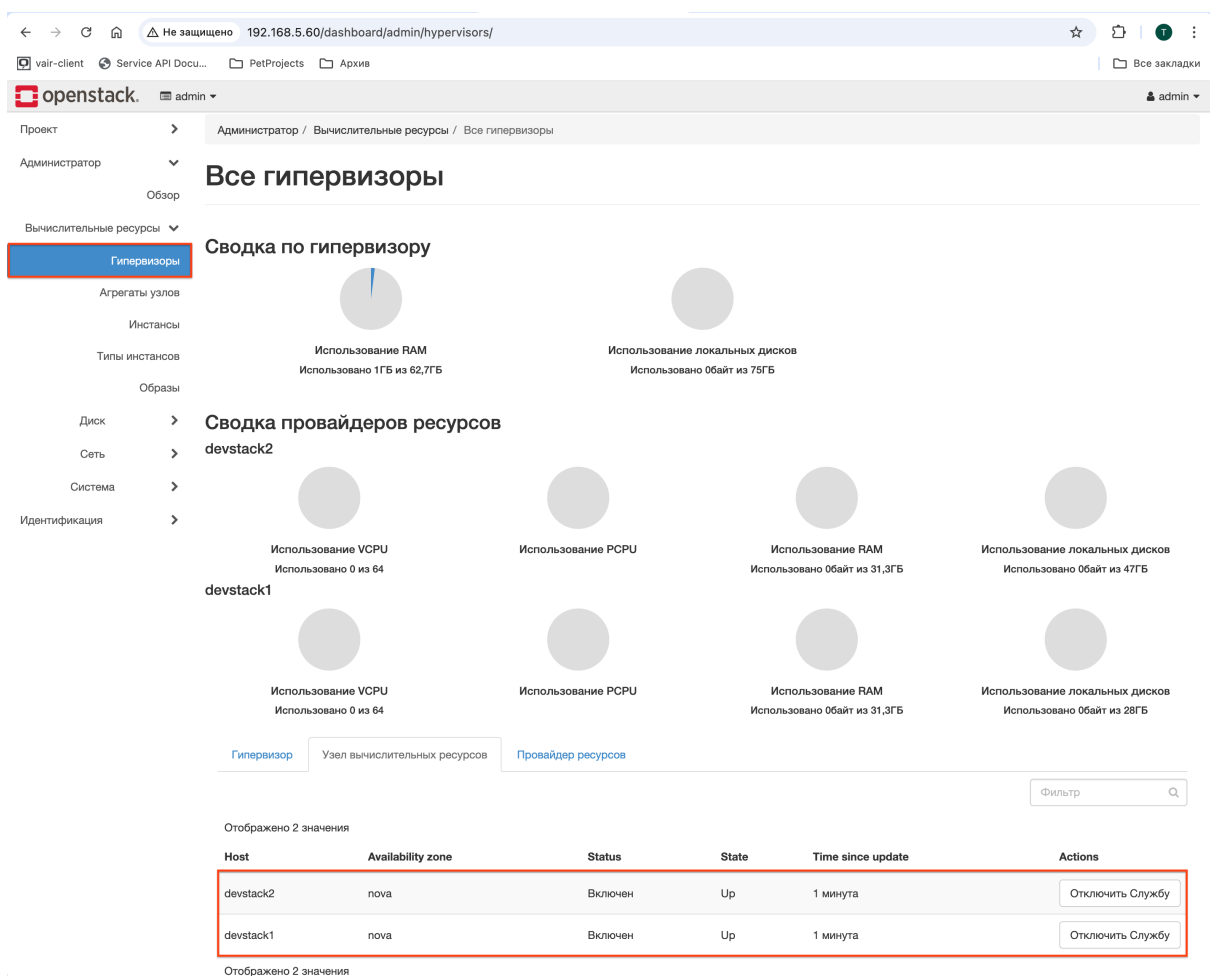


Рис.2: Вычислительные ресурсы присутствуют

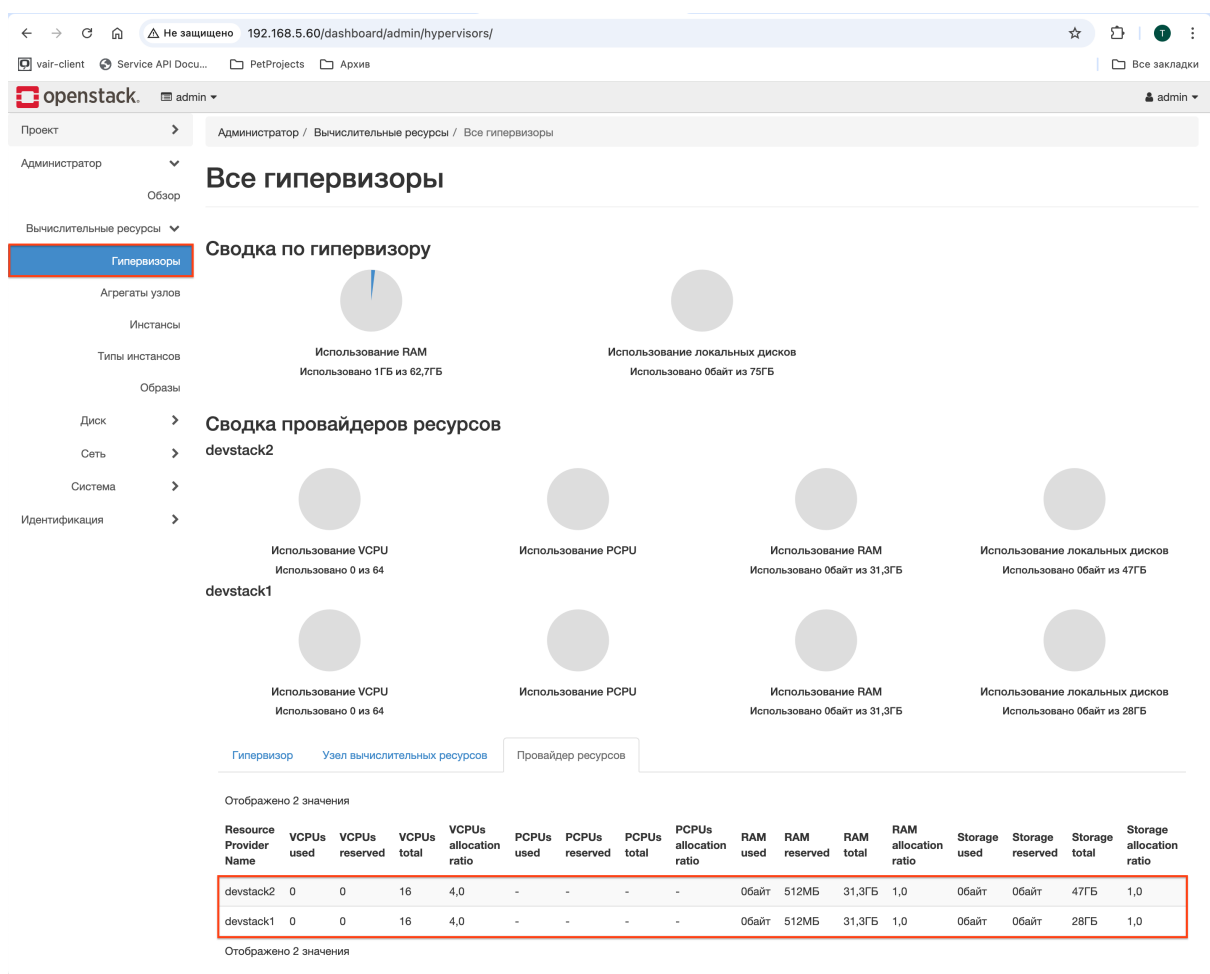


Рис.3: Провайдеры ресурсов присутствуют

