ОПЕРАЦИОННАЯ СИСТЕМА АЛЬТ СЕРВЕР ВИРТУАЛИЗАЦИИ 10.1

Описание функциональных характеристик

СОДЕРЖАНИЕ

1 ОБЩИЕ СВЕДЕНИЯ ОБ ОС АЛЬТ СЕРВЕР ВИРТУАЛИЗАЦИИ 10.1

1.1 Краткое описание возможностей

Операционная система «Альт Сервер Виртуализации» (далее – ОС «Альт Сервер Виртуализации»), представляет собой совокупность интегрированных программ, созданных на основе ОС «Linux», и обеспечивает обработку, хранение и передачу информации в круглосуточном режиме эксплуатации.

ОС «Альт Сервер Виртуализации» – серверный дистрибутив, нацеленный на предоставление функций виртуализации в корпоративной инфраструктуре. Дистрибутив включает в себя средства виртуализации:

- вычислений (ЦПУ и память);
- сети;
- хранения данных.

Управление системой виртуализации возможно через командный интерфейс, веб-интерфейс, с использованием API.

ОС «Альт Сервер Виртуализации» представляет собой решение уровня предприятия, позволяющее осуществить миграцию на импортозамещающее программное и аппаратное обеспечение.

ОС «Альт Сервер Виртуализации» обладает следующими функциональными характеристиками:

- обеспечивает возможность обработки, хранения и передачи информации;
- обеспечивает возможность функционирования в многозадачном режиме (одновременное выполнение множества процессов);
- обеспечивает возможность масштабирования системы: возможна эксплуатация ОС как на одной ПЭВМ, так и в информационных системах различной архитектуры;
- обеспечивает многопользовательский режим эксплуатации;
- обеспечивает поддержку мультипроцессорных систем;
- обеспечивает поддержку виртуальной памяти;
- обеспечивает поддержку запуска виртуальных машин;
- обеспечивает сетевую обработку данных, в том числе разграничение доступа к сетевым пакетам.

ОС «Альт Сервер Виртуализации» предоставляет 4 основных типа установки:

- «Базовый гипервизор». Включает в себя поддержку виртуализации KVM на уровне ядра Linux, утилиты запуска виртуальных машин qemu и унифицированный интерфейс создания и настройки виртуального окружения libvirt. Устанавливается на отдельно стоящий сервер или группу независимых серверов. Для управления используются интерфейс командной строки virsh или графическое приложение virt-manager на рабочей станции администратора.
- «Кластер серверов виртуализации на основе проекта PVE». Устанавливается на группу серверов (до 32 штук). Предназначено для управления виртуальным окружением KVM и контейнерами LXC, виртуальным сетевым окружением и хранилищем данных. Для управления используется интерфейс командной строки, а также веб-интерфейс. Возможна интеграция с корпоративными системами аутентификации (AD, LDAP и другие на основе PAM).
- «Облачная виртуализация уровня предприятия на основе проекта OpenNebula». Для использования необходимы 1 или 3 и более серверов управления (могут быть виртуальными), и группа серверов для запуска виртуальных окружений KVM или контейнеров LXC. Возможна интеграция с корпоративными системами аутентификации.
- «Контейнерная виртуализация». Для использования предлагаются Docker, Podman или LXC/LXD. Для построения кластера и управления контейнерами возможно использование Kubernetes.

ОС «Альт Сервер Виртуализации» поддерживает клиент-серверную архитектуру и может обслуживать процессы как в пределах одной компьютерной системы, так и процессы на других ПЭВМ через каналы передачи данных или сетевые соединения.

1.2 Структура программных средств

ОС «Альт Сервер Виртуализации» состоит из набора компонентов предназначенных для реализации функциональных задач необходимых пользователям (должностным лицам для выполнения определённых должностными инструкциями, повседневных действий) и поставляется в виде дистрибутива и комплекта эксплуатационной документации.

В структуре ОС «Альт Сервер Виртуализации» можно выделить следующие функциональные элементы:

- ядро ОС;
- системные библиотеки;
- утилиты и драйверы;
- средства обеспечения информационной безопасности;
- системные приложения;
- средства обеспечения облачных и распределенных вычислений, средства виртуализации и системы хранения данных;

- системы мониторинга и управления;
- средства подготовки исполнимого кода;
- средства версионного контроля исходного кода;
- библиотеки подпрограмм (SDK);
- среды разработки, тестирования и отладки;
- интерактивные рабочие среды;
- программные серверы;
- веб-серверы;
- системы управления базами данных;
- командные интерпретаторы.

Ядро ОС «Альт Сервер Виртуализации» управляет доступом к оперативной памяти, сети, дисковым и прочим внешним устройствам. Оно запускает и регистрирует процессы, управляет разделением времени между ними, реализует разграничение прав и определяет политику безопасности, обойти которую, не обращаясь к нему, нельзя.

Ядро работает в режиме «супервизора», позволяющем ему иметь доступ сразу ко всей оперативной памяти и аппаратной таблице задач. Процессы запускаются в «режиме пользователя»: каждый жестко привязан ядром к одной записи таблицы задач, в которой, в числе прочих данных, указано, к какой именно части оперативной памяти этот процесс имеет доступ. Ядро постоянно находится в памяти, выполняя системные вызовы – запросы от процессов на выполнение этих подпрограмм.

Системные библиотеки – наборы программ (пакетов программ), выполняющие различные функциональные задачи и предназначенные для динамического подключения к работающим программам, которым необходимо выполнение этих задач.

ОС «Альт Сервер Виртуализации» предоставляет набор дополнительных служб, востребованных в инфраструктуре виртуализации любой сложности и архитектуры:

- сервер сетевой файловой системы NFS;
- распределённая сетевая файловая система CEPH;
- распределённая сетевая файловая система GlusterFS;
- поддержка iSCSI как в качестве клиента, так и создание сервера;
- сетевые службы DNS и DHCP;
- виртуальный сетевой коммутатор Open vSwitch;
- служба динамической маршрутизации bird с поддержкой протоколов BGP, OSPF и др.;
- сетевой балансировщик нагрузки HAProxy, keepallived;
- веб-серверы Apache и Nginx.

В ОС «Альт Сервер Виртуализации» входят агенты мониторинга (Zabbix, telegraf, Prometheus) и архивирования (Bacula, UrBackup), которые могут использоваться совместно с сервисами на ОС «Альт Сервер».

2 ЗАГРУЗКА ОПЕРАЦИОННОЙ СИСТЕМЫ

2.1 Настройка загрузки

Вызов ОС «Альт Сервер Виртуализации», установленной на жесткий диск, происходит автоматически и выполняется после запуска ПЭВМ и отработки набора программ BIOS. ОС «Альт Сервер Виртуализации» вызывает специальный загрузчик.

Загрузчик настраивается автоматически и включает в свое меню все системы, установку которых на ПЭВМ он определил. Поэтому загрузчик также может использоваться для вызова других ОС, если они установлены на компьютере.

Примечание. При наличии на компьютере нескольких ОС (или при наличии нескольких вариантов загрузки), оператор будет иметь возможность выбрать необходимую ОС (вариант загрузки). В случае если пользователем ни один вариант не был выбран, то по истечении заданного времени будет загружена ОС (вариант загрузки), заданные по умолчанию.

При стандартной установке ОС «Альт Сервер Виртуализации» в начальном меню загрузчика доступны несколько вариантов загрузки (Рис. 1): обычная загрузка, загрузка с дополнительными параметрами (например, «recovery mode» – загрузка с минимальным количеством драйверов), загрузка в программу проверки оперативной памяти (memtest).

По умолчанию, если не были нажаты управляющие клавиши на клавиатуре, загрузка ОС «Альт Сервер Виртуализации» продолжится автоматически после небольшого времени ожидания (обычно несколько секунд). Нажав клавишу <Enter>, можно начать загрузку немедленно.

Для выбора дополнительных параметров загрузки нужно выбрать пункт «Дополнительные параметры для ALT Virtualization Server 10.1» («Advanced options for ALT Virtualization Server 10.1»).

Для выполнения тестирования оперативной памяти нужно выбрать пункт «Memtest86+-5.31».

Нажатием клавиши <E> можно вызвать редактор параметров загрузчика GRUB и указать параметры, которые будут переданы ядру ОС при загрузке.

Примечание. Если при установке системы был установлен пароль на загрузчик, потребуется ввести имя пользователя «boot» и заданный на шаге «Установка загрузчика» пароль.

В процессе загрузки ОС «Альт Сервер Виртуализации» пользователь может следить за информацией процесса загрузки, которая отображает этапы запуска различных служб и программных серверов в виде отдельных строк (Рис. 2), на экране монитора.

Варианты загрузки

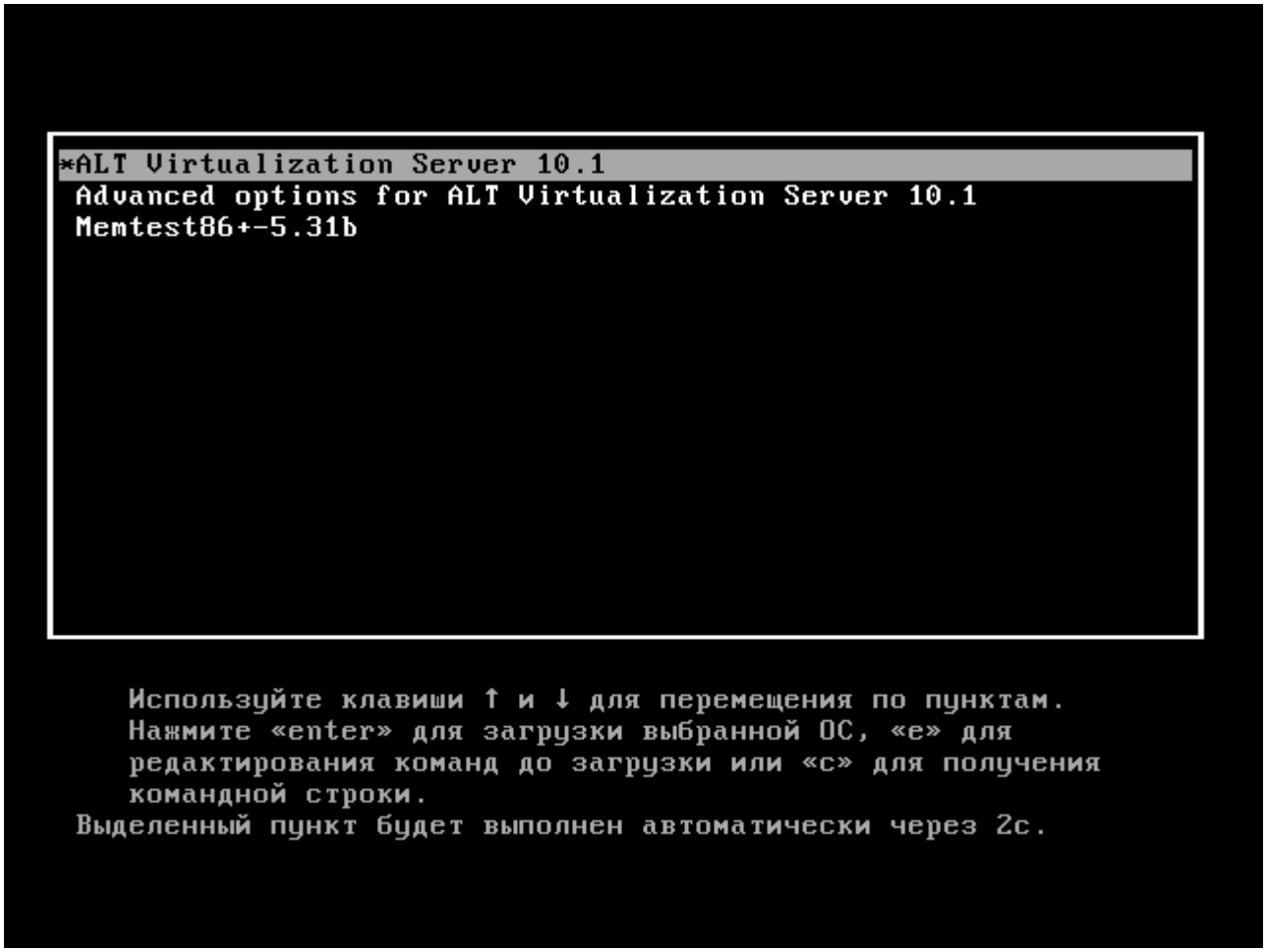


Рис. 1

Загрузка ОС

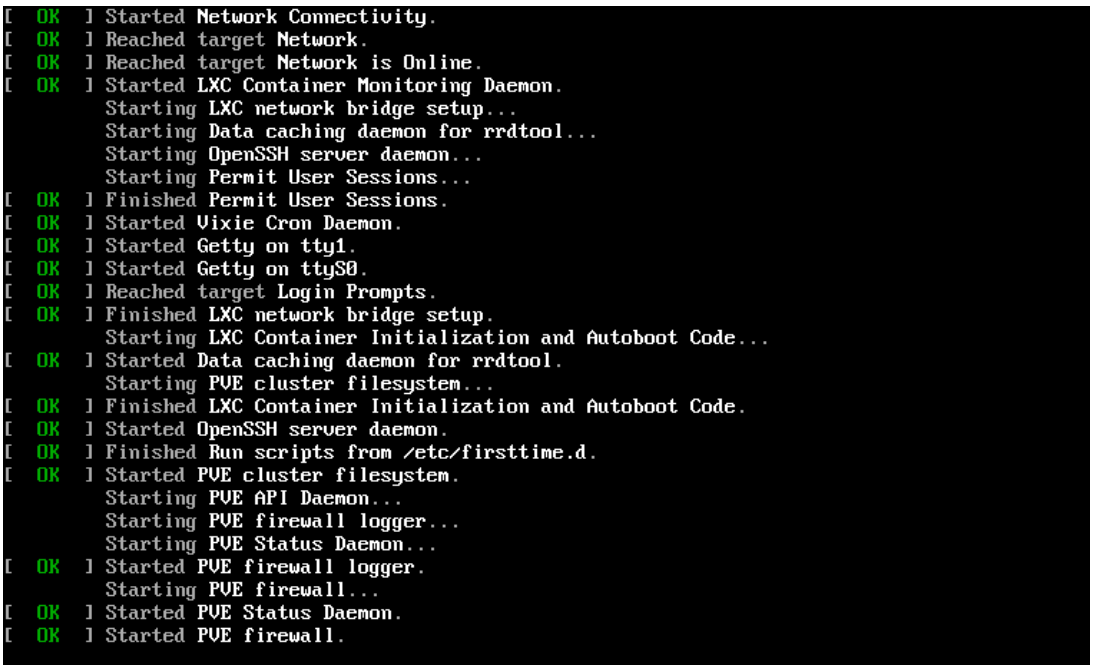


Рис. 2

При этом каждая строка начинается словом вида [XXXXXXX] (FAILED или OK), являющегося признаком нормального или ненормального завершения этапа загрузки. Слово XXXXXXX=FAILED (авария) свидетельствует о неуспешном завершении этапа загрузки, что требует вмешательства и специальных действий администратора системы.

Загрузка ОС может занять некоторое время, в зависимости от производительности компьютера. Основные этапы загрузки операционной системы – загрузка ядра, подключение (монтирование) файловых систем, запуск системных служб – периодически могут дополняться проверкой файловых систем на наличие ошибок. В этом случае время ожидания может занять больше времени, чем обычно.

2.2 Получение доступа к зашифрованным разделам

В случае если был создан шифрованный раздел, потребуется вводить пароль при обращении к этому разделу.

Например, если был зашифрован домашний раздел `/home`, то для того, чтобы войти в систему, потребуется ввести пароль этого раздела и затем нажать <Enter>.

Примечание. Если не ввести пароль за отведенный промежуток времени, то загрузка системы завершится ошибкой. В этом случае следует перезагрузить систему, нажав для этого два раза <Enter>, а затем клавиши <Ctrl>+<Alt>+<Delete>.

2.3 Вход и работа в системе в консольном режиме

Стандартная установка ОС «Альт Сервер Виртуализации» включает базовую систему, работающую в консольном режиме.

При загрузке в консольном режиме работа загрузчика ОС «Альт Сервер Виртуализации» завершается запросом на ввод логина и пароля учетной записи. Для дальнейшего входа в систему необходимо ввести логин и пароль учетной записи пользователя. (Рис. 3).

Приглашение для ввода команд

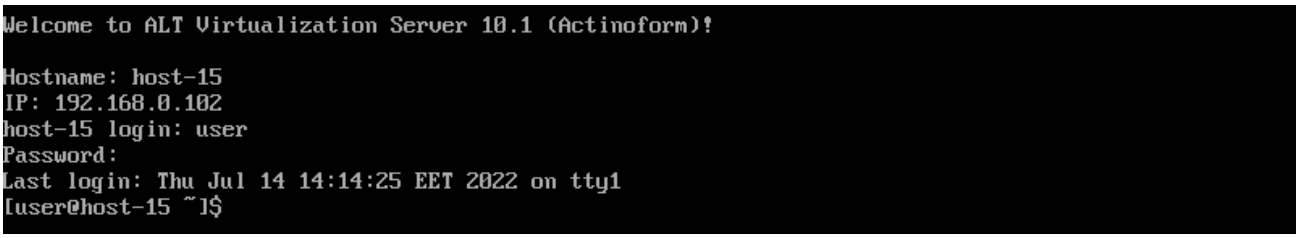


Рис. 3

В случае успешного прохождения процедуры аутентификации и идентификации будет выполнен вход в систему. ОС «Альт Сервер Виртуализации» перейдет к штатному режиму работы и предоставит дальнейший доступ к консоли

П р и м е ч а н и е . После загрузки будут показаны имя и IP-адрес компьютера, а также, если были установлены OpenNebula или PVE, адрес доступа к панели управления (Рис. 4).

IP-адрес компьютера и адрес панели управления PVE

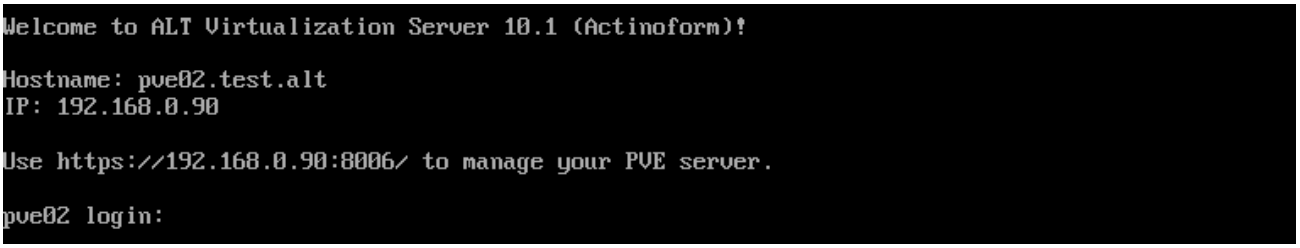


Рис. 4

2.4 Виртуальная консоль

В процессе работы ОС «Альт Сервер Виртуализации» активно несколько виртуальных консолей. Каждая виртуальная консоль доступна по одновременному нажатию клавиш <Ctrl>, <Alt> и функциональной клавиши с номером этой консоли от <F1> до <F6>.

На первых шести виртуальных консолях (от <Ctrl>+<Alt>+<F1> до <Ctrl>+<Alt>+<F6>) пользователь может зарегистрироваться и работать в текстовом режиме. Двенадцатая виртуальная консоль (<Ctrl>+<Alt>+<F12>) выполняет функцию системной консоли – на нее выводятся сообщения о происходящих в системе событиях.

3 OPENNEBULA

OpenNebula – это платформа облачных вычислений для управления разнородными инфраструктурами распределенных центров обработки данных. Платформа OpenNebula управляет виртуальной инфраструктурой центра обработки данных для создания частных, общедоступных и гибридных реализаций инфраструктуры как службы.

Облачная архитектура определяется 3-мя элементами: хранилищем данных, сетью и системой виртуализации.

OpenNebula состоит из следующих компонентов:

- Сервер управления (Front-end) – на нём выполняются сервисы OpenNebula;
- Серверы с виртуальными машинами;
- Хранилище данных – содержит образы ВМ;
- Физическая сеть – обеспечивает связь между хранилищем данных, серверами с ВМ, поддерживает VLAN-ы для ВМ, а также управление сервисами OpenNebula.

Примечание. Компоненты OpenNebula будут установлены в систему, если при установке дистрибутива выбрать профиль «Вычислительный узел Opennebula KVM» или «Сервер Opennebula».

3.1 Планирование ресурсов

3.1.1 Сервер управления

Минимальные требования к серверу управления показаны в таблице 1.

Таблица 1 – Минимальные требования к серверу управления

Ресурс	Минимальное значение
Оперативная память	2 ГБ
CPU	1 CPU (2 ядра)
Диск	100 ГБ
Сеть	2 интерфейса

Максимальное количество серверов (узлов виртуализации), управляемых одним сервером управления, зависит от инфраструктуры, особенно от производительности хранилища. Обычно рекомендуется не управлять более чем 500 серверами из одной точки, хотя существуют примеры с более чем 1000 серверами.

3.1.2 Серверы виртуализации

Серверы виртуализации – это физические машины, на которых выполняются виртуальные машины. Подсистема виртуализации – это компонент, который отвечает за связь с гипервизором,

установленным на узлах, и выполнение действий, необходимых для каждого этапа жизненного цикла виртуальной машины (ВМ).

Серверы (узлы) виртуализации имеют следующие характеристики и их рекомендованные значения:

- CPU – в обычных условиях каждое ядро, предоставляемое ВМ, должно быть реальным ядром физического процессора. Например, для обслуживания 40 ВМ с двумя процессорами в каждой, облако должно иметь 80 физических ядер. При этом они могут быть распределены по разным серверам: 10 серверов с восемью ядрами или 5 серверов с 16 ядрами на каждом. В случае перераспределения недостаточных ресурсов используются атрибуты CPU и VCPU: CPU определяет физические ядра, выделенные для ВМ, а VCPU – виртуальные ядра для гостевой ОС;
- Память – по умолчанию, OpenNebula не предоставляет памяти для гостевых систем больше, чем есть на самом деле. Желательно рассчитывать объём памяти с запасом в 10% на гипервизор. Например, для 45 ВМ с 2 ГБ памяти на каждой, необходимо 90 ГБ физической памяти. Важным параметром является количество физических серверов: каждый сервер должен иметь 10% запас для работы гипервизора, так, 10 серверов с 10 ГБ памяти на каждом могут предоставить по 9 ГБ для виртуальных машин и смогут обслужить 45 машин из этого примера (10% от 10 ГБ = 1 ГБ на гипервизор).

3.1.3 Хранилище данных

OpenNebula работает с двумя видами данных в хранилище: образцами виртуальных машин и образами (дисками) самих ВМ.

В хранилище образов (Images Datastore) OpenNebula хранит все зарегистрированные образы, которые можно использовать для создания ВМ.

Системное хранилище (System Datastore) – используется для хранения дисков виртуальных машин, работающих в текущий момент. Образы дисков перемещаются, или клонируются, в хранилище образов или из него при развертывании и отключении ВМ, при подсоединении или фиксировании мгновенного состояния дисков.

Одним из основных способов управления хранилищем данных является ограничение хранилища, доступного для пользователей, путем определения квот по максимальному количеству ВМ, а также максимального объема энергозависимой памяти, который может запросить пользователь, и обеспечения достаточного пространства хранения системных данных и образов, отвечающего предельным установленным квотам. OpenNebula позволяет администратору добавлять хранилища системных данных и образов.

Планирование хранилища – является критически важным аспектом, поскольку от него зависит производительность облака. Размер хранилищ сильно зависит от базовой технологии.

Например, при использовании Ceph для среднего по размеру облака, необходимо взять как минимум 3 сервера в следующей конфигурации: 5 дисков по 1 ТБ, 16 ГБ памяти, 2 CPU по 4 ядра в каждом и как минимум 2 сетевые карты.

3.1.4 Сетевая инфраструктура

Сетевая инфраструктура должна быть спланирована так, чтобы обеспечить высокую надёжность и пропускную способность. Рекомендуется использовать 2 сетевых интерфейса на сервере управления и по 4 на каждом сервере виртуализации (публичный, внутренний, для управления и для связи с хранилищем).

3.2 Запуск сервера управления OpenNebula

3.2.1 Установка пароля для пользователя oneadmin

При установке OpenNebula система автоматически создает нового пользователя oneadmin, все дальнейшие действия по управлению OpenNebula необходимо выполнять от этого пользователя.

Примечание. Файл `/var/lib/one/.one/one_auth` будет создан со случайно сгенерированным паролем. Необходимо поменять этот пароль перед запуском OpenNebula.

Для установки пароля для пользователя oneadmin необходимо выполнить команду:

```
# passwd oneadmin
```

Теперь зайдя под пользователем oneadmin, следует заменить содержимое `/var/lib/one/.one/one_auth`. Он должен содержать следующее: `oneadmin: <пароль>`. Например:

```
$ echo "oneadmin:mypassword" > ~/.one/one_auth
```

3.2.2 Настройка MySQL (MariaDB) для хранения конфигурации

По умолчанию OpenNebula работает с SQLite. Если планируется использовать OpenNebula с MySQL, следует настроить данную конфигурацию перед первым запуском OpenNebula, чтобы избежать проблем с учетными данными oneadmin и serveradmin.

Примечание. Задать пароль root для mysql и настройки безопасности:

```
# mysql_secure_installation
```

Создать нового пользователя, предоставить ему привилегии в базе данных opennebula (эта база данных будет создана при первом запуске OpenNebula) и настроить уровень изоляции:

```
$ mysql -u root -p
Enter password:

MariaDB > GRANT ALL PRIVILEGES ON opennebula.* TO 'oneadmin' IDENTIFIED BY
'<thepassword>';
Query OK, 0 rows affected (0.003 sec)
```

```
MariaDB > SET GLOBAL TRANSACTION ISOLATION LEVEL READ COMMITTED;
Query OK, 0 rows affected (0.001 sec)

MariaDB > quit
```

Перед запуском сервера OpenNebula в первый раз необходимо настроить параметры доступа к базе данных в конфигурационном файле `/etc/one/oned.conf`:

```
#DB = [ BACKEND = "sqlite" ]

# Sample configuration for MySQL
DB = [ BACKEND = "mysql",
       SERVER  = "localhost",
       PORT    = 0,
       USER    = "oneadmin",
       PASSWD  = "<thepassword>",
       DB_NAME = "opennebula",
       CONNECTIONS = 50 ]
```

3.2.3 Запуск OpenNebula

Для запуска OpenNebula необходимо выполнить следующие команды:

```
# systemctl start opennebula
# systemctl start opennebula-sunstone
```

3.2.4 Проверка установки

После запуска OpenNebula в первый раз, следует проверить, что команды могут подключаться к демону OpenNebula. Это можно сделать в командной строке или в графическом интерфейсе пользователя: Sunstone.

В командной строке:

```
$ oneuser show
USER 0 INFORMATION
ID              : 0
NAME            : oneadmin
GROUP           : oneadmin
PASSWORD        : 3bc15c8aae3e4124dd409035f32ea2fd6835efc9
AUTH_DRIVER     : core
ENABLED         : Yes
USER TEMPLATE
TOKEN_PASSWORD="ec21d27e2fe4f9ed08a396cbd47b08b8e0a4ca3c"
VMS USAGE & QUOTAS
```

```
VMS USAGE & QUOTAS - RUNNING
DATASTORE USAGE & QUOTAS
NETWORK USAGE & QUOTAS
IMAGE USAGE & QUOTAS
```

Также можно попробовать войти в веб-интерфейс Sunstone. Для этого необходимо перейти по адресу http://<внешний адрес>:9869. Если все в порядке, будет предложена страница входа (Рис. 5).

Страница авторизации opennebula-sunstone

Рис. 5

Необходимо ввести в соответствующие поля имя пользователя (oneadmin) и пароль пользователя (тот, который находится в файле `/var/lib/one/.one/one_auth`).

После входа в систему будет доступна панель инструментов (Рис. 6).

Для смены языка интерфейса необходимо в левом меню выбрать пункт «Settings», и на открывшейся странице в выпадающем списке «Language» выбрать пункт «Russian (ru_RU)» (Рис. 7). Язык интерфейса будет изменён на русский (Рис. 8).

Панель инструментов opennebula-sunstone

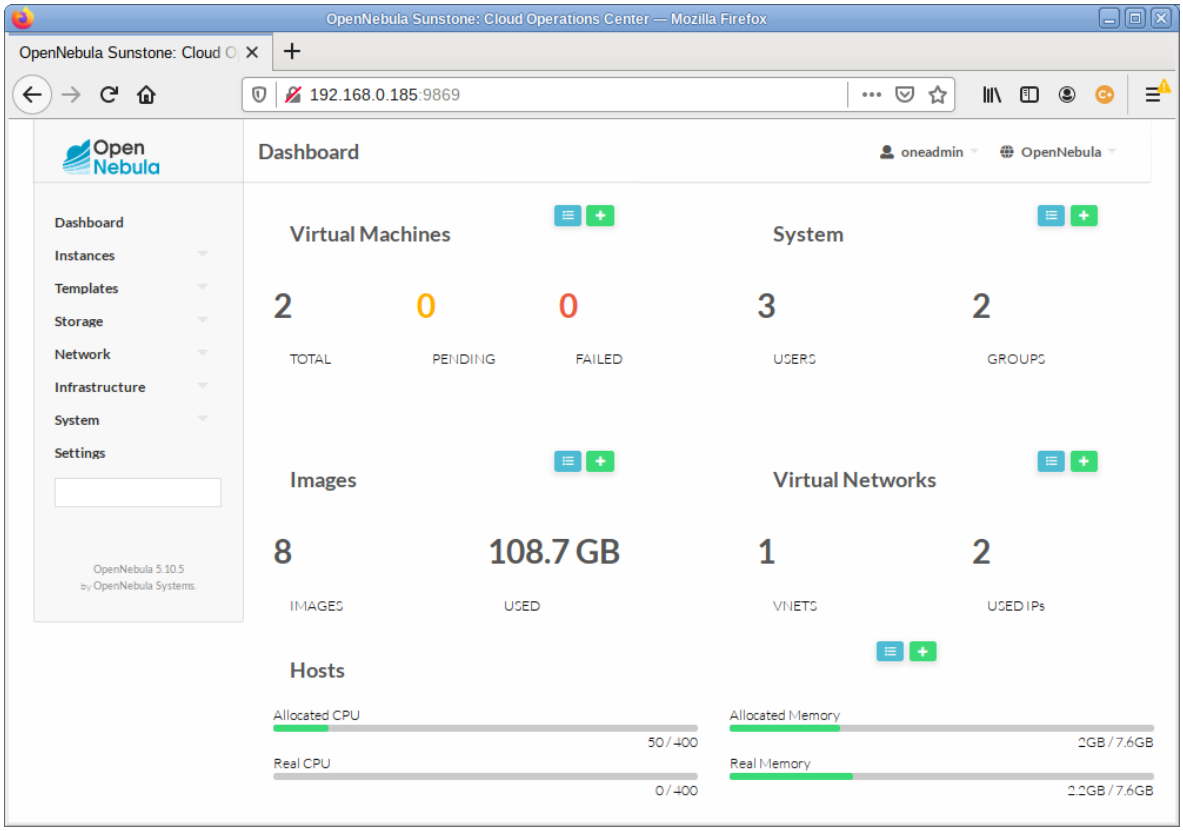


Рис. 6

Выбор языка интерфейса

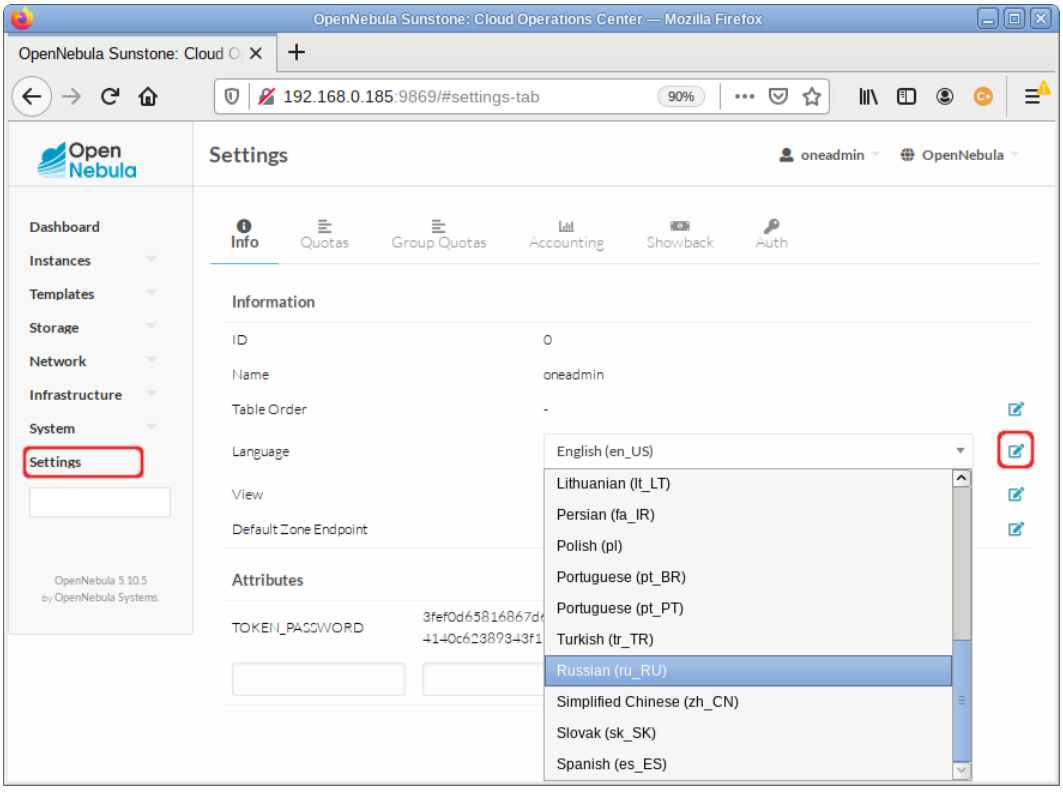


Рис. 7

Панель инструментов opennebula-sunstone с русским языком интерфейса

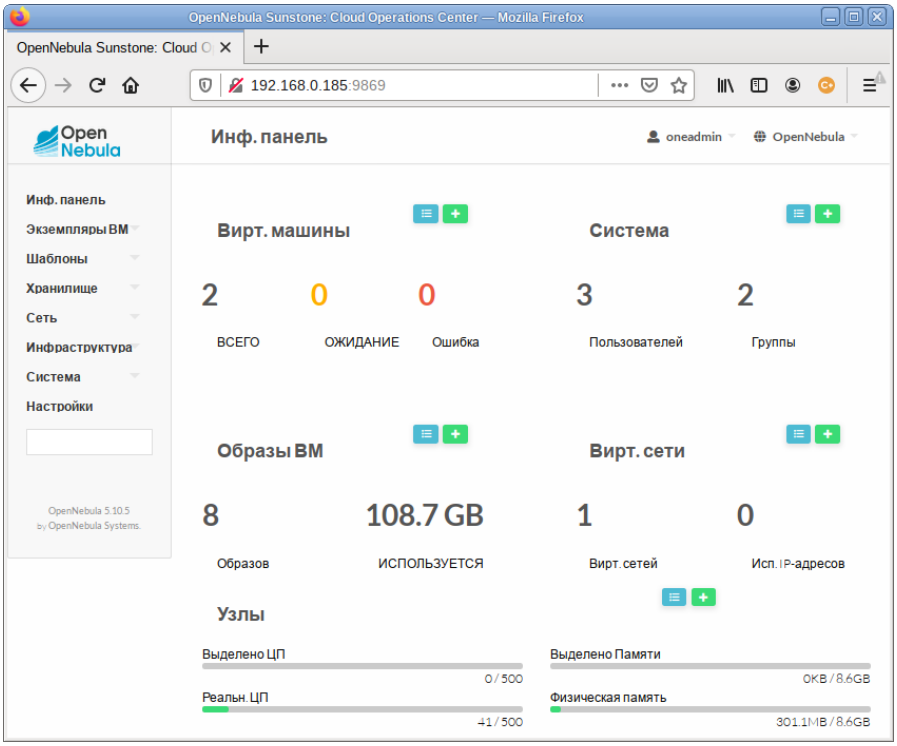


Рис. 8

3.2.5 Ключи для доступа по SSH

Сервер управления OpenNebula подключается к хостам гипервизора по SSH. Необходимо распространить открытый ключ пользователя oneadmin со всех машин в файл `/var/lib/one/.ssh/authorized_keys` на всех машинах.

При установке сервера управления OpenNebula ключ SSH был сгенерирован и добавлен в авторизованные ключи. Необходимо синхронизировать id_rsa, id_rsa.pub и authorized_keys сервера управления и узлов. Кроме того, следует создать файл known_hosts и также синхронизировать его с узлами. Чтобы создать файл known_hosts, необходимо выполнить следующую команду (от пользователя oneadmin на сервере управления) со всеми именами узлов и именем сервера управления в качестве параметров:

```
$ ssh-keyscan <сервер_управления> <узел1> <узел2> <узел3> ... >> /var/lib/one/.ssh/known_hosts
```

Примечание. Команду ssh-keyscan необходимо выполнить, как для имён, так и для IP-адресов узлов:

```
$ ssh-keyscan 192.168.0.185 server 192.168.0.190 host01 >> /var/lib/one/.ssh/known_hosts
```

Далее необходимо скопировать каталог `/var/lib/one/.ssh` на все узлы. Самый простой способ – установить временный пароль для oneadmin на всех хостах и скопировать каталог с сервера управления:

```
$ scp -rp /var/lib/one/.ssh <узел1>:/var/lib/one/
$ scp -rp /var/lib/one/.ssh <узел2>:/var/lib/one/
$ scp -rp /var/lib/one/.ssh <узел3>:/var/lib/one/
...
```

После этого следует убедиться, что ни одно из этих подключений (под пользователем oneadmin) не заканчивается ошибкой, и пароль не запрашивается:

- от сервера управления к самому серверу управления;
- от сервера управления ко всем узлам;
- от всех узлов на все узлы;
- от всех узлов к серверу управления.

Эту проверку можно выполнить, например, на сервере управления:

```
# от сервера управления к самому серверу управления
ssh <сервер_управления>
exit

# от сервера управления к узлу, обратно на сервер управления и к другим узлам
ssh <узел1>
ssh <сервер_управления>
exit
ssh <узел2>
exit
ssh <узел3>
exit
exit
```

И так далее для всех узлов.

Если требуется дополнительный уровень безопасности, можно хранить закрытый ключ только на сервере управления, а не копировать его на весь гипервизор. Таким образом, пользователь oneadmin в гипервизоре не сможет получить доступ к другим гипервизорам. Это достигается путем изменения `/var/lib/one/.ssh/config` на сервере управления и добавления параметра `ForwardAgent` к хостам гипервизора для пересылки ключа:

```
$ cat /var/lib/one/.ssh/config
 Host host01
    User oneadmin
    ForwardAgent yes
```

```
Host host02
    User oneadmin
    ForwardAgent yes
```

3.2.6 Конфигурация сети

Сервисам, работающим на сервере управления, необходим доступ к узлам с целью управления гипервизорами и их мониторинга, а также для передачи файлов образов. Для этой цели рекомендуется использовать выделенную сеть.

Примечание. Настройка сети необходима только на серверах с виртуальными машинами. Точное имя ресурсов (br0, br1 и т.д.) значения не имеет, но важно, чтобы мосты и сетевые карты имели одно и то же имя на всех узлах.

3.3 Настройка узлов

3.3.1 Установка и настройка узла OpenNebula KVM

Перед добавлением узла типа KVM на сервер OpenNebula следует настроить узел KVM.

Для создания узла типа KVM при установке дистрибутива нужно выбрать профиль «Вычислительный узел Opennebula KVM» (Рис. 9).

Установка сервера виртуализации KVM

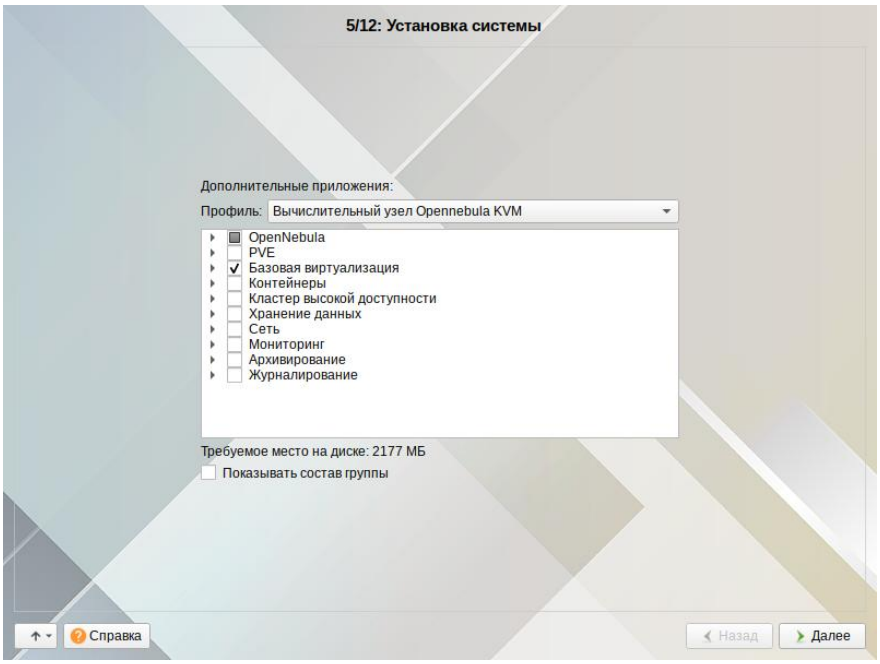


Рис. 9

Примечание. В уже установленной системе можно установить пакет opennebula-node-kvm:

```
# apt-get install opennebula-node-kvm
```

и добавить службу libvirtd в автозапуск и запустить её:

```
# systemctl enable --now libvirtd
```

После создания узла следует задать пароля для пользователя oneadmin:

```
# passwd oneadmin
```

и настроить доступ по SSH (см. раздел «Ключи для доступа по SSH»).

3.3.2 Настройка узла OpenNebula LXD

LXD – это гипервизор LXC контейнеров.

Примечание. Для работы с LXD в Opennebula должна быть настроена пара хранилищ (хранилище образов и системное) НЕ типа qcow2 (например, shared или ssh).

Перед добавлением хоста типа LXD на сервер OpenNebula следует настроить узел LXD.

Для создания узла типа LXD, при установке дистрибутива нужно выбрать профиль «Вычислительный узел Opennebula LXD» (Рис. 10).

Установка сервера контейнеризации LXD

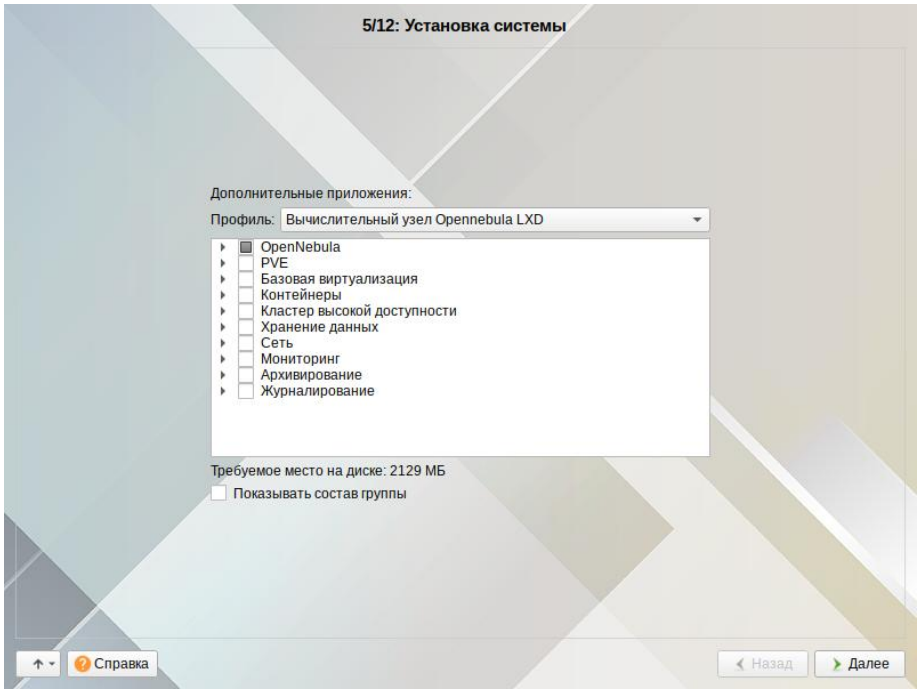


Рис. 10

Примечание. Для создания узла типа LXD в уже установленной системе необходимо выполнить следующие шаги:

- установить пакет opennebula-node-lxd:

```
# apt-get install opennebula-node-lxd
```

- запустить и добавить в автозапуск lxd.socket:

```
# systemctl enable --now lxd.socket
```

- инициализировать lxd, выполнив команду из файла `/usr/share/doc/opennebula-node-lxd-5.10.5/README.opennebula-lxd`;
- запустить и добавить в автозапуск libvirtd.socket:

```
# systemctl enable --now libvirtd.socket
```

В уже установленной системе можно установить пакет opennebula-node-lxd:

```
# apt-get install opennebula-node-lxd
```

После создания узла следует задать пароля для пользователя oneadmin:

```
# passwd oneadmin
```

и настроить доступ по SSH (см. раздел Ключи для доступа по SSH).