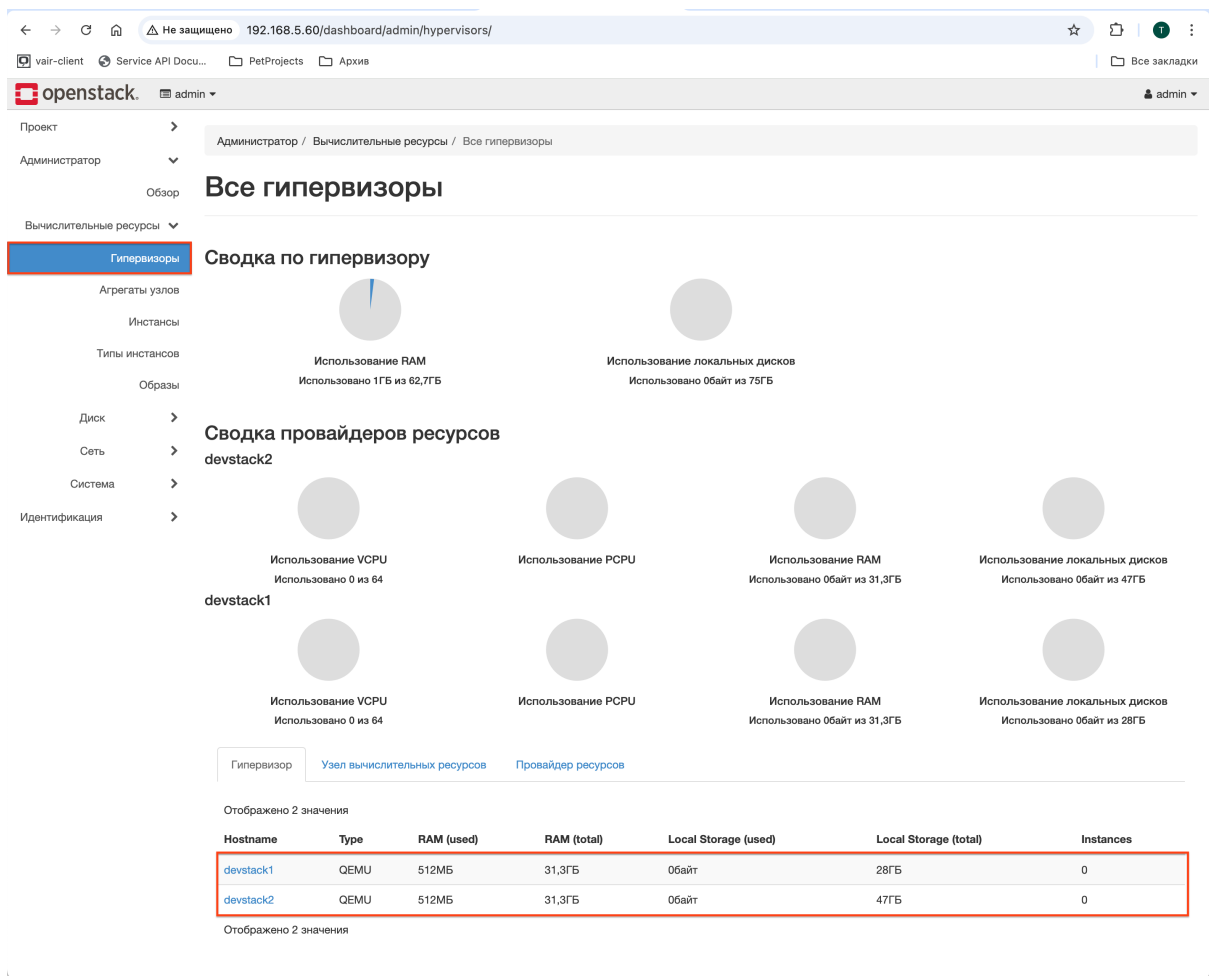


Рис.4: Гипервизоры появились

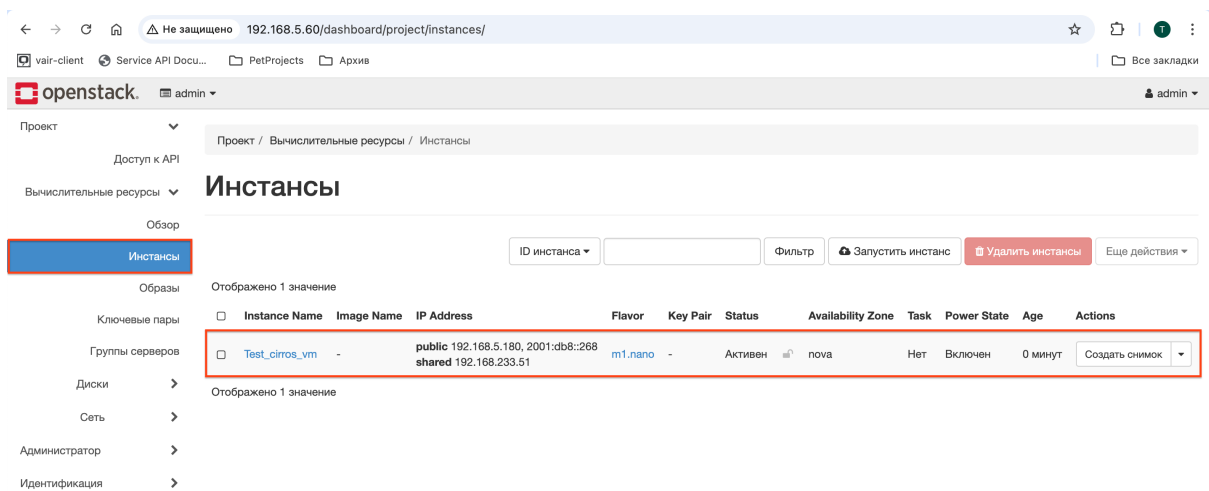


Рис.5: Пример удачно созданного инстанса 1