3.4 Добавление узлов в OpenNebula

Чтобы использовать существующие физические узлы, их необходимо зарегистрировать в OpenNebula. Регистрация узла в OpenNebula может быть выполнена в командной строке или в веб-интерфейсе Sunstone.

Примечание. Перед добавлением узла следует убедиться, что к узлу можно подключиться по SSH без запроса пароля.

3.4.1 Добавление узла типа KVM в OpenNebula-Sunstone

Для добавления узла, необходимо в левом меню выбрать «Инфраструктура» → «Узлы» и на загруженной странице нажать кнопку «+» (Рис. 11).

Далее необходимо указать тип виртуализации, заполнить поле «Имя хоста» (можно ввести IP-адрес узла, или его имя) и нажать кнопку «Создать» (Рис. 12).

Добавление узла в OpenNebula-Sunstone

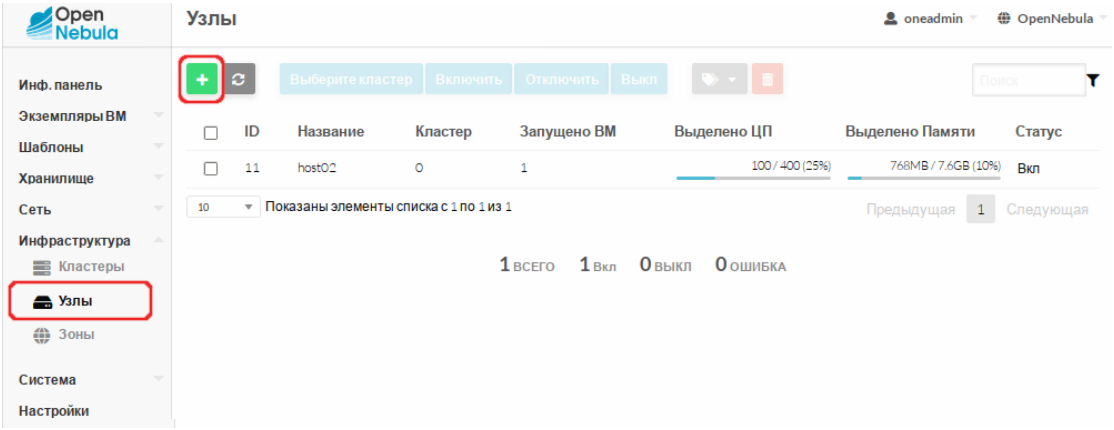


Рис. 11

Добавление узла в OpenNebula-Sunstone

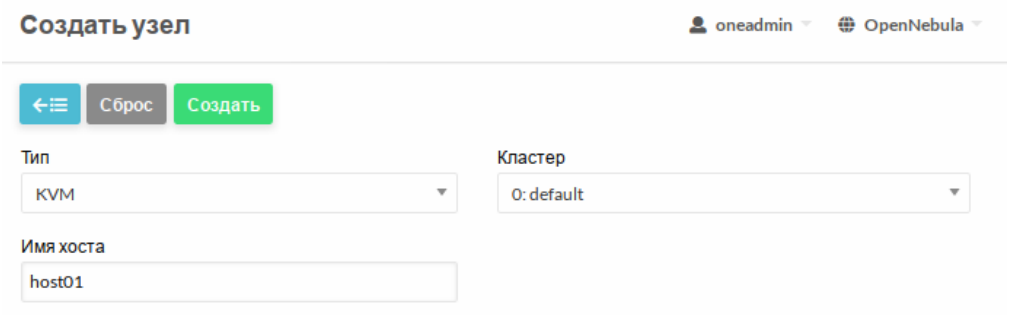


Рис. 12

Затем следует вернуться к списку узлов и убедиться, что узел перешел в состояние «Вкл» (это должно занять от 20 секунд до 1 минуты, можно нажать кнопку «Обновить» для обновления состояния) (Рис. 13).

Добавление узла в OpenNebula-Sunstone

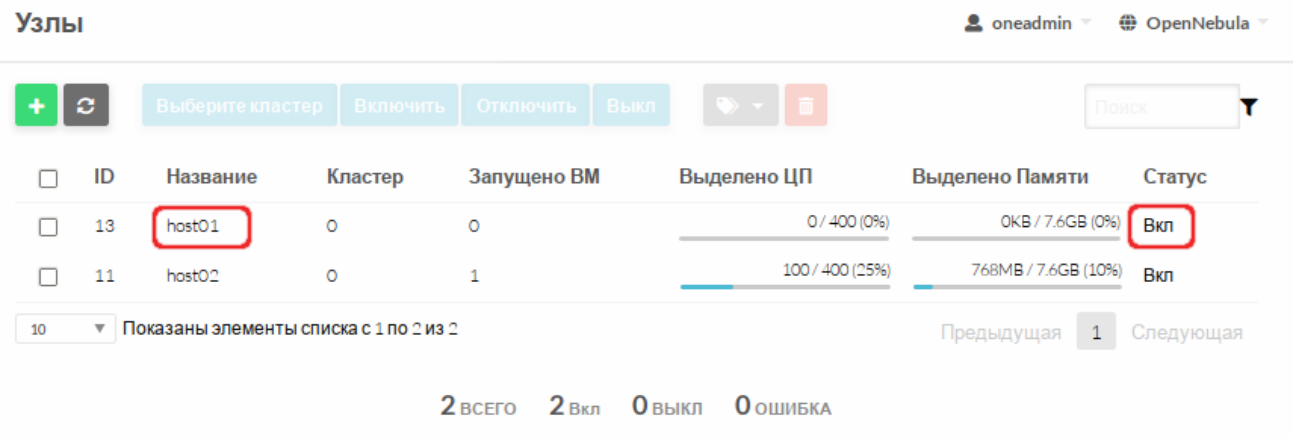


Рис. 13

3.4.2 Добавление узла типа LXD в OpenNebula-Sunstone

Для добавления узла типа LXD на сервере OpenNebula, необходимо в левом меню выбрать «Инфраструктура» → «Узлы» и на загруженной странице нажать кнопку «+».

Далее необходимо указать тип виртуализации – LXD, заполнить поле «Имя хоста» (можно ввести IP-адрес узла, или его имя) и нажать кнопку «Создать» (Рис. 14).

Добавление узла типа LXD в OpenNebula-Sunstone

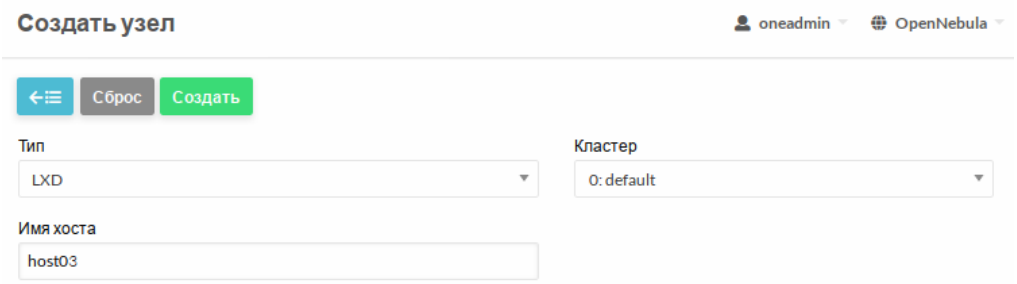


Рис. 14

Затем следует вернуться к списку узлов и убедиться, что узел перешел в состояние ВКЛ (это должно занять от 20 секунд до 1 минуты).

3.4.3 Работа с узлами в командной строке

onehost – это инструмент управления узлами в OpenNebula. Описание всех доступных опций утилиты onehost можно получить, выполнив команду:

```
$ man onehost
```

Для добавления узла KVM в облако, необходимо выполнить следующую команду от oneadmin на сервере управления:

```
$ onehost create host01 --im kvm --vm kvm
ID: 1
```

Добавление узла типа LXD в командной строке:

```
$ onehost create host03 --im lxd --vm lxd
ID: 3
```

Список узлов можно просмотреть, выполнив команду:

```
$ onehost list
  ID NAME          CLUSTER    TVM      ALLOCATED_CPU      ALLOCATED_MEM STAT
   1 host01        default      0       0 / 400 (0%)     0K / 7.6G (0%) on
```

Примечание. Если возникли проблемы с добавлением узла, то, скорее всего, неправильно настроен ssh. Ошибки можно просмотреть в `/var/log/one/oned.log`.

Для указания узла можно использовать его ID или имя. Например, удаление узла с указанием ID или имени:

```
$ onehost delete 1
$ onehost delete host01
```

Изменение статуса узла:

```
$ onehost disable host01 // деактивировать узел
$ onehost enable host01 // активировать узел
$ onehost offline host01 // полностью выключить узел
```

Просмотр информации об узле:

```
$ onehost show host01
```

Вывод данной команды содержит:

- общую информацию об узле;
- информацию о процессоре и объёме оперативной памяти (Host Shares);
- информацию о локальном хранилище данных (Local System Datastore), если хост настроен на использование локального хранилища данных;
- данные мониторинга;
- информацию о ВМ запущенных на узле.

3.5 Работа с хранилищами в OpenNebula

По умолчанию в OpenNebula созданы хранилище образов (Images), системное (System) и файлов (Files).

onedatastore – инструмент управления хранилищами в OpenNebula. Описание всех доступных опций утилиты onedatastore можно получить, выполнив команду:

```
$ man onedatastore
```

Вывести список хранилищ данных можно, выполнив команду:

```
$ onedatastore list
  ID NAME                SIZE AVA CLUSTERS IMAGES TYPE DS      TM      STAT
   2 files              104.8G 32% 0             1 fil  fs      ssh     on
   1 default            104.8G 32% 0             8 img  fs      ssh     on
   0 system                - -   0             0 sys  -       ssh     on
```

Информация о хранилище:

```
$ onedatastore show default
```

Создавать, включать, отключать, удалять и просматривать информацию о хранилищах можно в веб-интерфейсе (Рис. 15).

Работа с хранилищами в OpenNebula-Sunstone

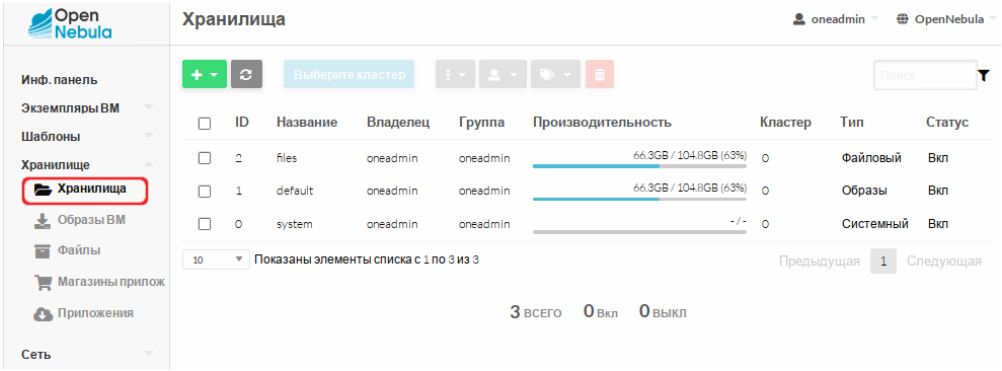


Рис. 15

3.6 Работа с образами в OpenNebula

Система хранилищ позволяет пользователям настраивать/устанавливать образы, которые могут быть образами ОС или данных, для использования в ВМ. Данные образы могут использоваться несколькими ВМ одновременно, а также предоставляться другим пользователями.

Типы образов для дисков ВМ (хранятся в хранилище образов):

– OS – образ загрузочного диска;
– CDROM – файл образа, содержащий CDROM. Эти образы предназначены только для чтения. В каждом шаблоне ВМ, можно использовать только один образ данного типа;
– DATABLOCK – файл образа, содержащий блок данных, создаваемый как пустой блок.

Типы файлов (хранятся в файловом хранилище):

– KERNEL – файл, который будет использоваться в качестве ядра ВМ (kernels);
– RAMDISK – файл, для использования в качестве виртуального диска;
– CONTEXT – файл для включения в контекстный CD-ROM.

Образы могут работать в двух режимах:

– persistent (постоянные) – изменения, внесенные в такие образы, будут сохранены после завершения работы ВМ. В любой момент времени может быть только одна ВМ, использующая постоянный образ.
– non-persistent (непостоянный) – изменения не сохранятся после завершения работы ВМ. Непостоянные образы могут использоваться несколькими ВМ одновременно, поскольку каждая из них будет работать со своей собственной копией.

Управлять образами можно, используя команду `oneimage`. Также управлять образами можно в веб-интерфейсе, на вкладке «Образы ВМ» (Рис. 16).

Управление образами в OpenNebula-Sunstone

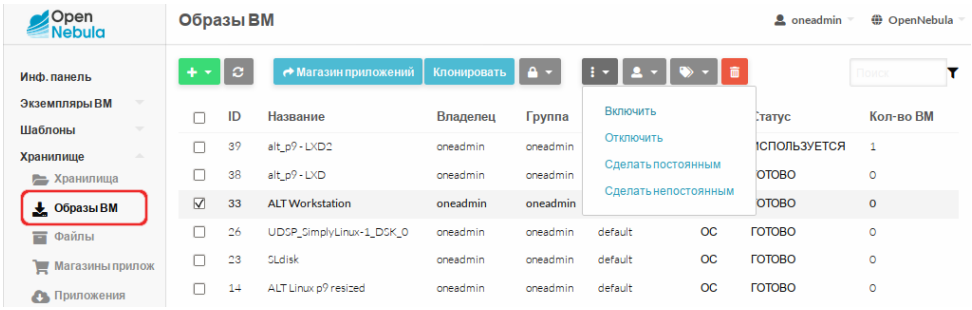


Рис. 16

3.6.1 Работа с образами в OpenNebula

Для создания образа ОС, необходимо подготовить ВМ и извлечь её диск.

3.6.1.1 Создание образов дисков

Создать образ типа CDROM с установочным ISO-образом.

Для этого перейти в раздел «Хранилище» → «Образы ВМ», на загруженной странице нажать «+» и выбрать пункт «Создать» (Рис. 17).

Создание образа в OpenNebula-Sunstone

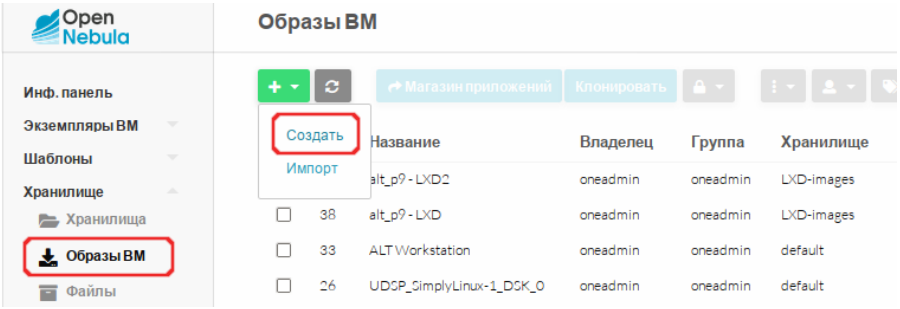


Рис. 17

В открывшемся окне заполнить поле «Название», выбрать в выпадающем списке «Тип» значение «CD-ROM только для чтения», выбрать хранилище, отметить в «Расположение образа» пункт «Путь на сервере OpenNebula» («Path in OpenNebula Server»), указать путь к файлу (.iso) и нажать кнопку «Создать» (Рис. 18).

Создание образа типа CD-ROM

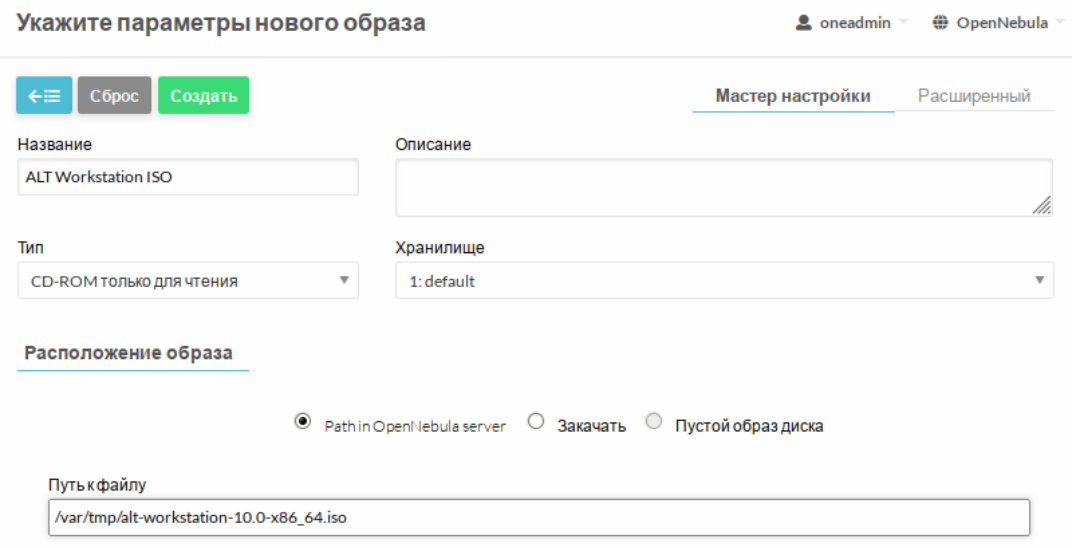


Рис. 18

Примечание. ISO-образ должен быть загружен в папку, к которой имеет доступ пользователь oneadmin.

Создать пустой образ диска, на который будет установлена операционная система.

Для этого создать новый образ. Заполнить поле «Название», в выпадающем списке «Тип» выбрать значение «Generic storage datablock», в выпадающем списке «Этот образ является постоянным» выбрать значение «Да», выбрать хранилище, в разделе «Расположение образа» выбрать пункт «Пустой образ диска», установить размер выбранного блока, например 45GB, в разделе «Расширенные настройки» указать драйвер «qcow2» и нажать кнопку «Создать» (Рис. 19).

Создание диска

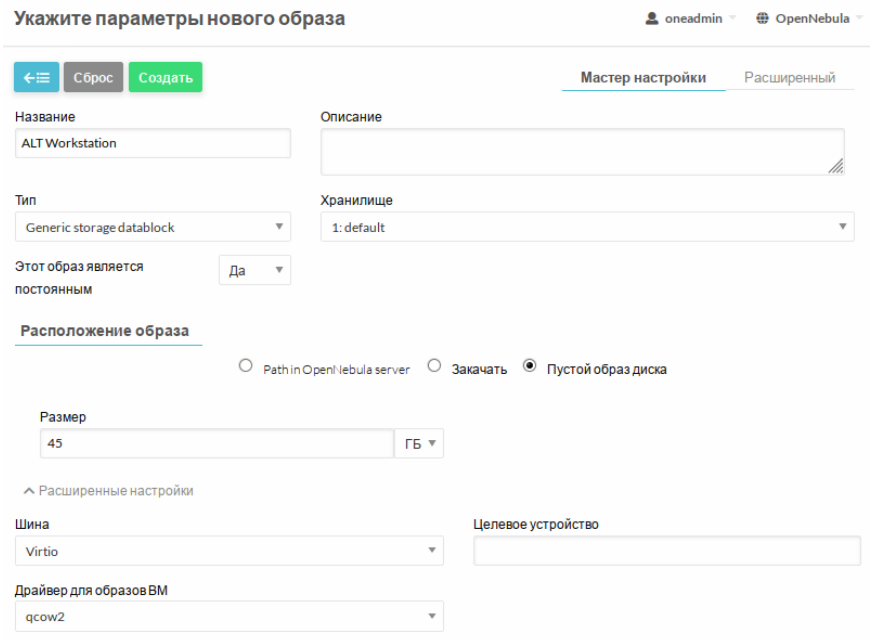


Рис. 19

Эти же действия можно выполнить в командной строке.

Создать образ типа CDROM в хранилище данных по умолчанию (ID = 1):

```
$ oneimage create -d 1 --name "ALT Workstation ISO" --path /var/tmp/alt-workstation-10.0-x86_64.iso --type CDROM
ID: 31
```

Создать пустой образ диска (тип образа – DATABLOCK, размер 45 ГБ, драйвер qcow2):

```
$ oneimage create -d 1 --name "ALT Workstation" --type DATABLOCK --size 45G --persistent --driver qcow2
ID: 33
```

3.6.1.2 Создание шаблона ВМ

Создание шаблона в командной строке:

1) Создать файл template со следующим содержимым:

```
NAME = "ALT Workstation"
CONTEXT = [
  NETWORK = "YES",
  SSH_PUBLIC_KEY = "$USER[SSH_PUBLIC_KEY]" ]
CPU = "0.25"
DISK = [
  IMAGE = "ALT Workstation ISO",
  IMAGE_UNAME = "oneadmin" ]
DISK = [
  DEV_PREFIX = "vd",
  IMAGE = "ALT Workstation",
  IMAGE_UNAME = "oneadmin" ]
GRAPHICS = [
  LISTEN = "0.0.0.0",
  TYPE = "SPICE" ]
HYPERVISOR = "kvm"
INPUTS_ORDER = ""
LOGO = "images/logos/alt.png"
MEMORY = "1024"
MEMORY_UNIT_COST = "MB"
NIC = [
  NETWORK = "VirtNetwork",
  NETWORK_UNAME = "oneadmin",
  SECURITY_GROUPS = "0" ]
NIC_DEFAULT = [
  MODEL = "virtio" ]
OS = [
  BOOT = "disk1,disk0" ]
SCHED_REQUIREMENTS = "ID=\"0\""
```

2) Создать шаблон:

```
$ onetemplate create template
ID: 22
```

Ниже рассмотрен пример создания шаблона в веб-интерфейсе.

Для создания шаблона ВМ, необходимо в левом меню выбрать «Шаблоны» → «ВМ» и на загруженной странице нажать кнопку «+» и выбрать пункт «Создать».

На вкладке «Общие» необходимо указать параметры процессора, оперативной памяти, а также гипервизор (Рис. 20).

Создание шаблона ВМ. Вкладка «Общие»

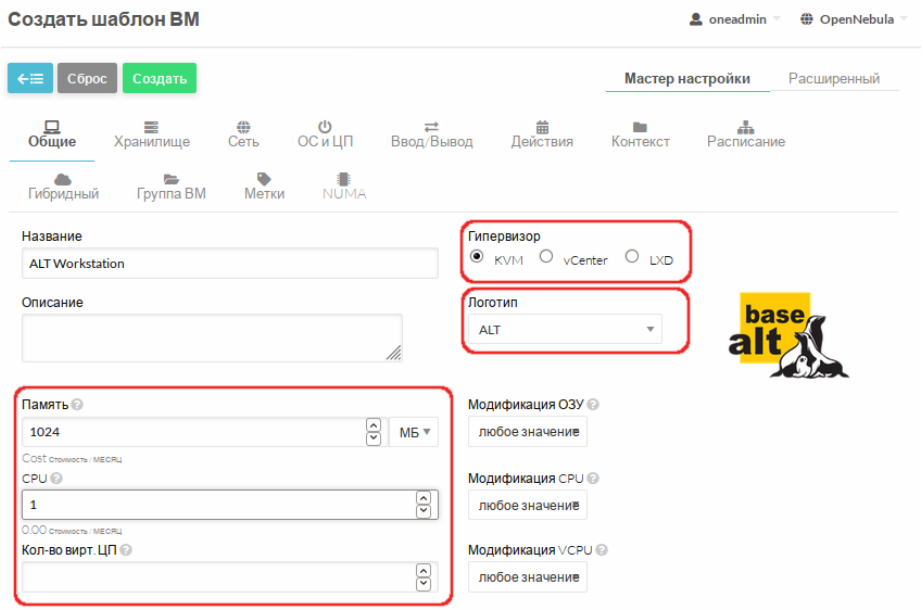


Рис. 20

На вкладке «Хранилище» необходимо указать ранее созданный пустой диск (DATABLOCK), в разделе «Расширенные настройки» в выпадающем списке «Шина» выбрать «Virtio». Далее следует добавить новый диск и указать диск с установщиком ОС (Рис. 21).

Создание шаблона ВМ. Вкладка «Хранилище»

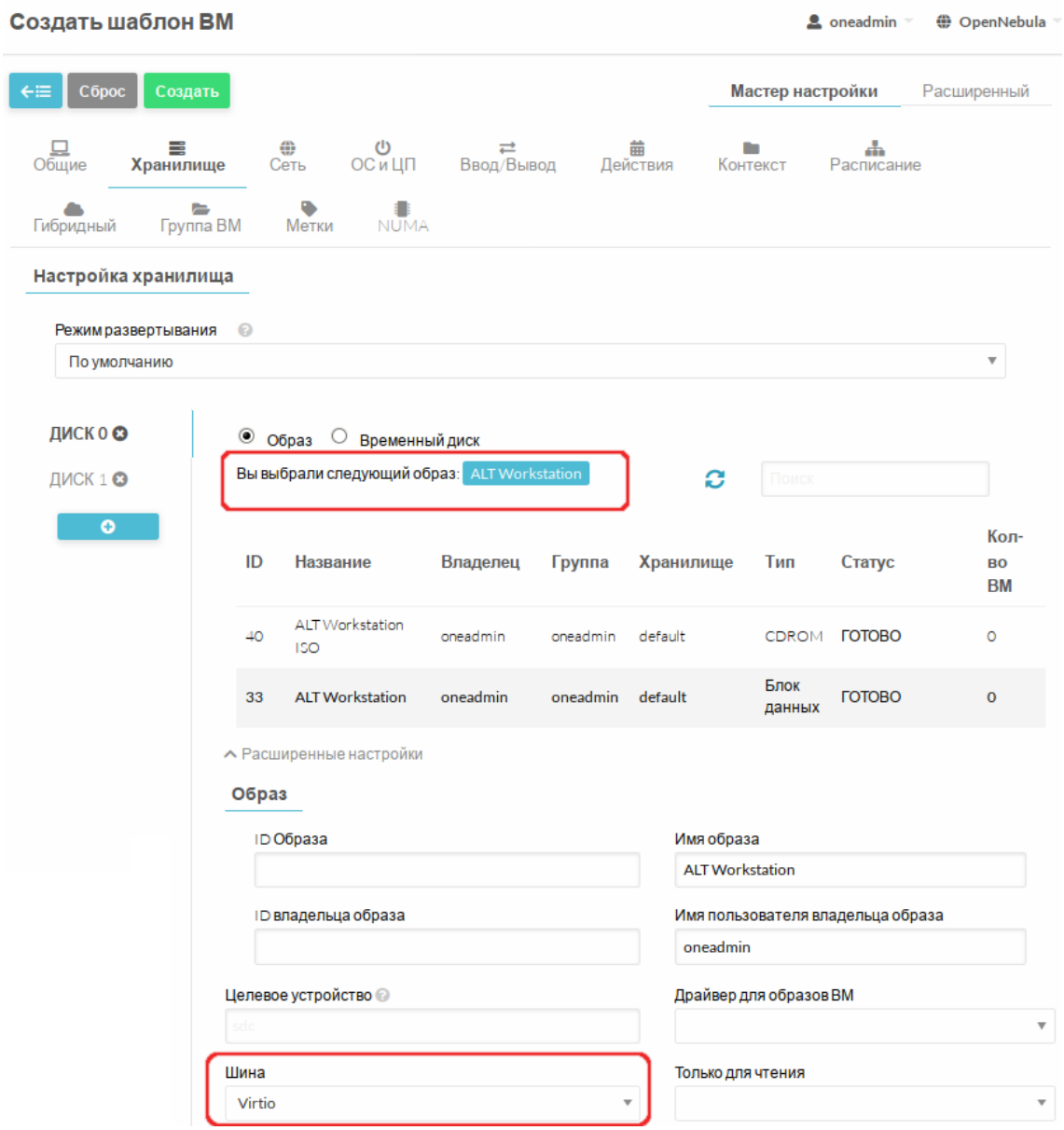


Рис. 21

На вкладке «Сеть» в поле «Default hardware model to emulate for all NICs» следует указать Virtio и если необходимо выбрать сеть (Рис. 22).

На вкладке «ОС и ЦПУ» необходимо указать архитектуру устанавливаемой системы и выбрать порядок загрузки. Можно установить в качестве первого загрузочного устройства – пустой диск (DATABLOCK), а в качестве второго – CDROM (Рис. 23). При такой последовательности загрузочных устройств при пустом диске загрузка произойдёт с CDROM, а в дальнейшем, когда ОС будет уже установлена на диск, загрузка будет осуществляться с него.

Создание шаблона ВМ. Вкладка «Сеть»

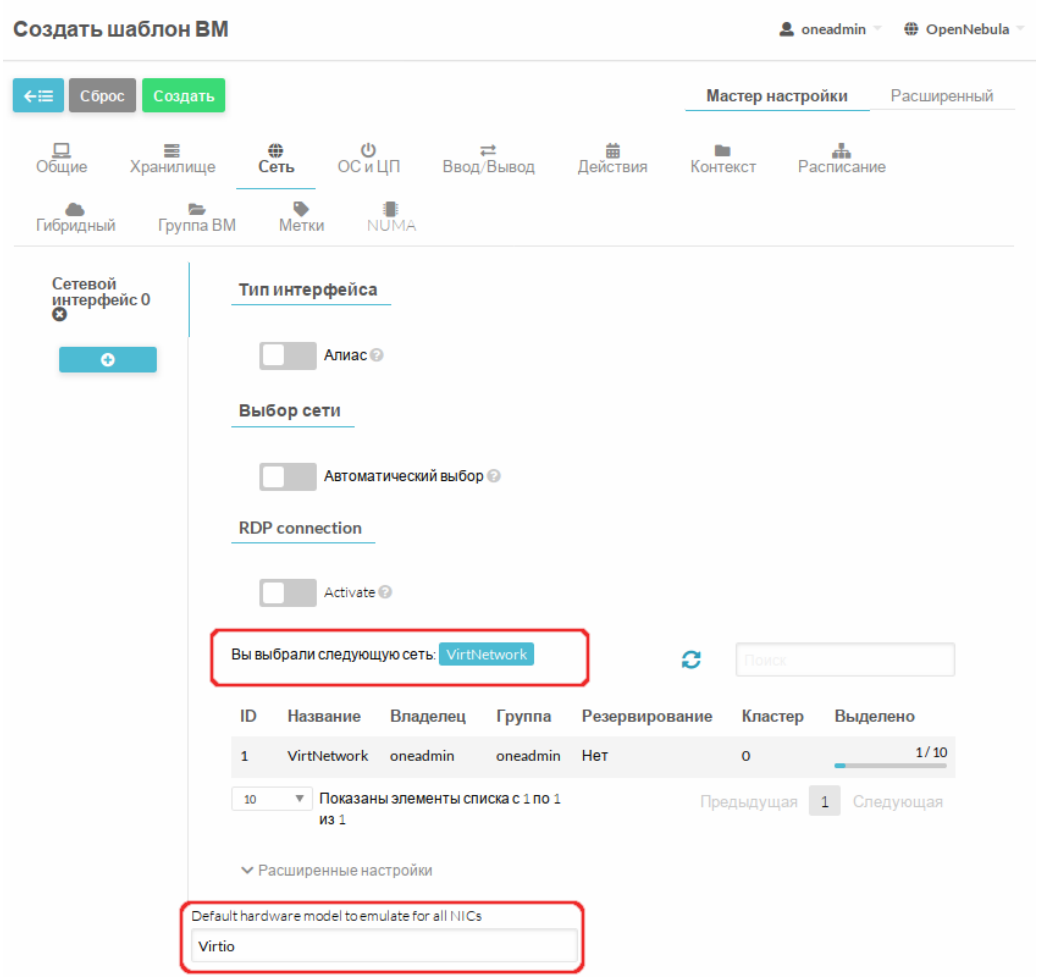


Рис. 22

Создание шаблона ВМ. Вкладка «ОС и ЦПУ»

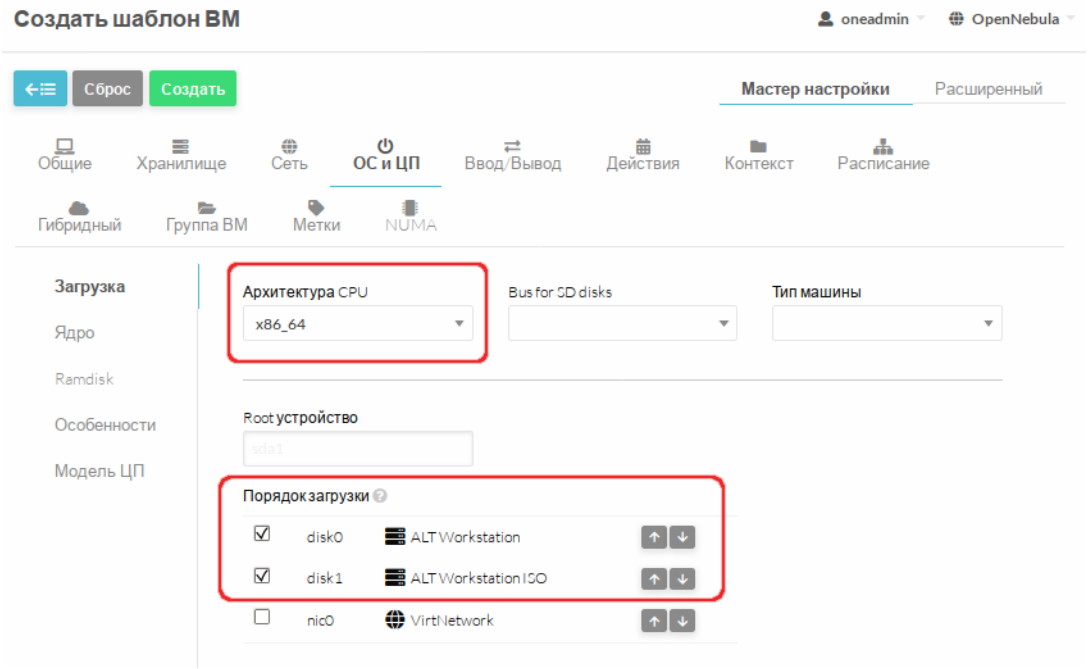


Рис. 23

На вкладке «Ввод/Вывод» следует включить «SPICE» (Рис. 24).

Создание шаблона ВМ. Вкладка «Ввод/Вывод»

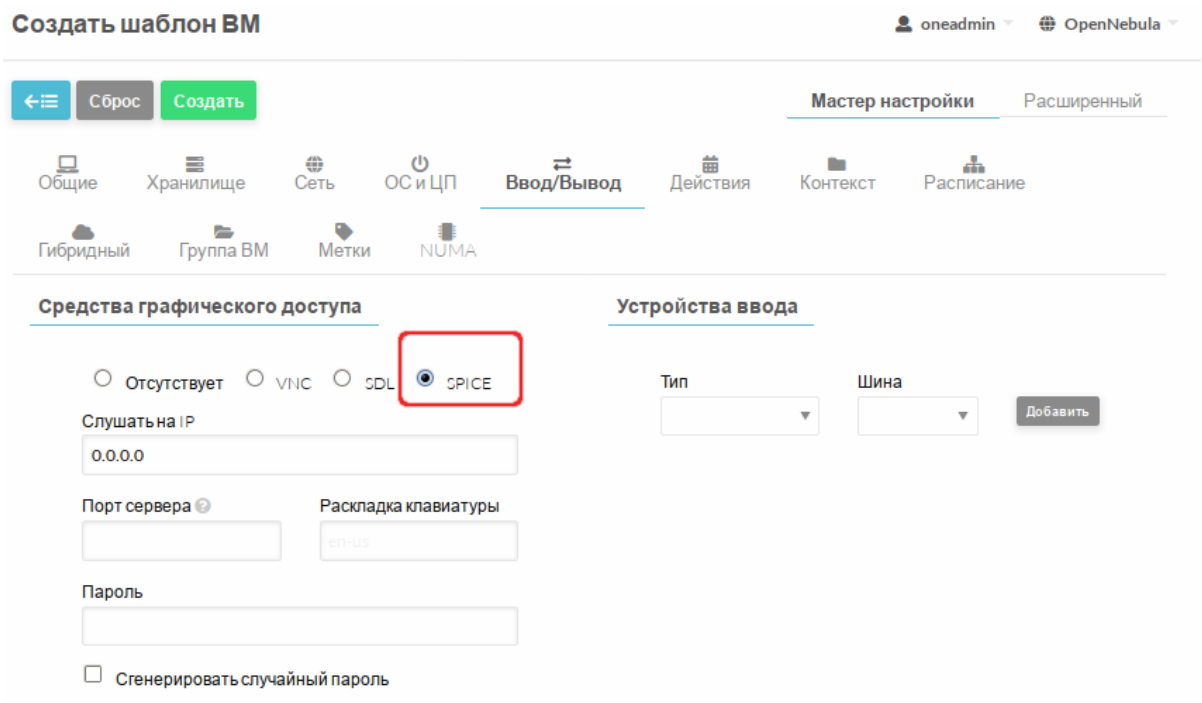


Рис. 24

На вкладке «Контекст» необходимо включить параметр «Использовать сетевое задание контекста», а также авторизацию по RSA-ключам (Рис. 25). Укажите открытый SSH (.pub) для доступа к ВМ по ключу. Если оставить это поле пустым, будет использована переменная $USER[SSH_PUBLIC_KEY].

Создание шаблона ВМ. Вкладка «Контекст»

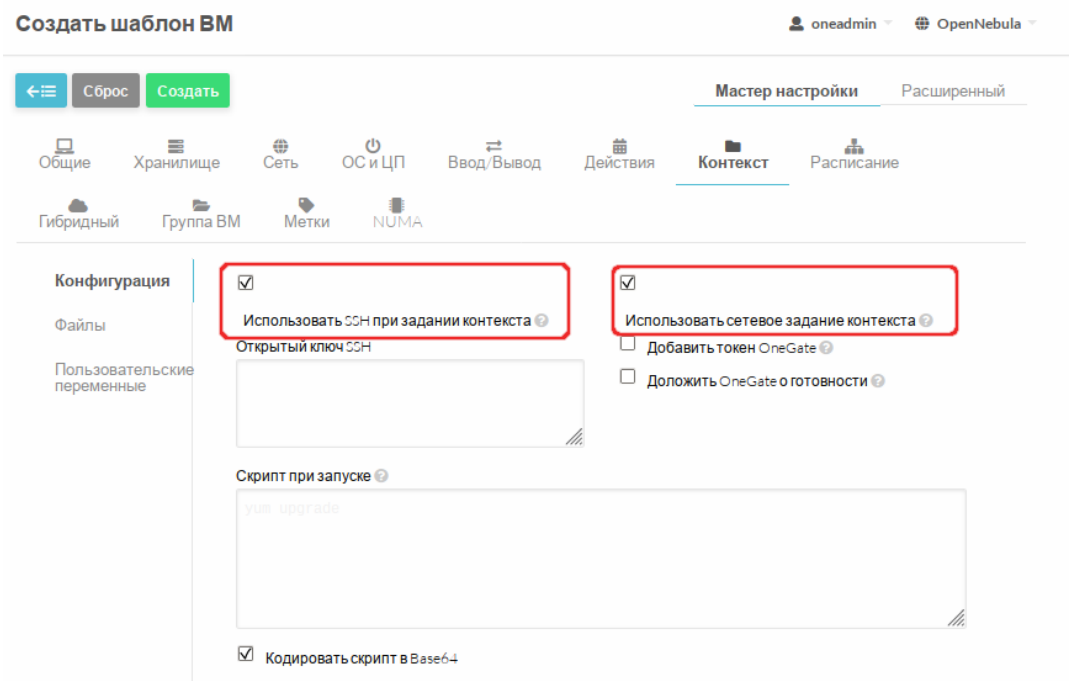


Рис. 25

На вкладке «Расписание» если необходимо можно выбрать кластер/хост, на котором будет размещаться виртуальное окружение (Рис. 26).

Создание шаблона ВМ. Вкладка «Расписание»

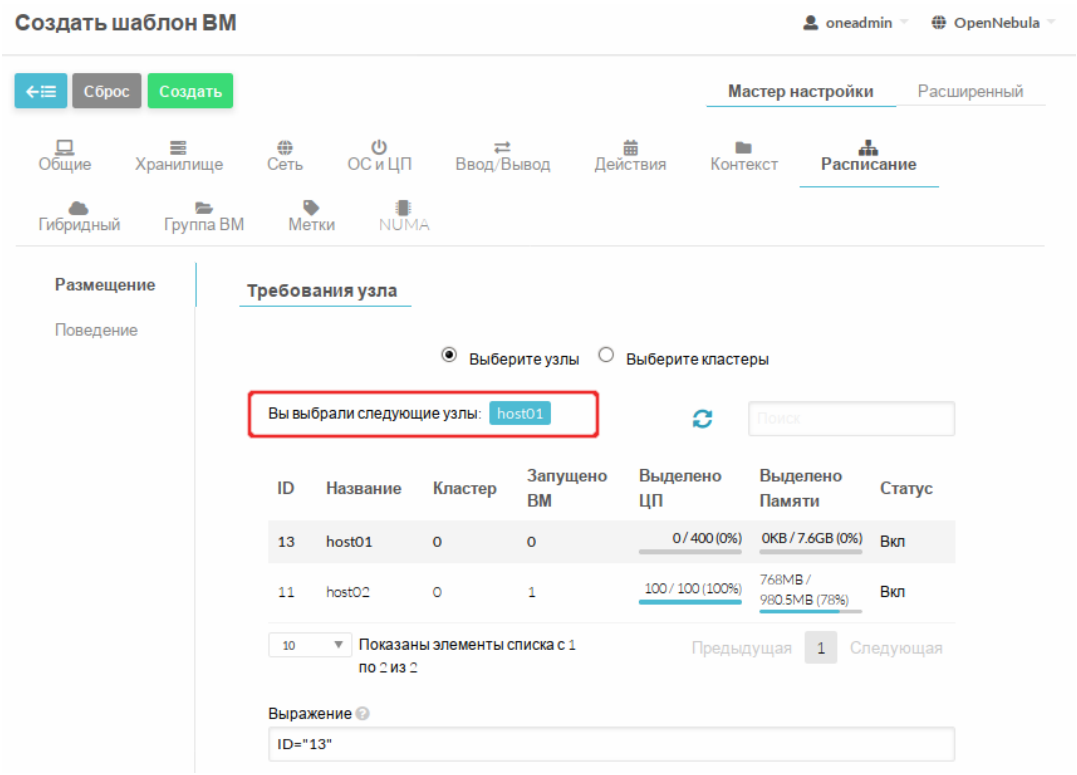


Рис. 26

Для создания шаблона ВМ нажать кнопку «Создать».

3.6.1.3 Создание ВМ

Для инициализации установки ОС из созданного шаблона в левом меню следует выбрать пункт «Шаблоны» → «ВМ», выбрать шаблон и нажать кнопку «Создать ВМ» (Рис. 27).

Создание экземпляра ВМ из шаблона

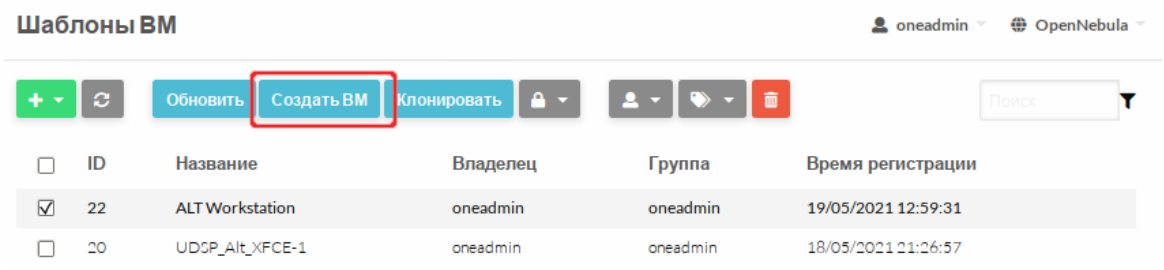


Рис. 27

В открывшемся окне необходимо указать имя ВМ и нажать кнопку «Создать ВМ» (Рис. 28).