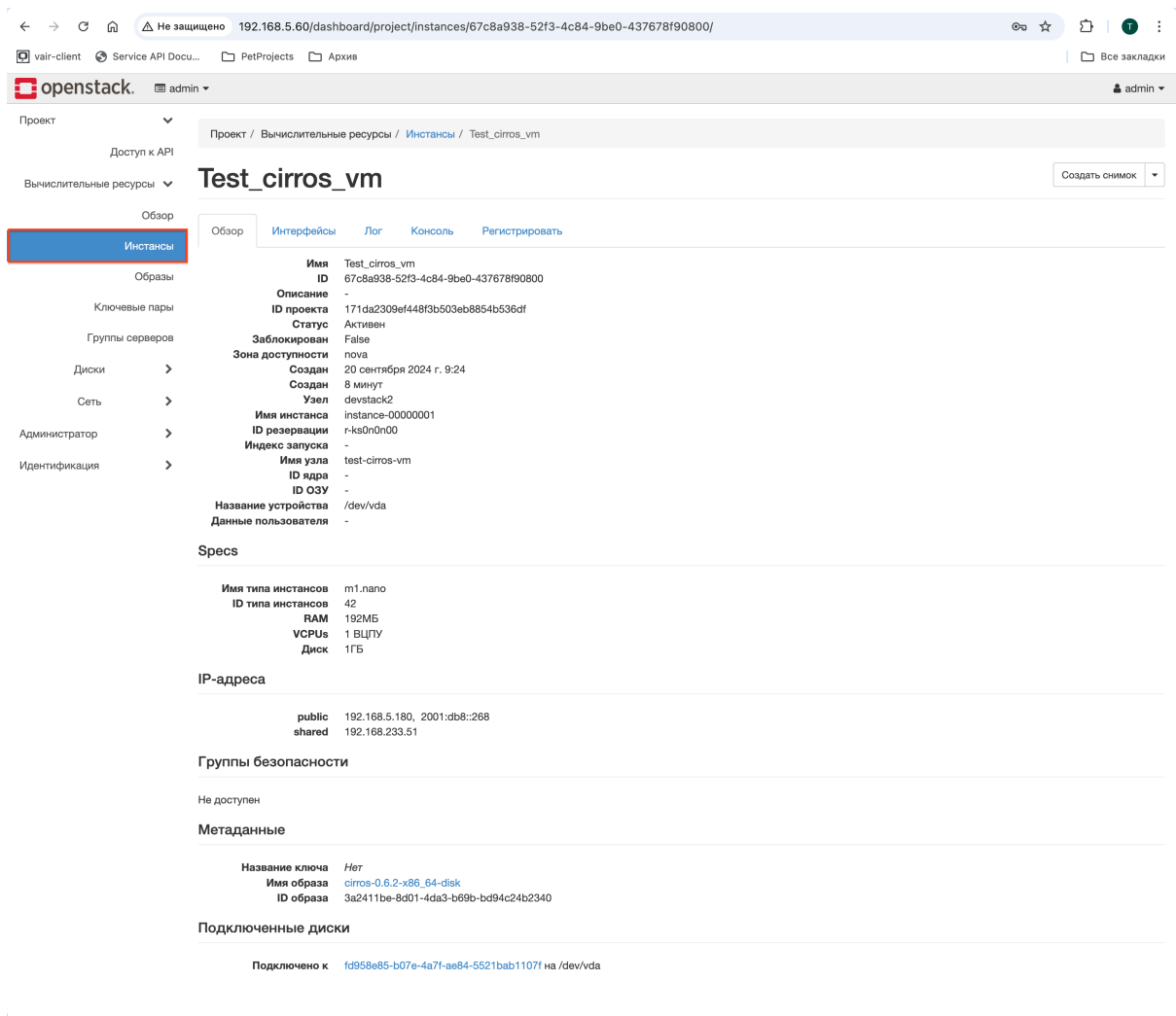


Рис.6: Пример удачно созданного инстанса 2

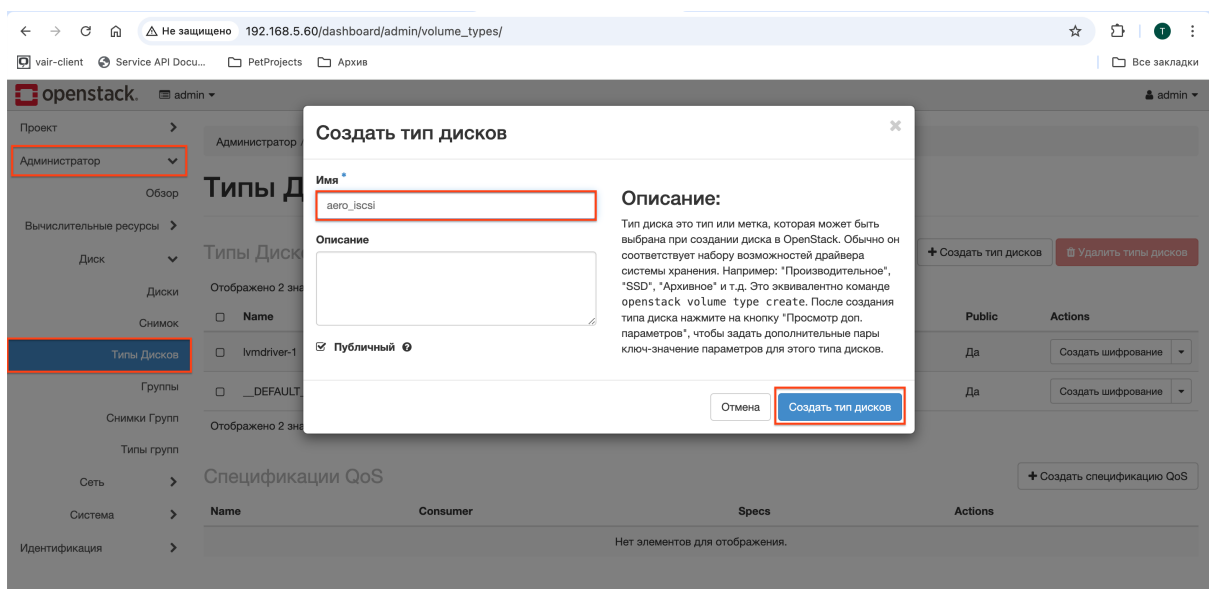


Рис.7: Создание типа диска

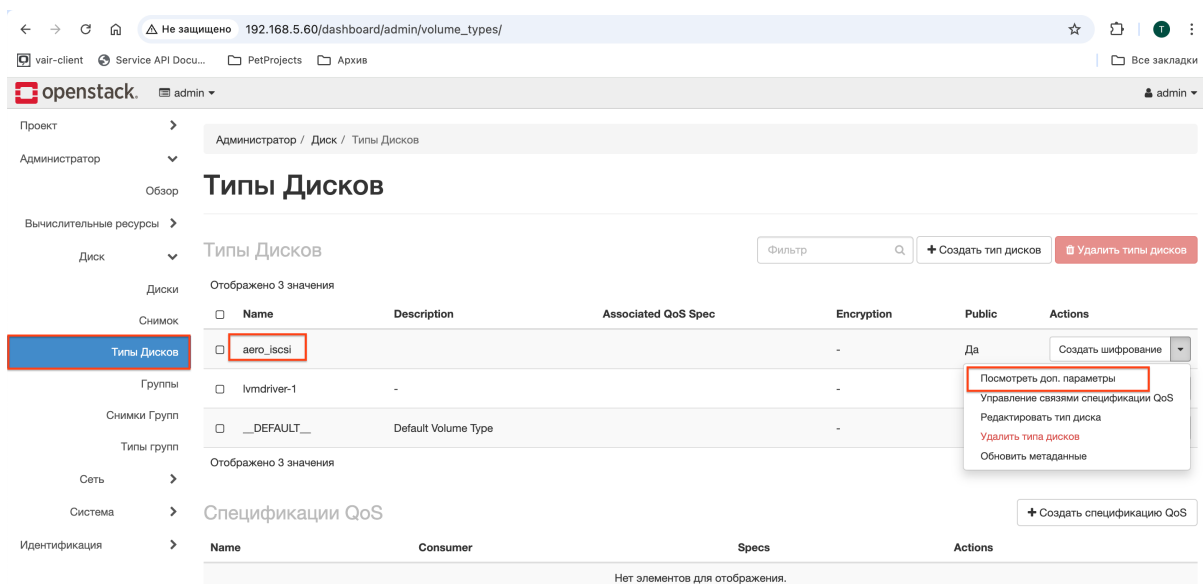


Рис.8: Создание доп. параметров 1

Создать доп. параметры типа дисков

Ключ *

Значение *

Описание:

Создать новый "доп. параметр" - пара ключ-значение - для типа дисков.

Отмена

Создать

Рис.9: Создание доп. параметров 2

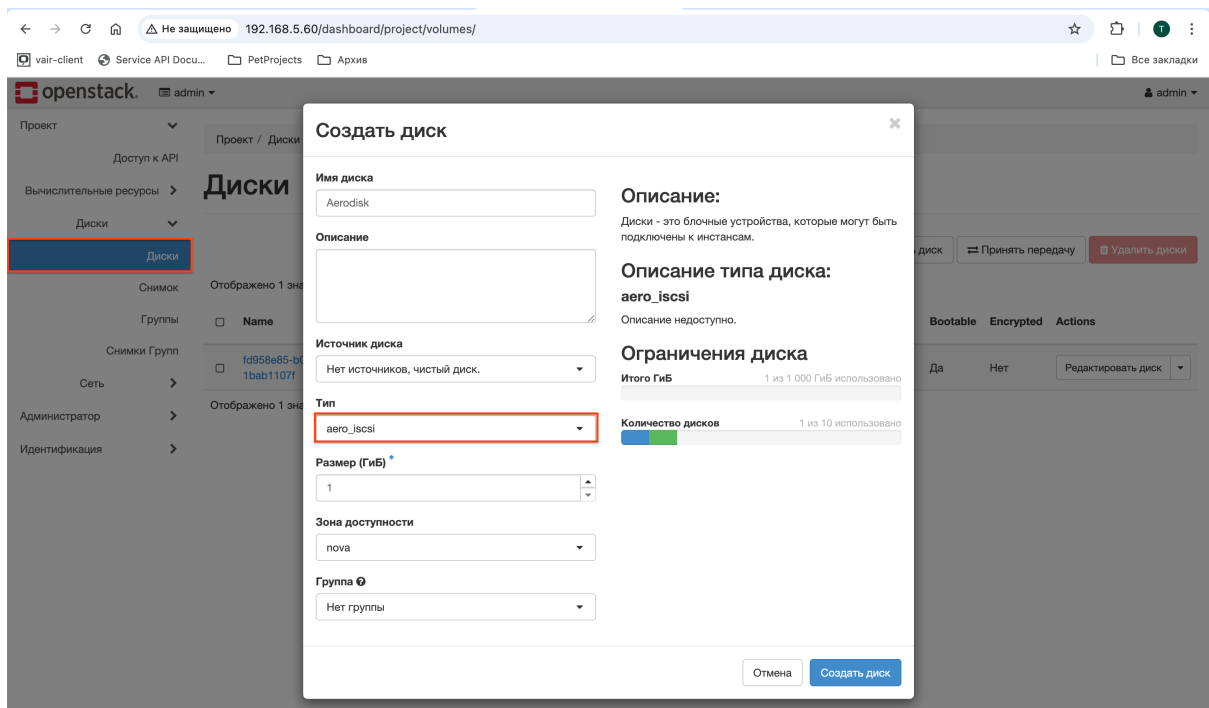


Рис.10: Создание виртуального диска

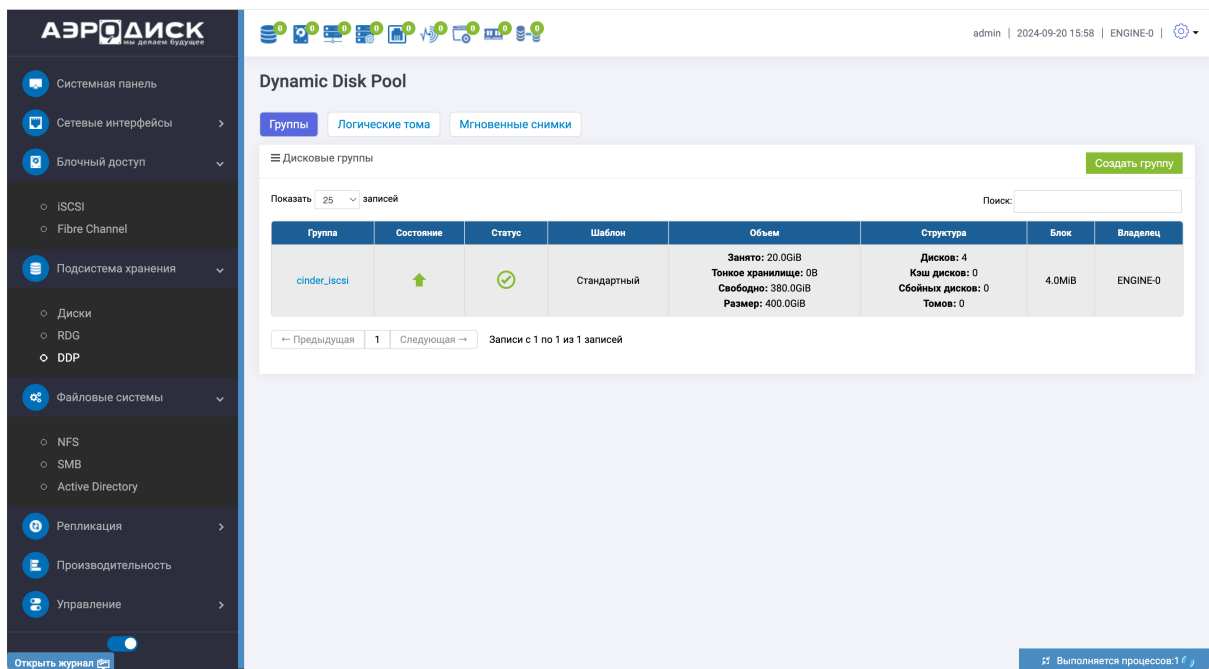


Рис.11: Созданная DDP группа для iscsi

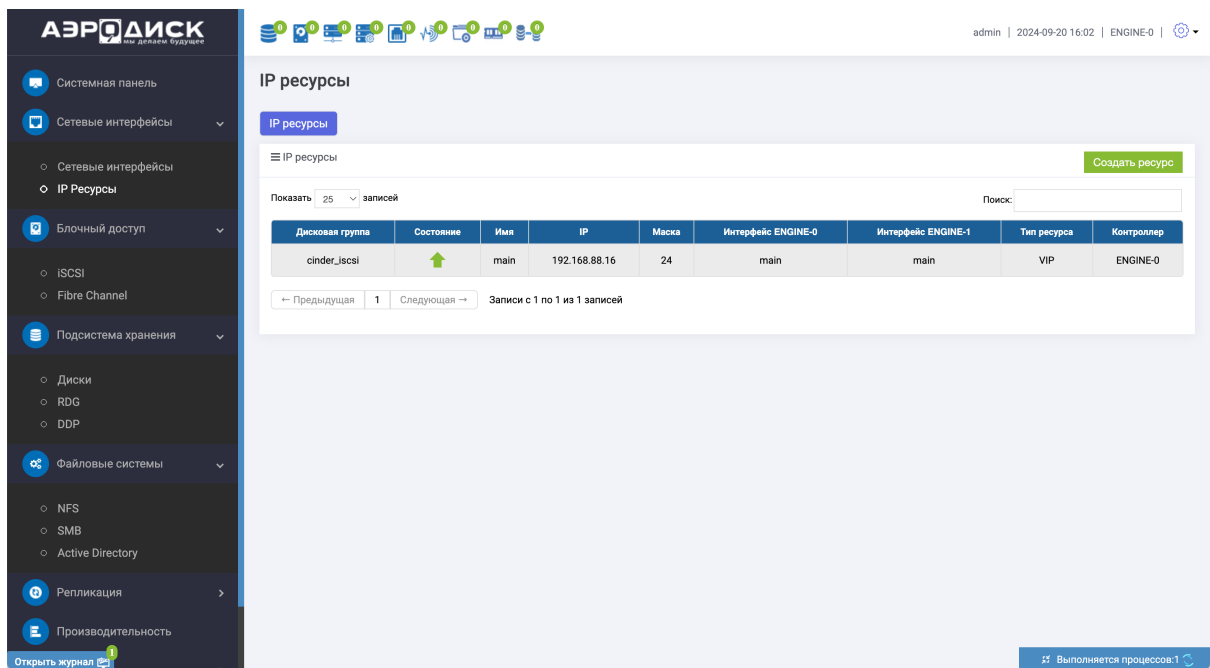


Рис.12: Созданный ip ресурс для iscsi

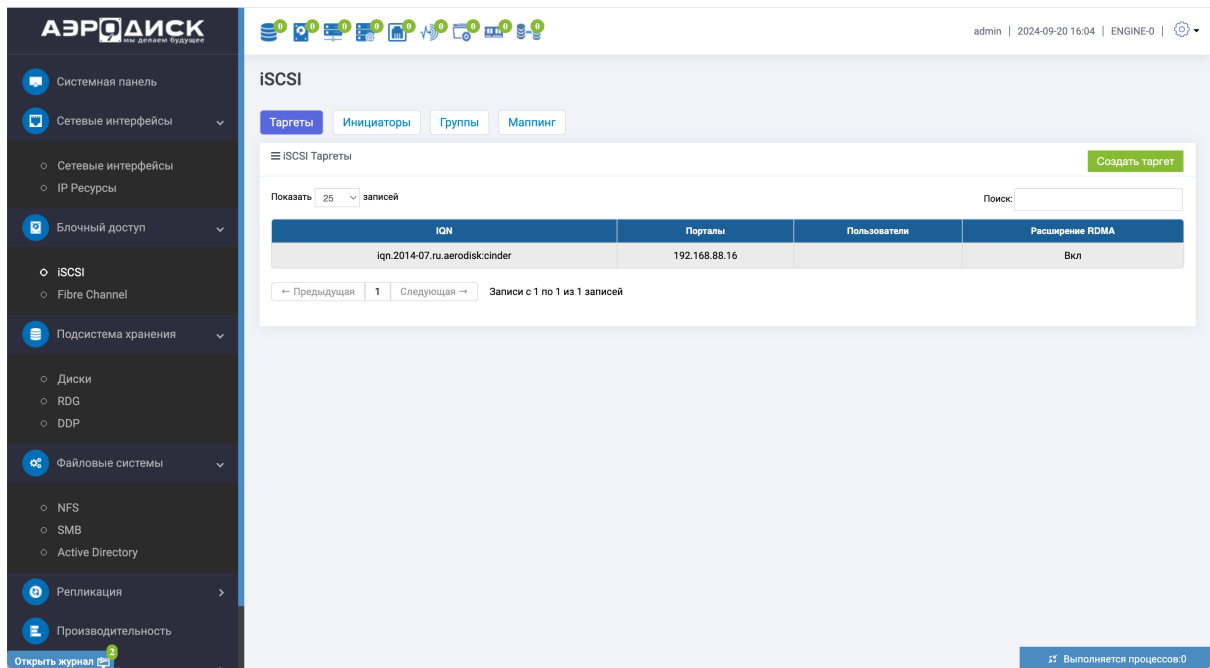


Рис.13: Созданный таргет для iscsi