Инициализация установки ОС из шаблона

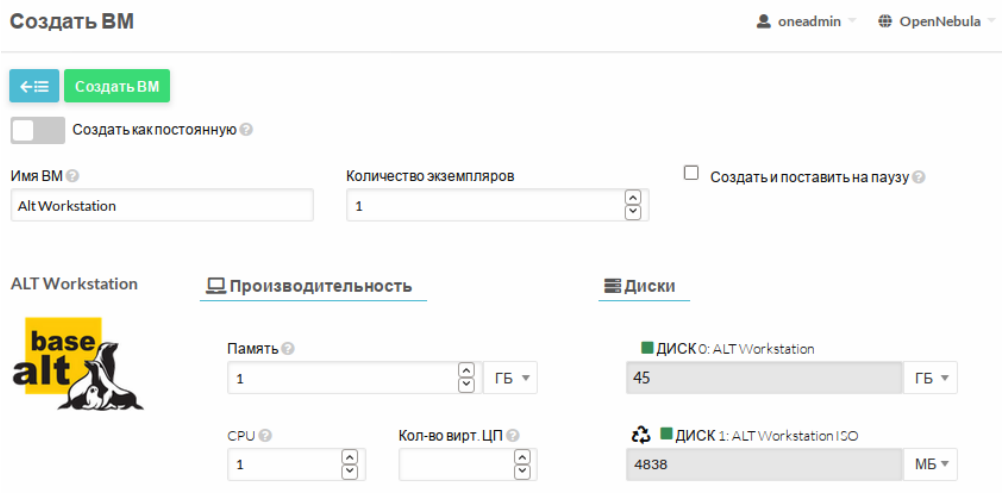


Рис. 28

Создание экземпляра ВМ из шаблона в командной строке:

```
$ onetemplate instantiate 22
VM ID: 9
```

3.6.1.4 Подключение к ВМ и установка ОС

Примечание. Процесс создания ВМ может занять несколько минут. Следует дождаться статуса – «ЗАПУЩЕНО» («RUNNING»).

Подключиться к ВМ можно как из веб-интерфейса Sunstone, раздел «Экземпляры ВМ» → «ВМ» выбрать ВМ и подключиться по SPICE (Рис. 29).

Подключение к ВМ

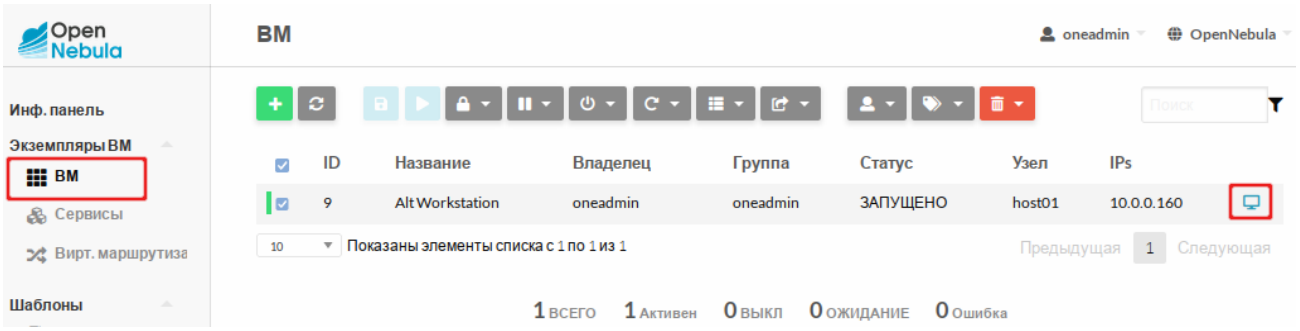


Рис. 29

Так и используя, любой клиент SPICE:

```
spice://192.168.0.190:5909
```

где 192.168.0.190 – IP-адрес хоста с ВМ, а 9 – идентификатор ВМ (номер порта 5900 + 9).

Далее необходимо провести установку системы (Рис. 30).

Установка ВМ

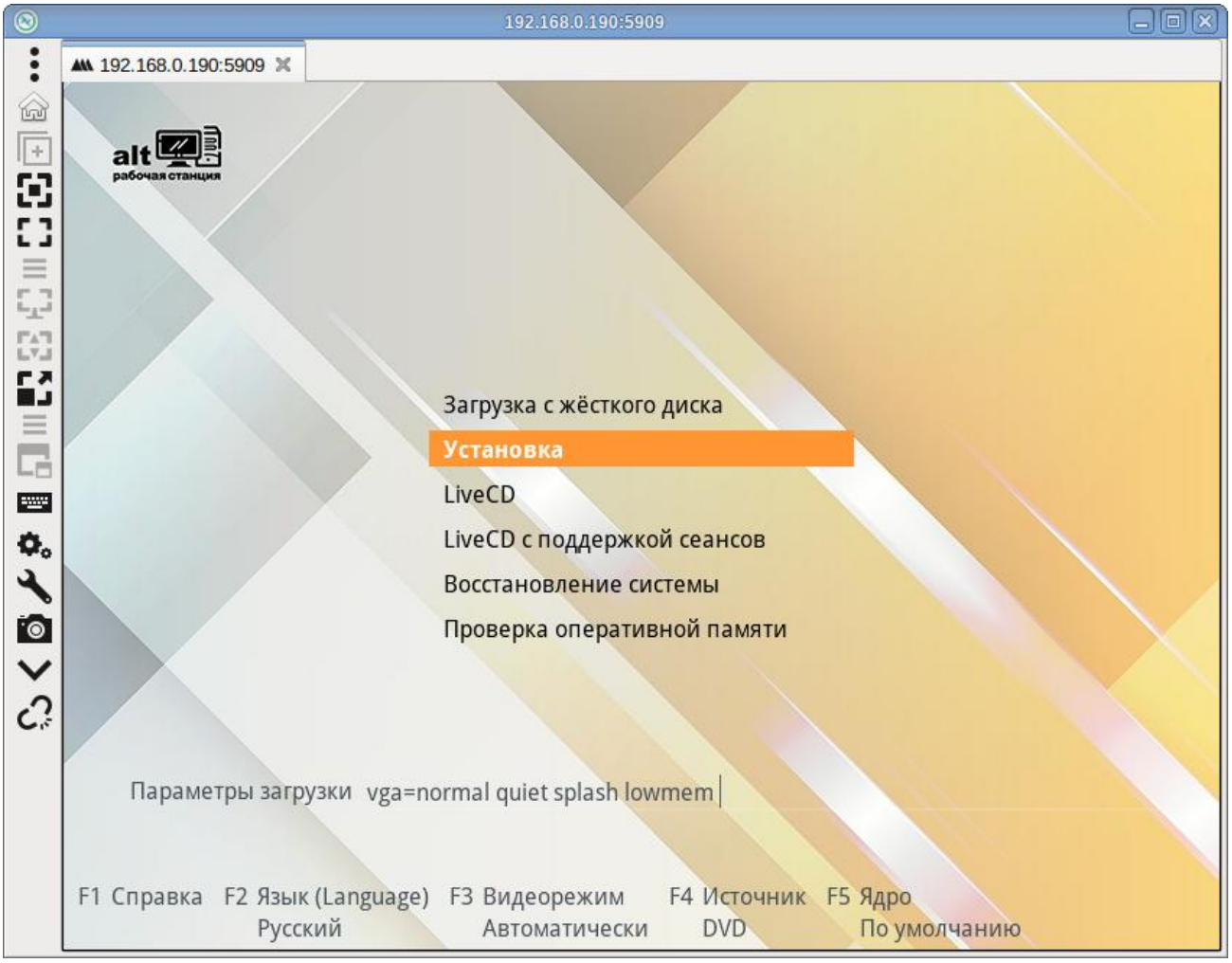


Рис. 30

3.6.1.5 Настройка контекстуализации

OpenNebula использует метод, называемый контекстуализацией, для отправки информации на ВМ во время загрузки. Контекстуализация позволяет установить или переопределить данные ВМ, имеющие неизвестные значения или значения по умолчанию (имя хоста, IP-адрес, .ssh/authorized_keys).

Пример настройки контекстуализации на установленной ОС Альт:

1) Подключиться к ВМ через SPICE или по ssh.

2) Установить пакет opennebula-context:

```
# apt-get update && apt-get install opennebula-context
```

3) Переключиться на systemd-networkd:

– установить пакет systemd-timesyncd:

```
# apt-get install systemd-timesyncd
```

– создать файл автонастройки всех сетевых интерфейсов по DHCP `/etc/systemd/network/lan.network` со следующим содержимым:

```
[Match]
Name = *

[Network]
DHCP = ipv4
```

– переключиться с etcnet/NetworkManager на systemd-networkd:

```
# systemctl disable network NetworkManager && systemctl enable systemd-networkd systemd-timesyncd
```

4) Перезагрузить систему.

После перезагрузки доступ в систему будет возможен по ssh-ключу, ВМ будет назначен IP-адрес, который OpenNebula через механизм IPAM (подсистема IP Address Management) выделит из пула адресов.

3.6.1.6 Создание образа типа ОС

После завершения установки системы следует выключить и удалить ВМ. Диск находится в состоянии «Persistent», поэтому все внесенные изменения будут постоянными.

Для удаления ВМ в левом меню следует выбрать пункт «Экземпляры ВМ» → «ВМ», выбрать ВМ и нажать кнопку «Уничтожить» (Рис. 31).

Удаление ВМ

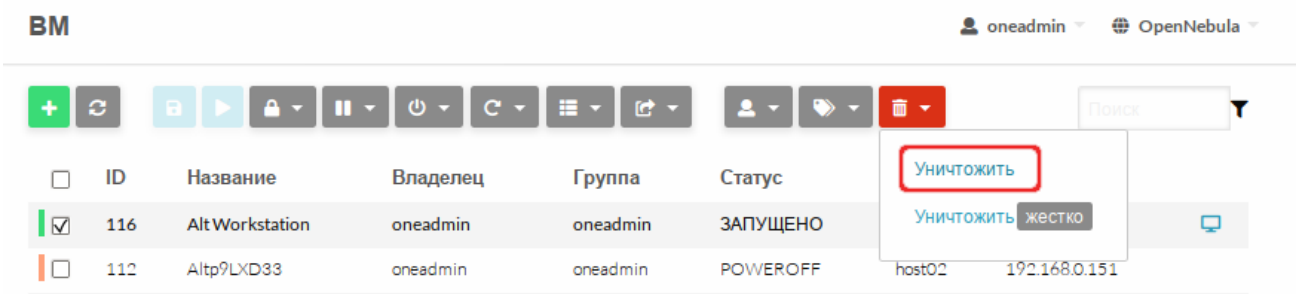


Рис. 31

Примечание. Удаление ВМ в командной строке:

```
$ onevm terminate 9
```

Затем перейти в «Хранилище» → «Образы ВМ», выбрать образ с установленной ОС (ALT Workstation) и изменить тип блочного устройства с «Блок данных» на «ОС» и состояние на «Не постоянный» (Рис. 32).

Примечание. Изменить тип блочного устройства на OS и состояние на Non Persistent в командной строке:

```
$ oneimage chtype 33 OS
$ oneimage nonpersistent 33
```

Образ готов. Далее можно использовать как имеющийся шаблон, так и создать новый на основе образа диска «ALT Workstation».

Изменение типа блочного устройства

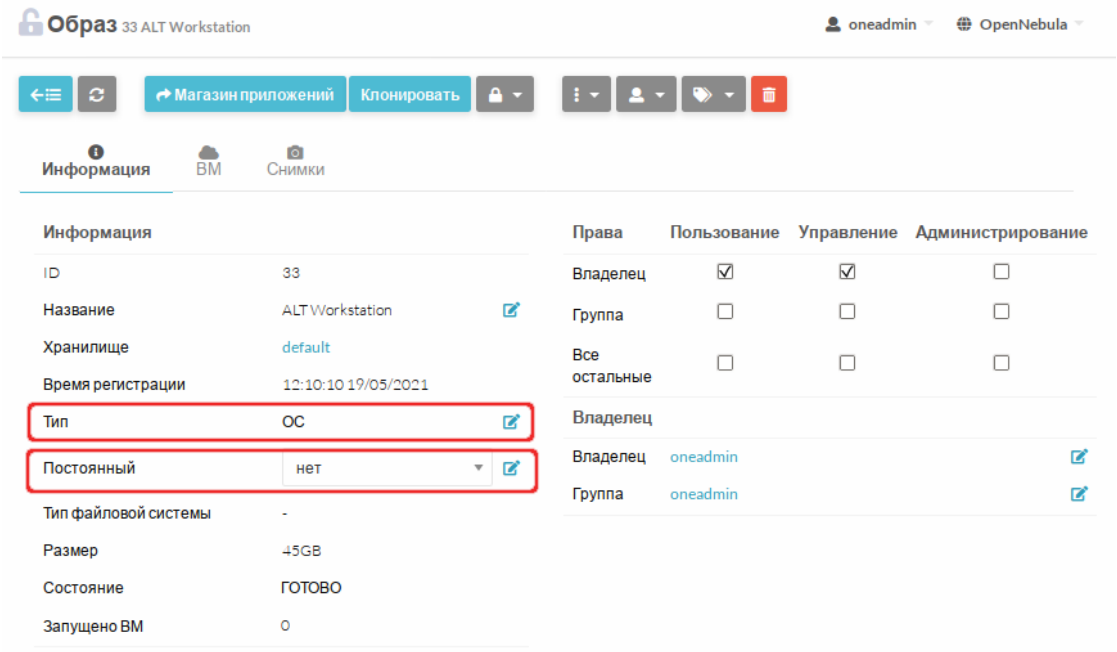


Рис. 32

3.6.2 Использование магазина приложений OpenNebula

Для загрузки приложения из магазина необходимо перейти в «Хранилище» → «Магазины приложений», выбрать «OpenNebula Public» → «Приложения». Появится список доступных приложений (Рис. 33).

Магазин приложений OpenNebula

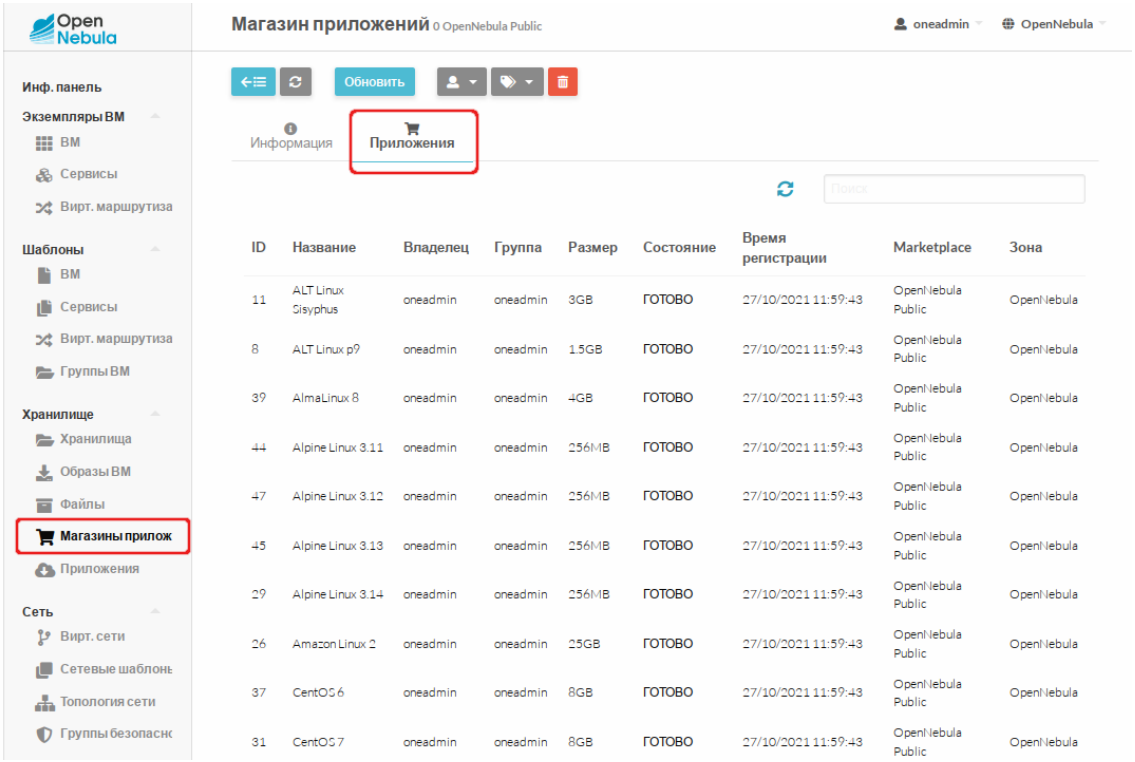


Рис. 33

Каждое приложение содержит образ и шаблон.

Чтобы импортировать приложение, необходимо его выбрать и нажать кнопку «Import into Datastore» (Рис. 34).

Импорт приложения из магазина приложений OpenNebula

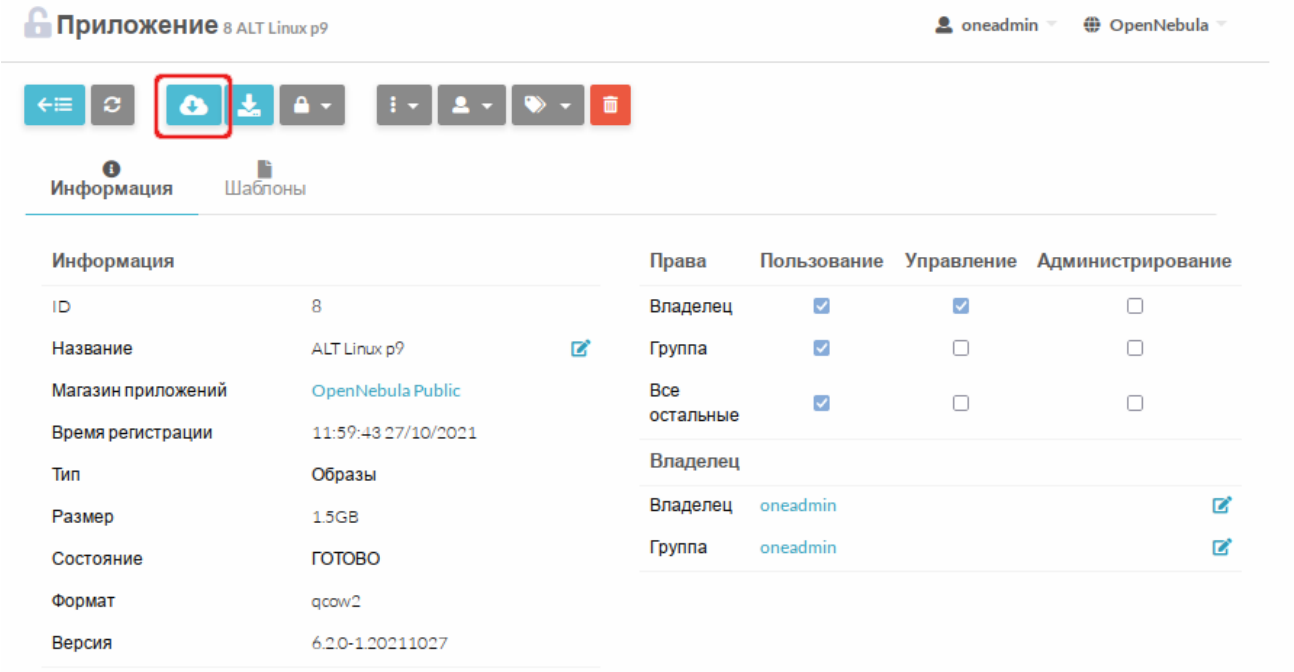


Рис. 34

В открывшемся окне указать имя для образа и шаблона, выбрать хранилище и нажать кнопку «Загрузить» (Рис. 35).

Изменение типа блочного устройства

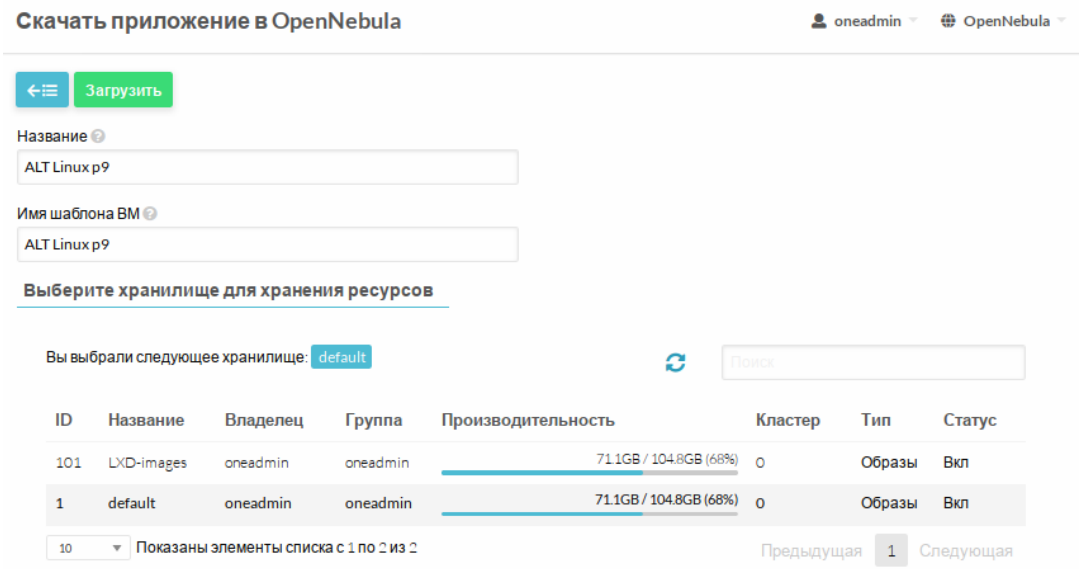


Рис. 35

Настройка образов, загруженных из магазина приложений:

1) изменить состояние образа на «Постоянный» (необходимо дождаться состояния «Готово»);

2) настроить шаблон;

3) создать на основе шаблона ВМ;

4) подключиться к ВМ. Установить/настроить необходимые компоненты;

5) удалить ВМ;

6) изменить состояние образа на «Не постоянный»;

7) далее можно создать новые шаблоны на основе этого образа или использовать существующий.

3.6.3 Скачивание шаблона контейнера из магазина приложений

Для загрузки контейнера из магазина необходимо перейти в «Хранилище» → «Магазины приложений, выбрать «Linux Containers» → «Приложения» (Рис. 36).