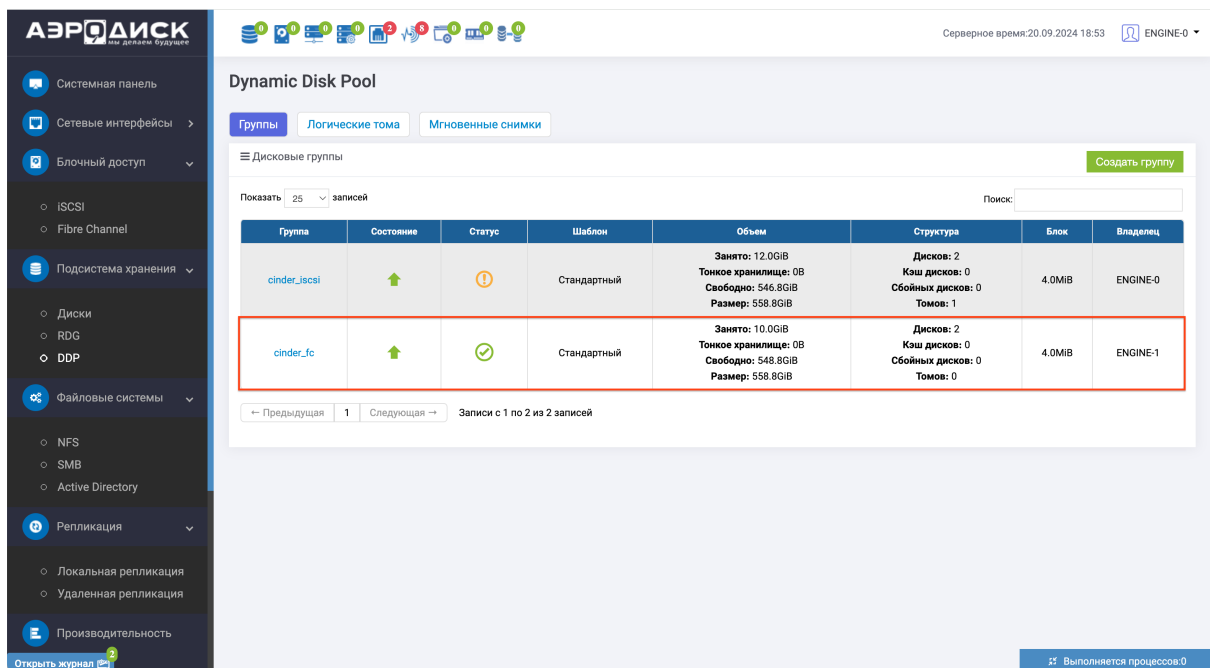


Рис.14: Созданная DDP группа для fibre-channel

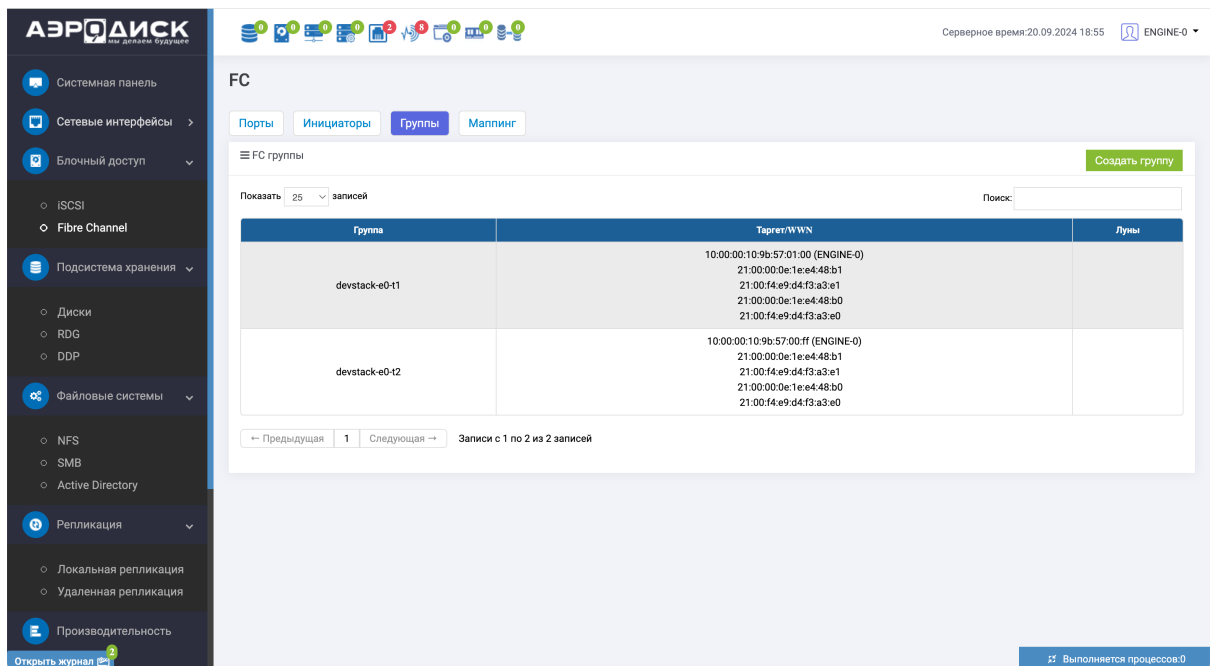


Рис.15: Созданные группы доступа fibre-channel

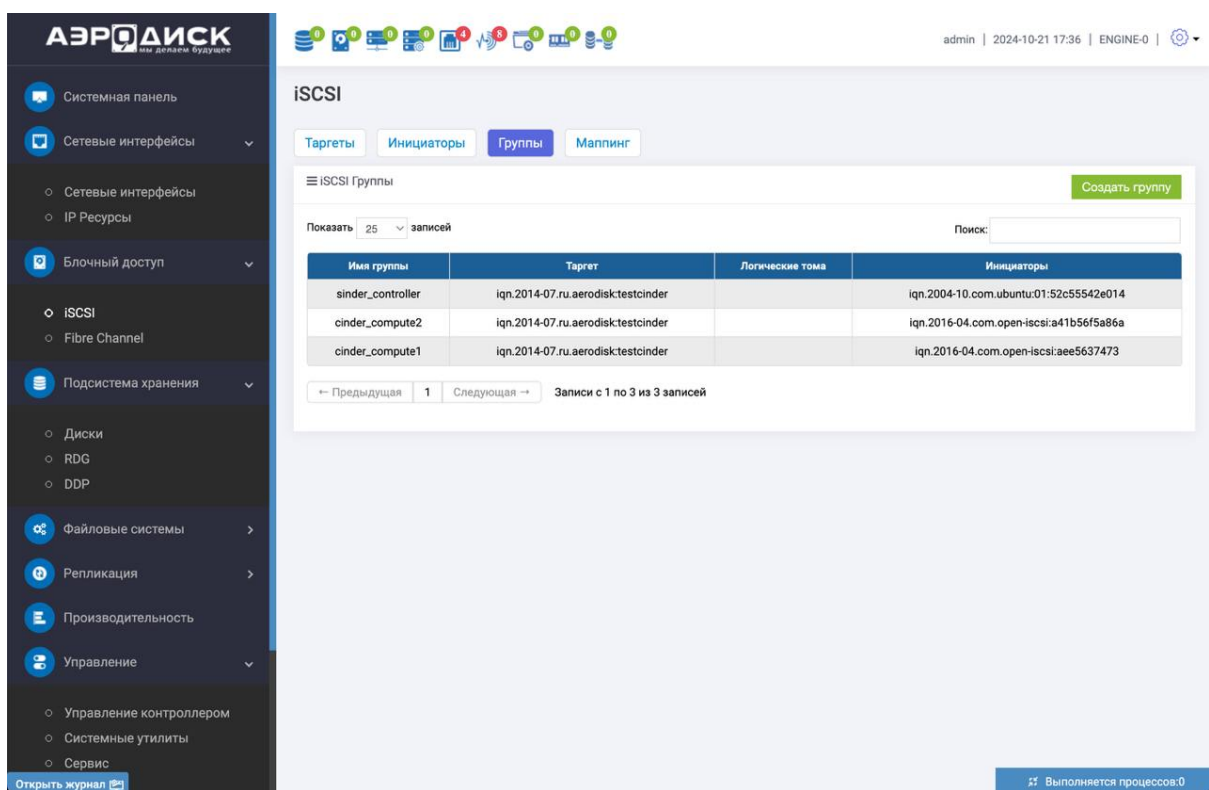


Рис.16: Создание групп доступа iscsi

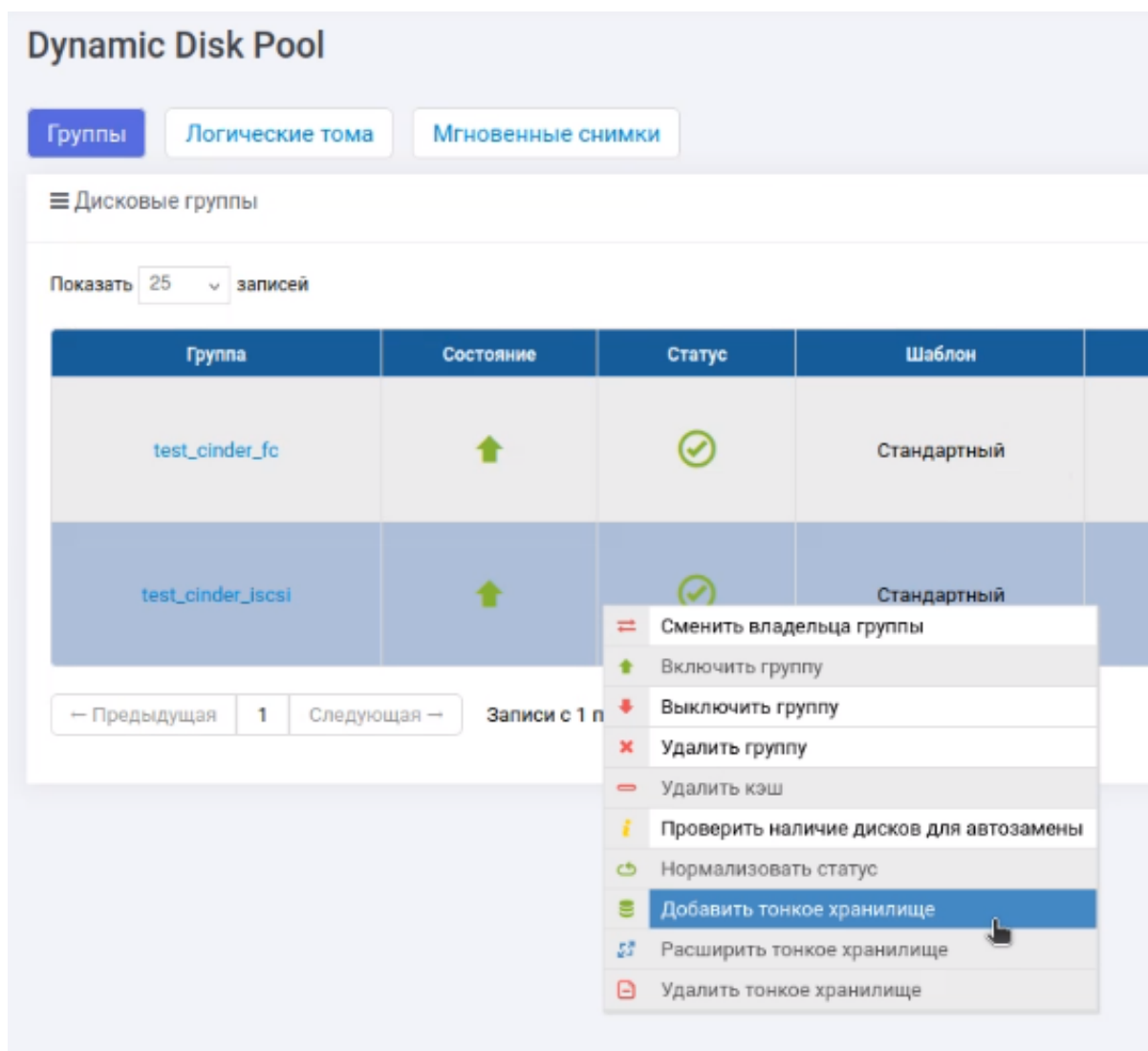


Рис.17: Создание тонкого хранилища 1