Выбрать LXD образ. Чтобы импортировать контейнер, необходимо его выбрать и нажать кнопку «Import into Datastore» (Рис. 37).

Каждый контейнер содержит образ и шаблон.

В открывшемся окне указать название для образа и шаблона, выбрать хранилище и нажать кнопку «Загрузить» (Рис. 38).

Магазин приложений OpenNebula

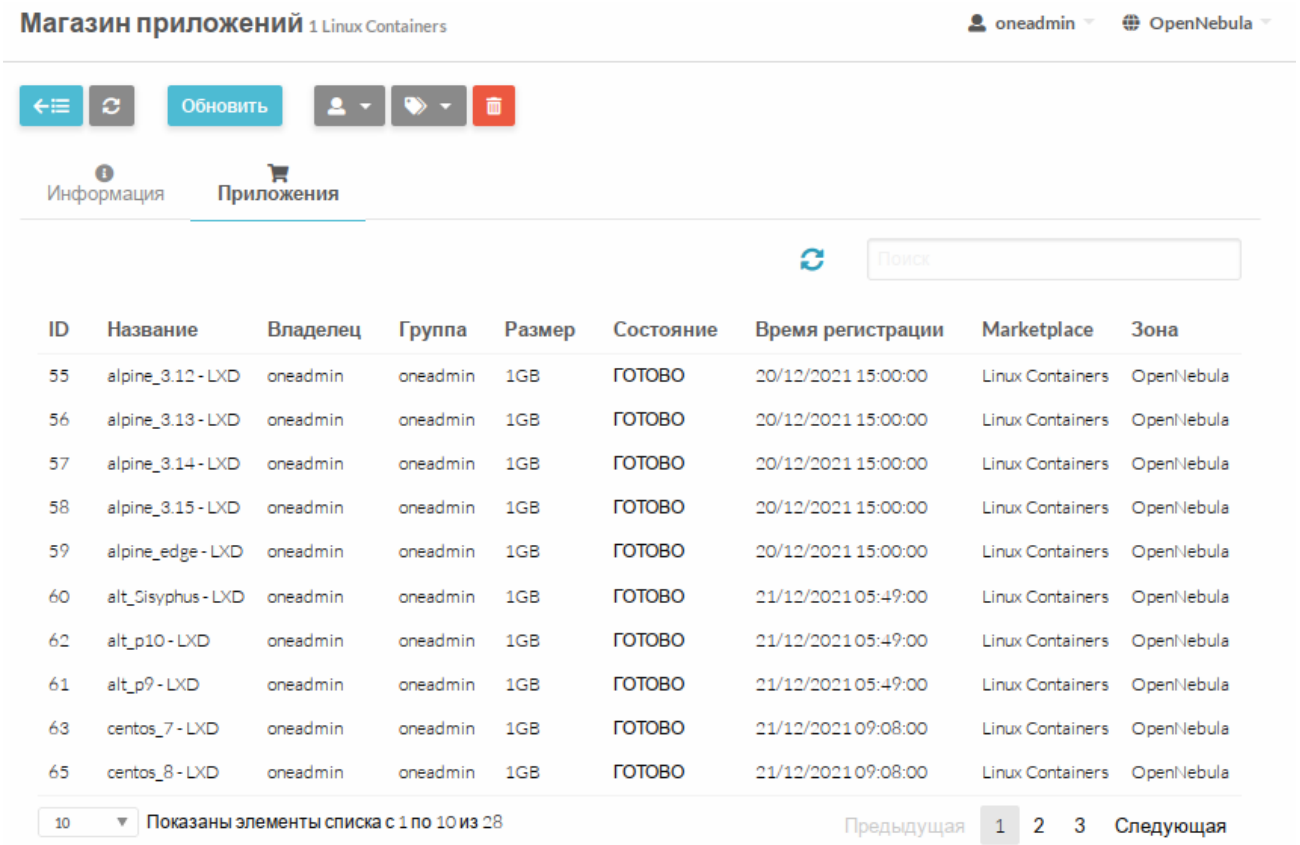


Рис. 36

Импорт контейнера из магазина приложений OpenNebula

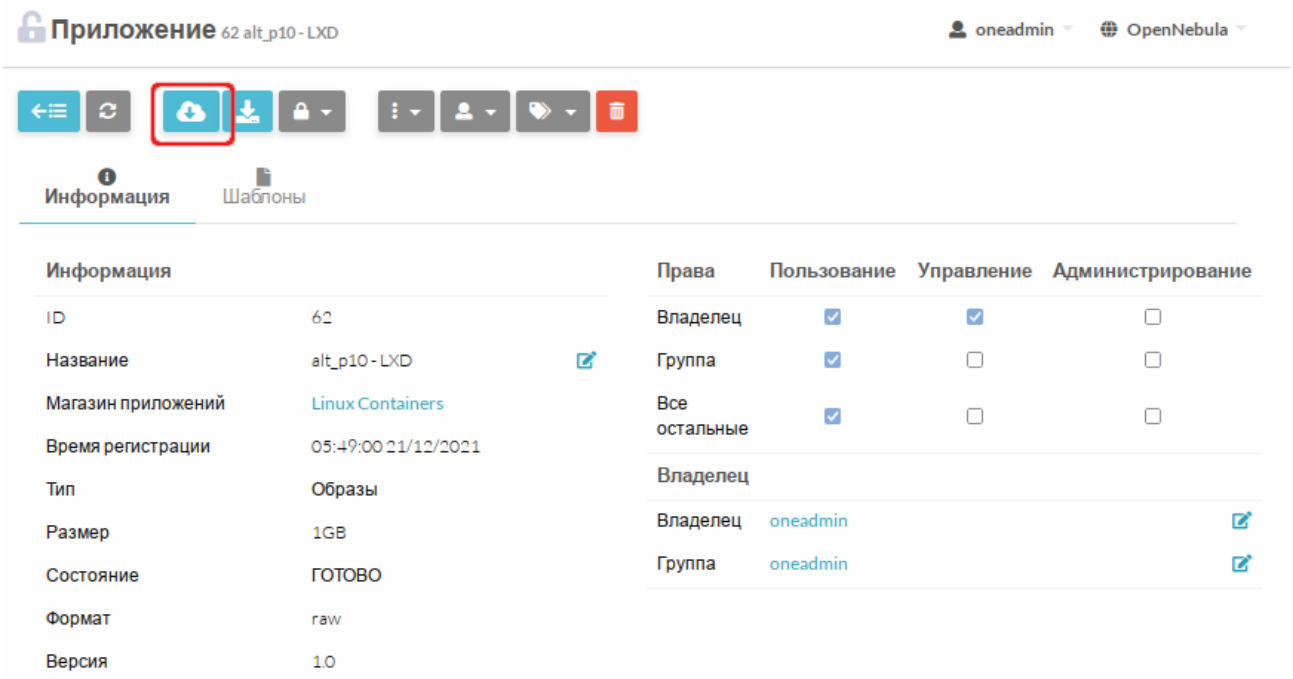


Рис. 37

Импорт контейнера из магазина приложений OpenNebula

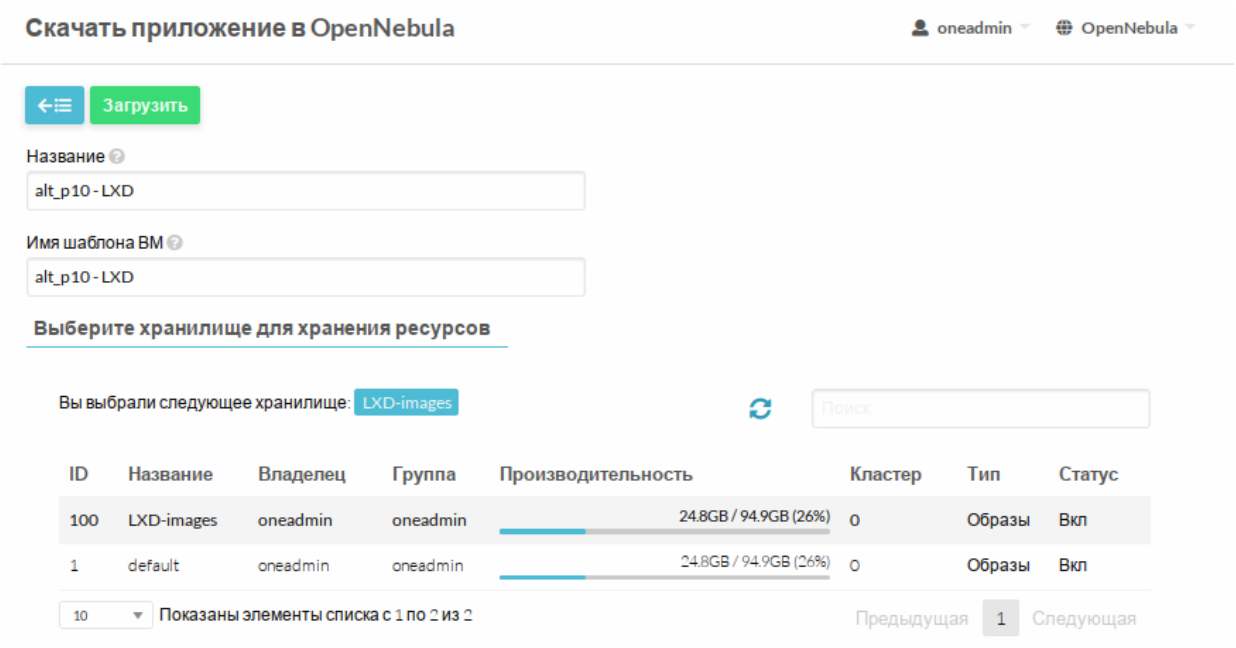


Рис. 38

Из полученного шаблона можно разворачивать контейнеры (ВМ в терминологии Opennebula). Процесс разворачивания контейнера из шаблона такой же, как и процесс разворачивания ВМ из шаблона (Рис. 39).

Разворачивание контейнера из шаблона

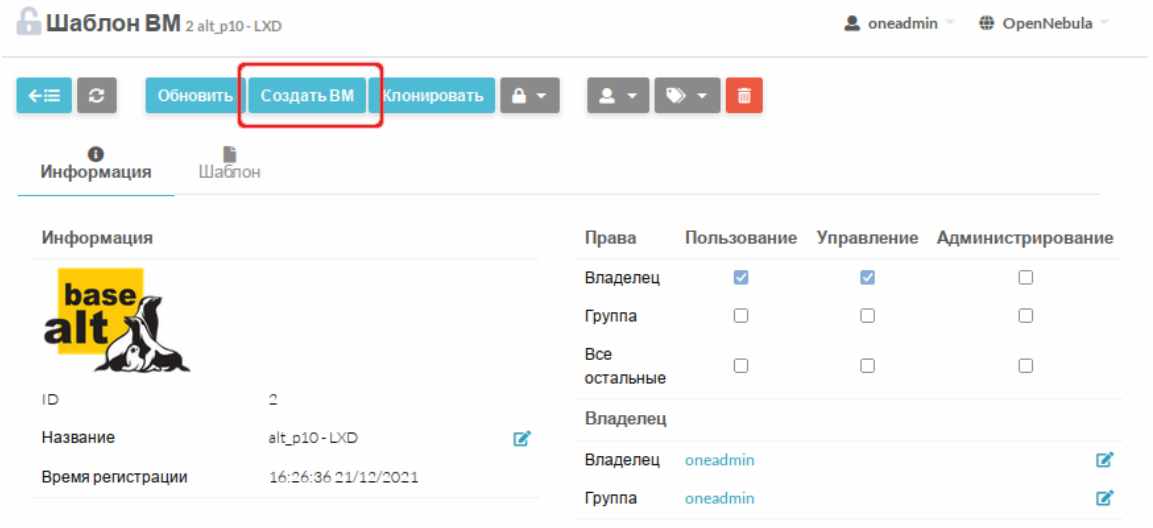


Рис. 39

3.7 Управление пользователями

OpenNebula поддерживает учётные записи пользователей и группы.

Ресурсы, к которым пользователь может получить доступ в OpenNebula, контролируются системой разрешений. По умолчанию только владелец ресурса может использовать и управлять им. Пользователи могут делиться ресурсами, предоставляя разрешения на использование или управление другим пользователям в своей группе или любому другому пользователю в системе.

3.7.1 Пользователи

Пользователь в OpenNebula определяется именем пользователя и паролем. Каждый пользователь имеет уникальный идентификатор и принадлежит как минимум к одной группе.

При установке OpenNebula создаются две административные учетные записи (oneadmin и serveradmin).

oneuser – инструмент командной строки для управления пользователями в OpenNebula.

Посмотр списка пользователей:

```
$ oneuser list
  ID NAME                GROUP    AUTH            VMS      MEMORY        CPU
   1 serveradmin         oneadmin server_c    0 /   -       0M /   0.0 /   -
   0 oneadmin            oneadmin core              -          -           -
```

Создание нового пользователя:

```
$ oneuser create <user_name> <password>
```

По умолчанию новый пользователь будет входить в группу users. Изменить группу пользователя можно, выполнив команду:

```
$ oneuser chgrp <user_name> oneadmin
```

Что бы удалить пользователя из группы, необходимо переместить его обратно в группу users.

Удалить пользователя:

```
$ oneuser delete <user_name>
```

Все операции с пользователями можно производить в веб-интерфейсе (Рис. 40).

Пользователь может аутентифицироваться в веб-интерфейсе OpenNebula и изменить настройки (изменить язык интерфейса, пароль, добавить ssh-ключ для доступа на ВМ и т.д.) (Рис. 41).

Примечание. Пользователи могут просматривать информацию о своей учётной записи и изменять свой пароль.

Управление пользователями в OpenNebula-Sunstone

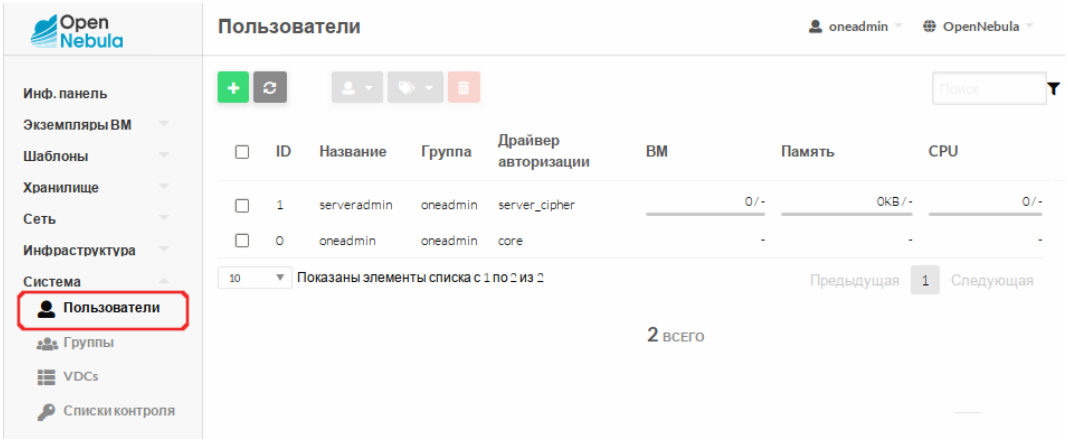


Рис. 40

Панель пользователя в OpenNebula-Sunstone

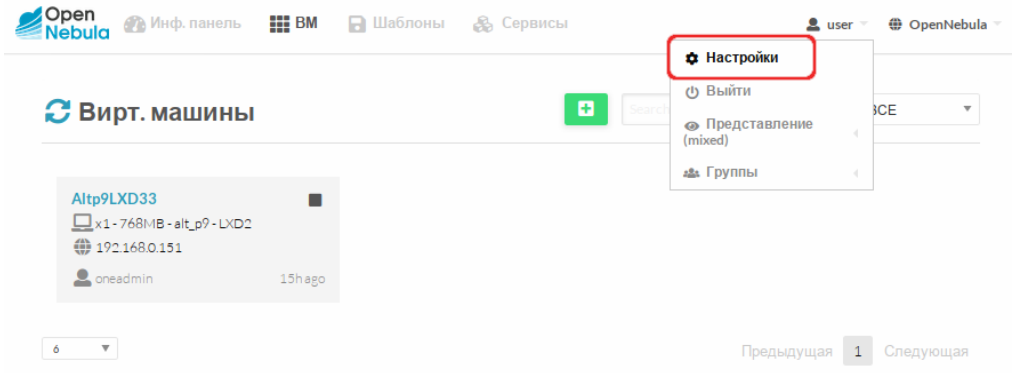


Рис. 41

3.7.2 Группы пользователей

При установке OpenNebula создаются две группы (oneadmin и users).

onegroup – инструмент командной строки для управления группами в OpenNebula.

Просмотр списка групп:

```
$ onegroup list
```

```
  ID NAME                 USERS       VMS              MEMORY          CPU
   1 users                    0    0 /   -       0M /        -   0.0 /    -
   0 oneadmin                 2          -                   -            -
```

Создание новой группы:

```
$ onegroup create group_name
ID: 100
```

Новая группа получила идентификатор 100, чтобы отличать специальные группы от групп, созданных пользователем.

После создания группы может быть создан связанный пользователь-администратор. По умолчанию этот пользователь сможет создавать пользователей в новой группе.

Пример создания новой группы с указанием, какие ресурсы могут быть созданы пользователями группы (по умолчанию VM+IMAGE+TEMPLATE):

```
$ onegroup create --name testgroup \
--admin_user testgroup-admin --admin_password somestr \
--resources TEMPLATE+VM
```

При выполнении данной команды также будет создан администратор группы.

Сделать существующего пользователя администратором группы:

```
$ onegroup addadmin <groupid_list> <userid>
```

Все операции с группами можно производить в веб-интерфейсе (Рис. 42).

Создание группы в OpenNebula-Sunstone

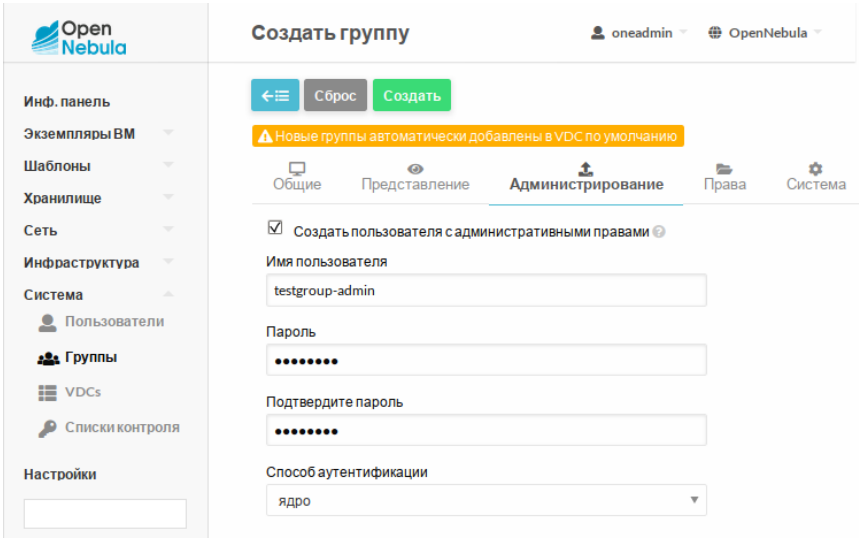


Рис. 42

3.7.3 Управление разрешениями

У ресурсов OpenNebula (шаблонов, ВМ, образов и виртуальных сетей) есть права назначенные владельцу, группе и всем остальным. Для каждой из этих групп можно установить три права: «Использование» (use), «Управление» (manage) и «Администрирование» (admin).

Просмотреть/изменить права доступа можно в веб-интерфейсе, выбрав соответсвующий ресурс (Рис. 43).

Управление разрешениями в OpenNebula-Sunstone

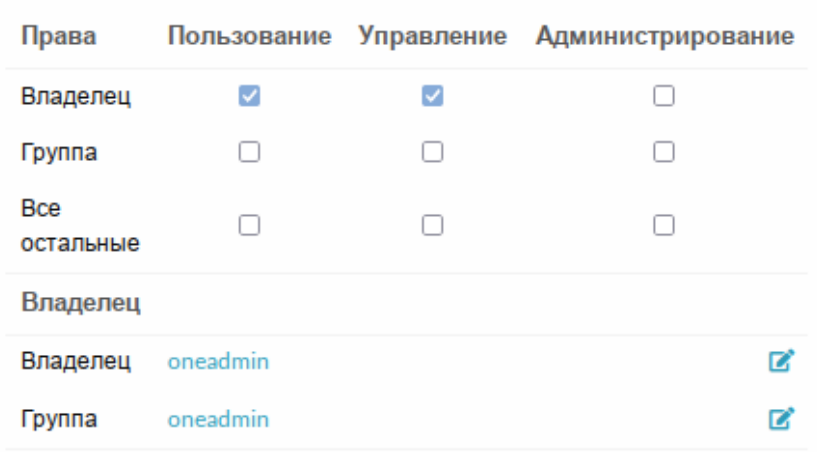


Рис. 43

Просмотреть права можно и в командной строке:

```
$ onetemplate show 3
TEMPLATE 3 INFORMATION
ID             : 3
NAME           : SL
USER           : oneadmin
GROUP          : oneadmin
LOCK           : None
REGISTER TIME  : 08/25 15:02:02

PERMISSIONS
OWNER          : um-
GROUP          : ---
OTHER          : ---
…
```

В данном примере показаны права на шаблон с ID=3.

Для изменения прав в командной строке используется команда `chmod`. Права записываются в числовом формате. Пример изменения прав:

```
$ onetemplate chmod 3 607
$ onetemplate show 3
…
PERMISSIONS
OWNER          : um-
GROUP          : ---
OTHER          : uma
```

Примечание. Разрешения по умолчанию для создаваемых ресурсов могут быть установлены:

- глобально, в `oned.conf` (атрибут DEFAULT_UMASK);
- индивидуально для каждого пользователя с помощью команды `oneuser umask`.

Маска должна состоять из 3 восьмеричных цифр. Каждая цифра – это маска, которая, соответственно, отключает разрешение для владельца, группы и всех остальных. Например, если значение маски равно 137, то для нового объекта будут установлены права 640 (um- u-- ---).

3.7.4 Управление правилами ACL

Система правил ACL позволяет точно настроить разрешенные операции для любого пользователя или группы пользователей. Каждая операция генерирует запрос на авторизацию, который проверяется на соответствие зарегистрированному набору правил ACL. Затем ядро может дать разрешение или отклонить запрос.