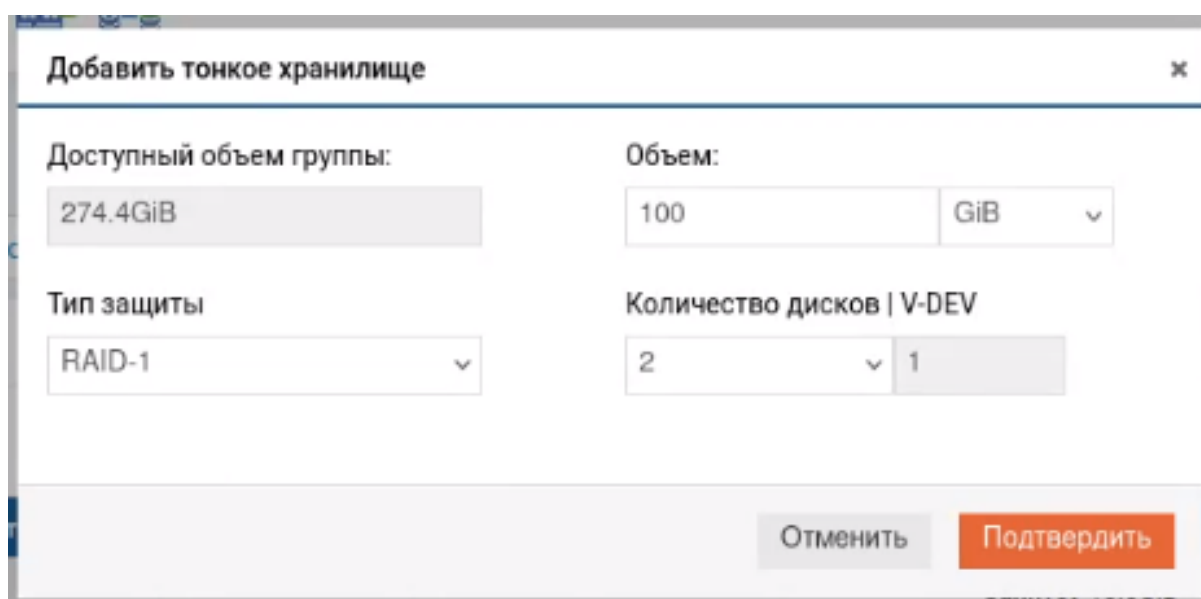


Рис.18: Создание тонкого хранилища 2

Dynamic Disk Pool

Группы

Логические тома

Мгновенные снимки

Дисковые группы

Создать группу

Показать 25 записей

Поиск:

Группа	Состояние	Статус	Шаблон	Объем	Структура	Блок	Владелец
test_cinder_fc	↑	✓	Стандартный	Занято: 10.0GiB Тонкое хранилище: 0B Свободно: 548.8GiB Размер: 558.8GiB	Дисков: 2 Кэш дисков: 0 Сбойных дисков: 0 Томов: 0	4.0MiB	ENGINE-0
test_cinder_iscsi	↑	✓	Стандартный	Занято: 212.0GiB Тонкое хранилище: 100.0GiB Свободно: 346.8GiB Размер: 558.8GiB	Дисков: 2 Кэш дисков: 0 Сбойных дисков: 0 Томов: 0	4.0MiB	ENGINE-1

— Предыдущая

1

Следующая —

Записи с 1 по 2 из 2 записей

Выбрано строк: 1

Рис.19: Создание тонкого хранилища 3