Просмотреть список правил можно, выполнив команду:

```
$ oneacl list
  ID     USER RES_VHNIUTGDCOZSvRMAPt   RID OPE_UMAC  ZONE
   0       @1     V--I-T---O-S----P-     *     ---c     *
   1        *     ----------Z-------     *     u---     *
   2        *     --------------MA--     *     u---     *
   3       @1     -H----------------     *     -m--    #0
   4       @1     --N---------------     *     u---    #0
   5       @1     -------D----------     *     u---    #0
   6       #3     ---I--------------   #30     u---    #0
```

Данные правила соответсвуют следующим:

```
@1      VM+IMAGE+TEMPLATE+DOCUMENT+SECGROUP/*   CREATE  *
*       ZONE/*                                  USE     *
*       MARKETPLACE+MARKETPLACEAPP/*            USE     *
@1      HOST/*                                  MANAGE  #0
@1      NET/*                                   USE     #0
@1      DATASTORE/*                             USE     #0
#3      IMAGE/#30                                    USE     *
```

Первые шесть правил были созданы при начальной загрузке OpenNebula, а последнее – с помощью `oneacl`:

```
$ oneacl create "#3 IMAGE/#30 USE"
ID: 6
```

Столбцы в выводе `oneacl list`:

- ID – идентификатор правила;

- USER – пользователь. В этом поле может быть указан идентификатор пользователя (#), идентификатор группы (@) или все пользователи (*);
- Resources – тип ресурса, к которому применяется правило:
  - V – ВМ;
  - H – узел;
  - N – виртуальная сеть;
  - I – образ;
  - U – пользователь;
  - T – шаблон;
  - G – группа;
  - D – хранилище;
  - C – кластер;
  - O – документ;
  - Z – зона;
  - S – группа безопасности;
  - v – виртуальный дата центр (VDC);
  - R – виртуальный маршрутизатор;
  - M – магазин приложений;
  - A – приложение из магазина приложений;
  - P – группа ВМ;
  - t – шаблон виртуальной сети;
- RID – идентификатор ресурса. В этом поле может быть указан идентификатор отдельного объекта (#), группы (@) или кластера (%), или все объекты (*);
- Operations – разрешённые операции:
  - U – использовать;
  - M – управлять;
  - A – администрировать;
  - C – создавать;
- Zone – зоны, в которых применяется правило. В этом поле может быть указан идентификатор отдельной зоны (#), или всех зон (*).

Для удаления правила используется команда:

```
$ oneacl delete <ID>
```

Управлять правилами ACL удобно в веб-интерфейсе (Рис. 44).

Управление правилами ACL в OpenNebula-Sunstone

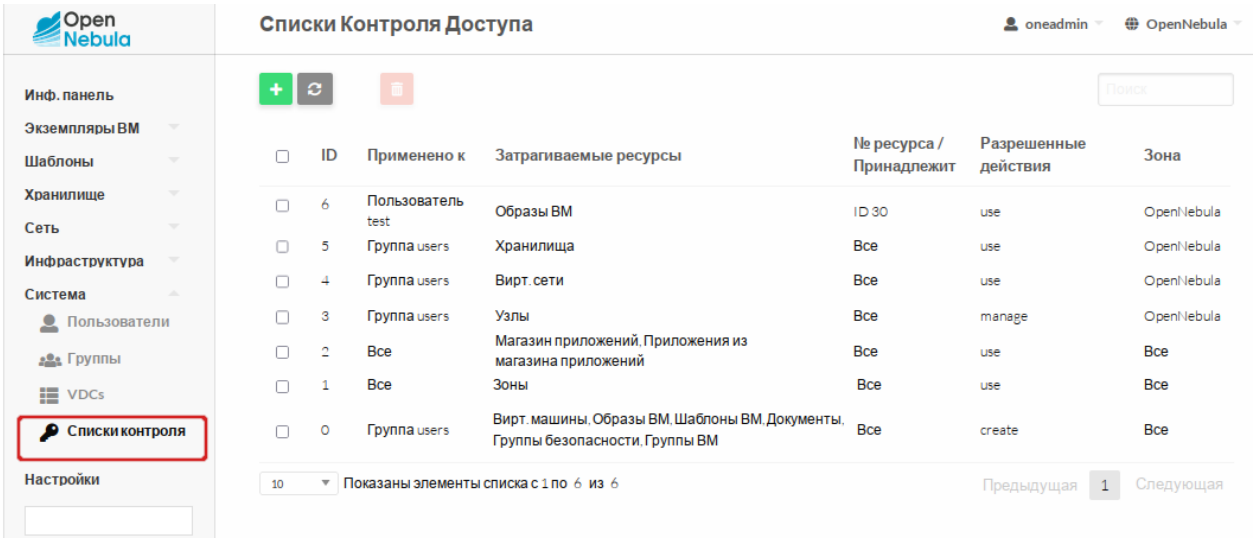


Рис. 44

Для создания нового правила ACL, следует нажать кнопку «Создать». В открывшемся диалоговом окне можно определить ресурсы, на которые распространяется правило, и разрешения которые им предоставляются (Рис. 45).

Управление правилами ACL в OpenNebula-Sunstone

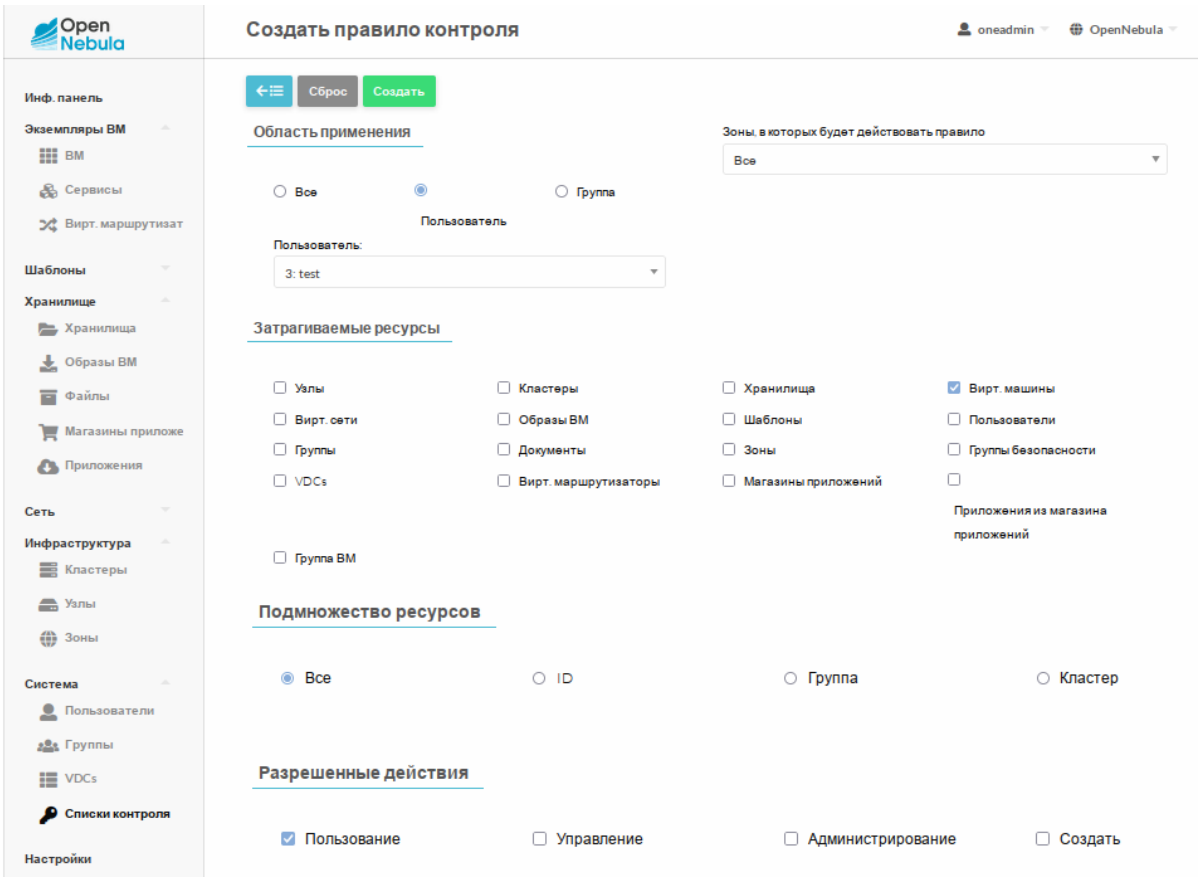


Рис. 45

Примечание. Каждое правило ACL добавляет новые разрешения и не может ограничивать существующие: если какое-либо правило даёт разрешение, операция разрешается.

3.7.5 Аутентификация пользователей

По умолчанию OpenNebula работает с внутренней системой аутентификации (пользователь/пароль). Но можно включить внешний драйвер аутентификации.

3.7.5.1 LDAP аутентификация

LDAP аутентификация позволяет пользователям и группам пользователей, принадлежащих практически любому аутентификатору на основе LDAP, получать доступ к виртуальным рабочим столам и приложениям.

Примечание. На сервере LDAP должна быть настроена отдельная учётная запись с правами чтения LDAP. От данной учетной записи будет выполняться подключение к серверу каталогов.

Для включения LDAP аутентификации необходимо в файл `/etc/one/oned.conf` добавить строку DEFAULT_AUTH = "ldap":

```
…
AUTH_MAD = [
    EXECUTABLE = "one_auth_mad",
    AUTHN = "ssh,x509,ldap,server_cipher,server_x509"
]

DEFAULT_AUTH = "ldap"
…
```

Примечание. В файле `/etc/one/sunstone-server.conf` для параметра `:auth` должно быть указано значение opennebula:

```
:auth: opennebula
```

Ниже приведён пример настройки аутентификации в Active Directory (домен test.alt).

Для подключения к Active Directory в файле `/etc/one/auth/ldap_auth.conf` необходимо указать:

- `:user` – пользователь AD с правами на чтение (пользователь указывается в формате opennebula@test.alt);
- `:password` – пароль пользователя;
- `:host` – имя или IP-адрес сервера AD;
- `:base` – базовый DN для поиска пользователя;
- `:user_field` – для этого параметра следует установить значение sAMAccountName;
- `:rfc2307bis` – для этого параметра следует установить значение true.

Пример файла /etc/one/auth/ldap_auth.conf:

```
 server 1:
    :user: 'opennebula@test.alt'
    :password: 'Pa$$word'
    :auth_method: :simple
    :host: dc2.test.alt
    :port: 389
    :base: 'dc=test,dc=alt'
    :user_field: 'sAMAccountName'
    :mapping_generate: false
    :mapping_timeout: 300
    :mapping_filename: server1.yaml
    :mapping_key: GROUP_DN
    :mapping_default: 100
    :rfc2307bis: true
:order:
    - server 1
```

Примечание. В параметре :order указывается порядок, в котором будут опрошены настроенные серверы. Элементы в параметре :order обрабатываются по порядку, пока пользователь не будет успешно аутентифицирован или не будет достигнут конец списка. Сервер, не указанный в параметре :order, не будет опрошен.

Примечание. Пример файла /etc/one/auth/ldap_auth.conf для настройки аутентификации в домене FreeIPA (домен example.test):

```
 server 1:
    :user: 'uid=admin,cn=users,cn=accounts,dc=example,dc=test'
    :password: '12345678'
    :auth_method: :simple
    :host: ipa.example.test
    :port: 389
    :base: 'dc=example,dc=test'
    :user_field: 'uid'
    :mapping_generate: false
    :mapping_timeout: 300
    :mapping_filename: server1.yaml
    :mapping_key: GROUP_DN
    :mapping_default: 100
    :rfc2307bis: true
:order:
    - server 1
```

После того как пользователь AD авторизуется в веб-интерфейсе OpenNebula, у администратора появится возможность изменять его настройки (Рис. 46).

Пользователи AD

Рис. 46

Новых пользователей можно автоматически включать в определенную группу/группы. Для этого создается сопоставление группы AD с существующей группой OpenNebula. Эта система использует файл сопоставления, указанный в параметре `:mapping_file` (файл должен находиться в каталоге `/var/lib/one/`).

Файл сопоставления может быть сгенерирован автоматически с использованием данных в шаблоне группы, который определяет, какая группа AD сопоставляется с этой конкретной группой (для параметра `:mapping_generate` должно быть установлено значение true). Если в шаблон группы добавить строку (Рис. 47):

```
GROUP_DN="CN=office,CN=Users,DC=test,DC=alt"
```

И в файле `/etc/one/auth/ldap_auth.conf` для параметра `:mapping_key` установить значение GROUP_DN, то поиск DN сопоставляемой группы будет осуществляться в этом параметре шаблона. В этом случае файл `/var/lib/one/server1.yaml` будет сгенерирован со следующим содержимым:

```
---
CN=office,CN=Users,DC=test,DC=alt: '100'
```

и пользователи из группы AD office, будут сопоставлены с группой ALT (ID=100).

Шаблон группы

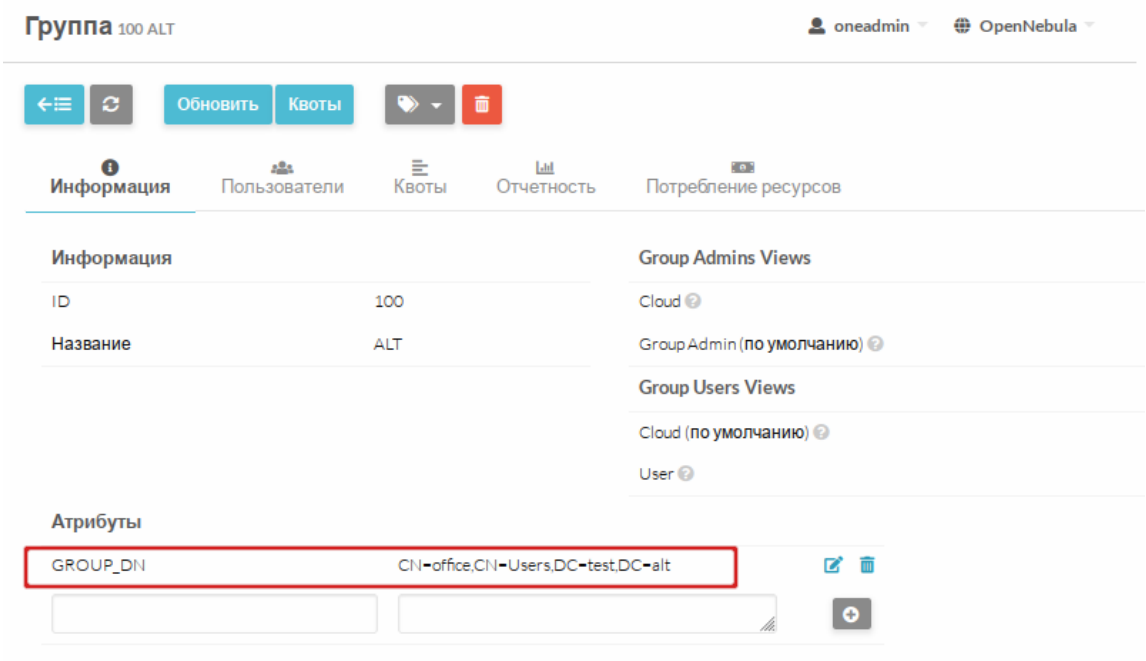


Рис. 47

Можно отключить автоматическую генерацию файла сопоставления, установив ние `:mapping_generate` равным false, и выполнить сопоставление вручную. Файл сопоставления имеет формат YAML и содержит хеш, где ключ – это DN группы AD, а значение – идентификатор группы OpenNebula. Например, если содержимое файла `/var/lib/one/server1.yaml`:

```
CN=office,CN=Users,DC=test,DC=alt: '100'
CN=Domain Admins,CN=Users,DC=test,DC=alt: '101'
```

то пользователи из группы AD office, будут сопоставлены с группой ALT (ID=100), а из группы AD Domain Admin – с группой Admin (ID=101) (Рис. 48).

Сопоставление пользователей AD

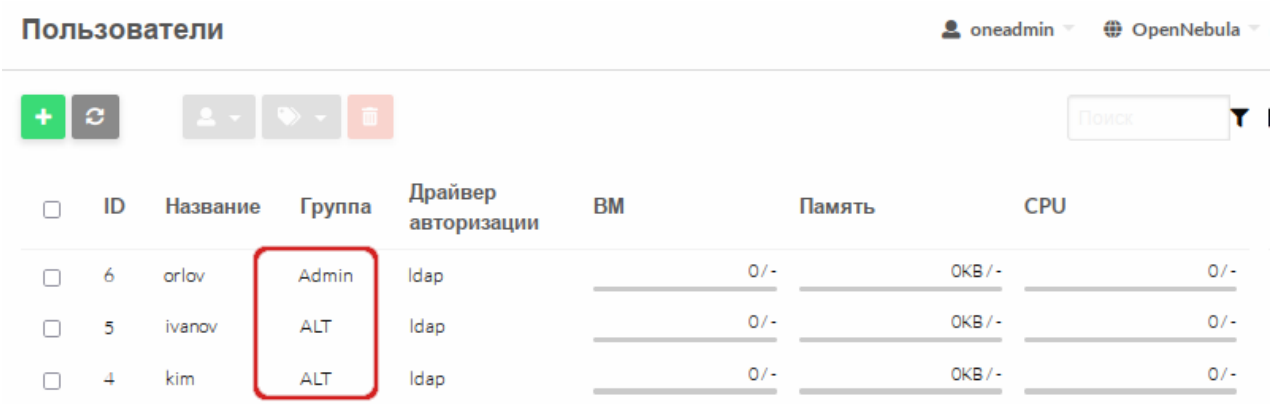


Рис. 48

3.8 Настройка отказоустойчивого кластера

В данном разделе рассмотрена настройка отказоустойчивого кластера (High Available, HA) для основных служб OpenNebula: core (oned), scheduler (mm_sched).

OpenNebula использует распределенный консенсусный протокол Raft, для обеспечения отказоустойчивости и согласованности состояний между службами. Алгоритм консенсуса построен на основе двух концепций:

- состояние системы – данные, хранящиеся в таблицах базы данных (пользователи, списки управления доступом или виртуальные машины в системе);
- журнал (log) – последовательность операторов SQL, которые последовательно применяются к базе данных OpenNebula на всех серверах для изменения состояния системы.

Чтобы сохранить согласованное представление о системе на всех серверах, изменения состояния системы выполняются через специальный узел, лидер или ведущий (Leader). Leader периодически посылает запросы (heartbeats) другим серверам, ведомым (Follower), чтобы сохранить свое лидерство. Если Leader не может послать запрос, Follower-серверы продвигаются к кандидатам и начинают новые выборы.

Каждый раз, когда система изменяется (например, в систему добавляется новая ВМ), Leader обновляет журнал и реплицирует запись у большинства Follower, прежде чем записывать её в базу данных. Таким образом, увеличивается задержка операций с БД, но состояние системы безопасно реплицируется, и кластер может продолжить свою работу в случае отказа узла.

Для настройки High Available требуется:

- нечетное количество серверов (рекомендуемый размер развертывания – 3 или 5 серверов, что обеспечивает отказоустойчивость при отказе 1 или 2 серверов соответственно);
- рекомендуется идентичная конфигурация серверов;
- идентичная программная конфигурация серверов (единственное отличие – это поле SERVER_ID в `/etc/one/oned.conf`);
- рекомендуется использовать подключение к базе данных одного типа (MySQL);
- серверы должны иметь беспарольный доступ для связи друг с другом;
- плавающий IP, который будет газначен лидеру;
- общая файловая система.

Добавлять дополнительные серверы или удалять старые можно после запуска кластера.

В данном примере показана настройка HA кластера из трех серверов:

- 192.168.0.186 opennebula
- 192.168.0.187 server02
- 192.168.0.188 server03

– 192.168.0.200 Floating IP

3.8.1 Первоначальная конфигурация Leader

Запустить сервис OpenNebula и добавить локальный сервер в существующую или новую зону (в примере зона с ID 0):

```
$ onezone list
C   ID NAME             ENDPOINT                                        FED_INDEX
*    0 OpenNebula       http://localhost:2633/RPC2                      -1

$ onezone server-add 0 --name opennebula --rpc http://192.168.0.186:2633/RPC2

$ onezone show 0
ZONE 0 INFORMATION
ID                : 0
NAME              : OpenNebula



ZONE SERVERS
ID NAME            ENDPOINT
 0 opennebula      http://192.168.0.186:2633/RPC2

HA & FEDERATION SYNC STATUS
ID NAME            STATE      TERM       INDEX      COMMIT     VOTE  FED_INDEX
 0 opennebula      solo       0          -1         0          -1    -1

ZONE TEMPLATE
ENDPOINT="http://localhost:2633/RPC2"
```

Остановить сервис opennebula и обновить конфигурацию SERVER_ID в файле /etc/one/oned.conf:

```
FEDERATION = [
    MODE          = "STANDALONE",
    ZONE_ID       = 0,
    SERVER_ID     = 0, # изменить с -1 на 0 (0 — это ID сервера)
    MASTER_ONED   = ""
]
```

Включить Raft-обработчики, чтобы добавить плавающий IP-адрес в кластер (файл /etc/one/oned.conf):

```
RAFT_LEADER_HOOK = [
    COMMAND = "raft/vip.sh",
    ARGUMENTS = "leader enp0s3 192.168.0.200/24"
]
```

```
# Executed when a server transits from leader->follower
RAFT_FOLLOWER_HOOK = [
    COMMAND = "raft/vip.sh",
    ARGUMENTS = "follower enp0s3 192.168.0.200/24"
]
```

Запустить сервис OpenNebula и проверить зону:

```
$ onezone show 0
ZONE 0 INFORMATION
ID                : 0
NAME              : OpenNebula


ZONE SERVERS
ID NAME            ENDPOINT
 0 opennebula      http://192.168.0.186:2633/RPC2

HA & FEDERATION SYNC STATUS
ID NAME            STATE      TERM       INDEX      COMMIT     VOTE  FED_INDEX
 0 opennebula      leader     1          5          5          0     -1

ZONE TEMPLATE
ENDPOINT="http://localhost:2633/RPC
```

Сервер opennebula стал Leader-сервером, так же ему присвоен плавающий адрес (Floating IP):

```
$ ip -o a sh enp0s3
2:  enp0s3        inet  192.168.0.186/24  brd  192.168.0.255  scope  global  enp0s3\
valid_lft forever preferred_lft forever
2: enp0s3     inet 192.168.0.200/24 scope global secondary enp0s3\        valid_lft
forever preferred_lft forever
2: enp0s3    inet6 fe80::a00:27ff:fec7:38e6/64 scope link \        valid_lft forever
preferred_lft forever
```

3.8.2 Добавление дополнительных серверов

Примечание. Данная процедура удалит полностью базу на сервере и заменит её актуальной с Leader-сервера.

Примечание. Рекомендуется добавлять по одному хосту за раз, чтобы избежать конфликта в базе данных.

На Leader создать полную резервную копию актуальной базы и перенести её на другие серверы вместе с файлами из каталога `/var/lib/one/.one/`:

```
$ onedb backup -u oneadmin -d opennebula -p oneadmin
```

```
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula_2021-6-23_13:43:21.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

$ scp /var/lib/one/mysql_localhost_opennebula_2021-6-23_13\:43\:21.sql <ip>:/tmp

$ ssh <ip> rm -rf /var/lib/one/.one
$ scp -r /var/lib/one/.one/ <ip>:/var/lib/one/
```

Остановить сервис OpenNebula на Follower-хостах и восстановить скопированную базу:

```
$ onedb restore -f -u oneadmin -p oneadmin -d opennebula
/tmp/mysql_localhost_opennebula_2021-6-23_13\:43\:21.sql
MySQL DB opennebula at localhost restored.
```

Перейти на Leader-сервер и добавить в зону новые хосты (рекомендуется добавлять серверы по-одному):

```
$ onezone server-add 0 --name server02 --rpc http://192.168.0.187:2633/RPC2
```

Проверить зону на Leader-сервере:

```
$ onezone show 0
ZONE 0 INFORMATION
ID                : 0
NAME              : OpenNebula


ZONE SERVERS
ID NAME            ENDPOINT
 0 opennebula      http://192.168.0.186:2633/RPC2
 1 server02        http://192.168.0.187:2633/RPC2

HA & FEDERATION SYNC STATUS
ID NAME            STATE      TERM       INDEX      COMMIT     VOTE  FED_INDEX
 0 opennebula      leader     4          22         22         0     -1
 1 server02        error      -          -          -          -     -

ZONE TEMPLATE
ENDPOINT="http://localhost:2633/RPC2"
```

Новый сервер находится в состоянии ошибки, так как OpenNebula на новом сервере не запущена. Следует запомнить идентификатор сервера, в данном случае он равен 1.

Переключиться на добавленный Follower-сервер и обновить конфигурацию SERVER_ID в файле `/etc/one/oned.conf`, (указать в качестве SERVER_ID значение из предыдущего шага). Включить Raft-обработчики, как это было выполнено на Leader.

Запустить сервис OpenNebula на Follower-сервере и проверить на Leader-сервере состояние зоны:

```
$ onezone show 0
ZONE 0 INFORMATION
ID                : 0
NAME              : OpenNebula


ZONE SERVERS
ID NAME            ENDPOINT
 0 opennebula      http://192.168.0.186:2633/RPC2
 1 server02        http://192.168.0.187:2633/RPC2

HA & FEDERATION SYNC STATUS
ID NAME            STATE      TERM       INDEX      COMMIT     VOTE  FED_INDEX
 0 opennebula      leader     4          28         28         0     -1
 1 server02        follower   4          28         28         0     -1

ZONE TEMPLATE
ENDPOINT="http://localhost:2633/RPC2""
```

Повторить данные действия, чтобы добавить третий сервер в кластер.

Примечание. Добавлять серверы в кластер, следует только в случае нормальной работы кластера (работает Leader, а остальные находятся в состоянии Follower). Если в состоянии Error присутствует хотя бы один сервер, необходимо это исправить.

Проверка работоспособности кластера:

```
$ onezone show 0
ZONE 0 INFORMATION
ID                : 0
NAME              : OpenNebula


ZONE SERVERS
ID NAME            ENDPOINT
 0 opennebula      http://192.168.0.186:2633/RPC2
 1 server02        http://192.168.0.187:2633/RPC2
 2 server03        http://192.168.0.188:2633/RPC2

HA & FEDERATION SYNC STATUS
ID NAME            STATE      TERM       INDEX      COMMIT     VOTE  FED_INDEX
 0 opennebula      leader     4          35         35         0     -1
 1 server02        follower   4          35         35         0     -1
```

```
 2 server03        follower   4          35         35         0      -1

ZONE TEMPLATE
ENDPOINT="http://localhost:2633/RPC2"
```

Если какой-либо хост находится в состоянии ошибки, следует проверить журнал (`/var/log/one/oned.log`), как в текущем Leader (если он есть), так и в хосте, который находится в состоянии Error. Все сообщения Raft будут регистрироваться в этом файле.

3.8.3 Добавление и удаление серверов

Команда добавления сервера:

```
$ onezone server-add <zoneid>
```

Параметры:

- -n, --name – имя сервера зоны;
- -r, --rpc – конечная точка RPC сервера зоны;
- -v, --verbose – подробный режим;
- --user name – имя пользователя, используемое для подключения к OpenNebula;
- --password password – пароль для аутентификации в OpenNebula;
- --endpoint endpoint – URL конечной точки интерфейса OpenNebula xmlrpc.

Команда удаления сервера:

```
$ onezone server-del <zoneid> <serverid>
```

Параметры:

- -v, --verbose – подробный режим;
- --user name – имя пользователя, используемое для подключения к OpenNebula;
- --password password – пароль для аутентификации в OpenNebula;
- --endpoint endpoint – URL конечной точки интерфейса OpenNebula xmlrpc.

3.8.4 Восстановление сервера

Если Follower -сервер в течение некоторого времени не работает, он может выпасть из окна восстановления. Чтобы восстановить этот сервер необходимо:

- Leader: создать резервную копию БД и скопировать её на отказавший сервер (повторно использовать предыдущую резервную копию нельзя).
- Follower: остановить OpenNebula.
- Follower: восстановить резервную копию БД.
- Follower: запустить OpenNebula.
- Leader: сбросить отказавший Follower, выполнив команду:

```
$ onezone server-reset <zone_id> <server_id_of_failed_follower>
```

3.8.5 Sunstone

Есть несколько способов развертывания Sunstone в среде HA. Базовым является Sunstone, работающий на каждом интерфейсном узле OpenNebula. Клиенты используют только один сервер – Leader с плавающим IP.

4 СРЕДСТВО УПРАВЛЕНИЯ ВИРТУАЛЬНЫМИ ОКРУЖЕНИЯМИ PVE

4.1 Краткое описание возможностей

Proxmox Virtual Environment (PVE) – средство управления виртуальными окружениями на базе гипервизора KVM и системы контейнерной изоляции LXC. Основными компонентами среды являются:

- физические серверы, на которых выполняются процессы гипервизора KVM, и процессы, работающие в контейнерах LXC;
- хранилища данных, в которых хранятся образы установочных дисков, образы дисков виртуальных машин KVM и файлы, доступные из контейнеров LXC;
- виртуальные сетевые коммутаторы, к которым подключаются сетевые интерфейсы виртуальных окружений.

PVE состоит из веб-интерфейса, распределенного хранилища данных конфигурации виртуальных окружений, а также утилит конфигурирования, работающих в командной строке. Все эти инструменты предназначены только для управления средой выполнения виртуальных окружений. За формирование среды отвечают компоненты системы, не входящие непосредственно в состав PVE. В первую очередь это сетевая и дисковая подсистемы, а также механизмы аутентификации.

4.1.1 Системные требования

Минимальные (для тестирования):

- CPU: 64bit (Intel EMT64 или AMD64), поддержка Intel VT/AMD-V CPU/Mainboard;
- минимум 1 Гб ОЗУ;
- жесткий диск;
- сетевая карта.

Рекомендуемые:

- CPU: мультипроцессорный 64bit (Intel EMT64 или AMD64), поддержка Intel VT/AMD-V CPU/Mainboard;
- мимнимум 2 Гб ОЗУ для ОС и сервисов PVE. Плюс выделенная память для гостевых систем. Для Ceph или ZFS требуется дополнительная память, примерно 1 ГБ ОЗУ на каждый ТБ используемого хранилища;
- хранилище для ОС: аппаратный RAID;

- хранилище для ВМ: аппаратный RAID для локального хранилища, или non-RAID для ZFS. Также возможно совместное и распределенное хранение;
- быстрые жёсткие диски 15krpm SAS, Raid10;
- сетевая карта.

Реальные системные требования определяются количеством и требованиями гостевых систем.

4.1.2 Веб-интерфейс

Веб-интерфейс PVE предназначен для решения следующих задач:

- создание, удаление, настройка виртуальных окружений;
- управление физическими серверами;
- мониторинг активности виртуальных окружений и использования ресурсов среды;
- фиксация состояний (snapshot-ы), создание резервных копий и шаблонов виртуальных окружений, восстановление виртуальных окружений из резервных копий.

Кроме решения пользовательских задач, веб-интерфейс PVE можно использовать еще и для встраивания в интегрированные системы управления – например, в панели управления хостингом. Для этого он имеет развитый RESTful API с JSON в качестве основного формата данных.

Для аутентификации пользователей веб-интерфейса можно использовать как собственные механизмы PVE, так и PAM. Использование PAM дает возможность, например, интегрировать PVE в домен аутентификации.

Так как используется кластерная файловая система (pmxcfs), можно подключиться к любому узлу для управления всем кластером. Каждый узел может управлять всем кластером.

Веб-интерфейс PVE доступен по адресу https://<имя-компьютера>:8006. Потребуется пройти аутентификацию (логин по умолчанию: root, пароль указывается в процессе установки ОС) (Рис. 49).

Пользовательский интерфейс PVE (Рис. 50) состоит из четырех областей:

- заголовок – верхняя часть. Показывает информацию о состоянии и содержит кнопки для наиболее важных действий;
- дерево ресурсов –левая сторона. Дерево навигации, где можно выбирать конкретные объекты;
- панель контента – центральная часть. Здесь отображаются конфигурация и статус выбранных объектов;
- панель журнала – нижняя часть. Отображает записи журнала для последних задач. Чтобы получить более подробную информацию или прервать выполнение задачи, следует дважды щелкнуть левой клавишей мыши по записи журнала.

Аутентификация в веб-интерфейсе PVE

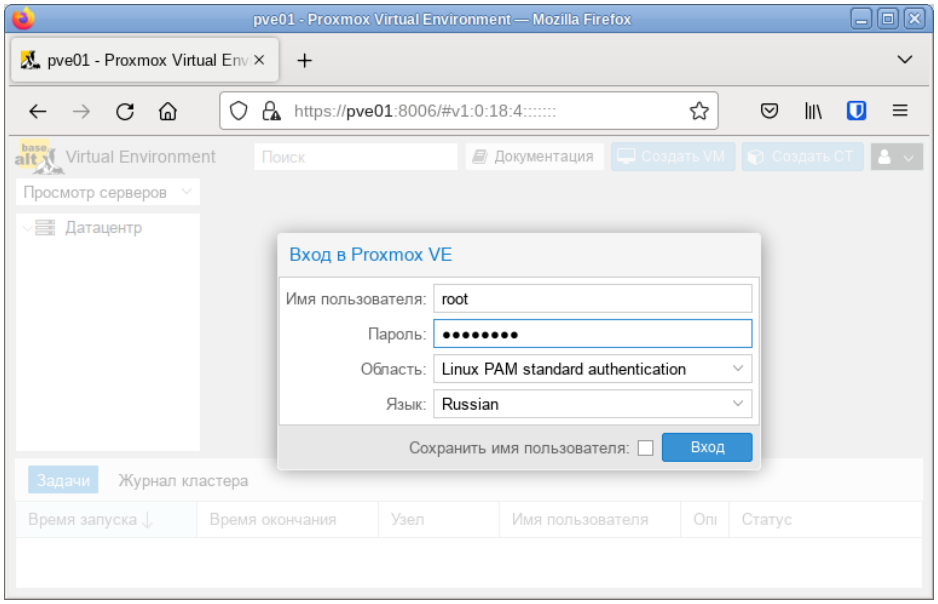


Рис. 49

Веб-интерфейс PVE

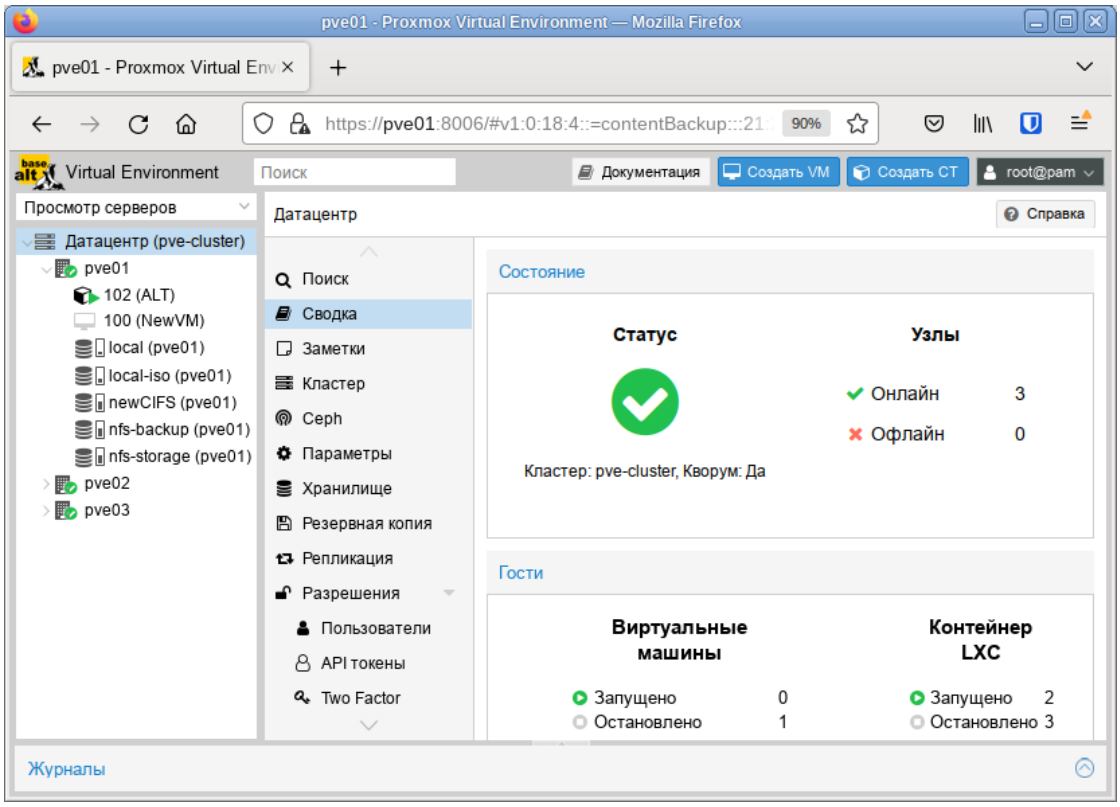


Рис. 50

Примечание. Есть возможность работы с PVE из мобильного приложения, например, Proxmox. В приложении можно получить доступ к узлам (Рис. 51), ВМ и контейнерам. Можно зайти в консоль ВМ с помощью noVNC или SPICE, осуществлять необходимые манипуляции внутри ВМ (Рис. 52).

Работа с PVE из мобильного приложения

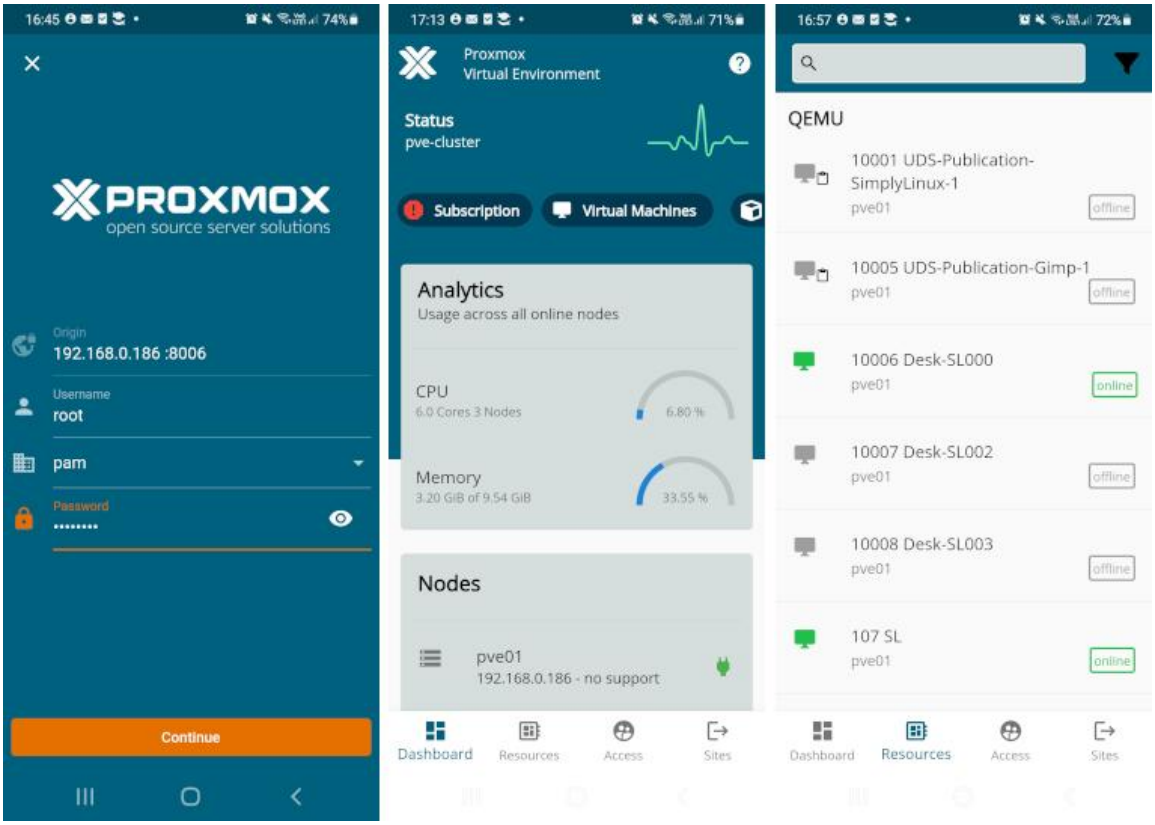


Рис. 51

Работа с ВМ из мобильного приложения

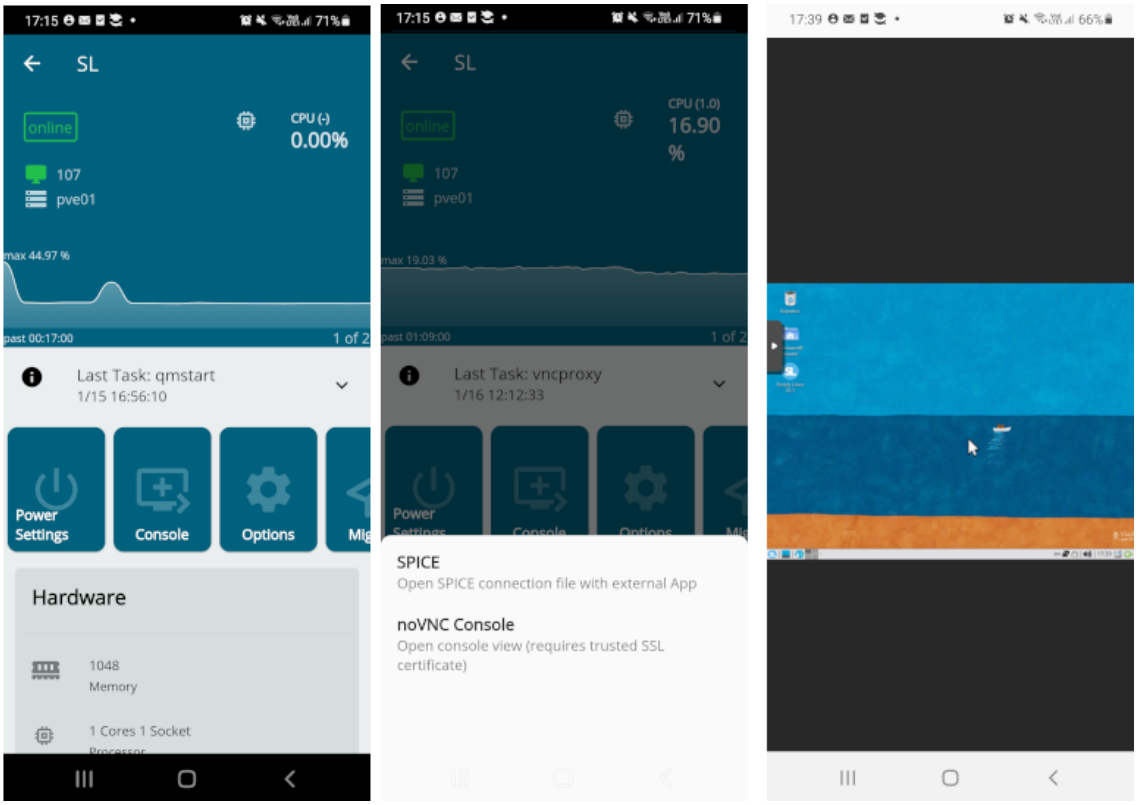


Рис. 52

4.1.3 Хранилище данных

В случае локальной установки PVE для размещения данных виртуальных окружений можно дополнительно ничего не настраивать и использовать локальную файловую систему сервера. Но в случае кластера из нескольких серверов имеет смысл настроить сетевую или распределенную сетевую файловую систему, обеспечивающую параллельный доступ к файлам со всех серверов, на которых выполняются процессы виртуальных окружений. В качестве таких файловых систем могут выступать, например, NFS или CEPH.

4.1.4 Сетевая подсистема

В отличие от хранилища данных, сетевая подсистема требует специальной настройки даже в случае локальной установки PVE. Это обусловлено тем, что сетевые интерфейсы виртуальных окружений должны подключаться к какому-то виртуальному устройству, обеспечивающему соединение с общей сетью передачи данных. Перед началом настройки сети следует принять решение о том, какой тип виртуализации (LXC/KVM) и какой тип подключения будет использоваться (маршрутизация/мост).

4.2 Установка и настройка PVE

Примечание. Компоненты PVE будут установлены в систему, если при установке дистрибутива выбрать профиль «Виртуальное окружение Proxmox». Также при установке дистрибутива, необходимо настроить Ethernet-мост vmbr0 и при заполнении поля с именем компьютера указать полное имя с доменом.

Все остальные настройки можно делать в веб-интерфейсе см. «Создание кластера PVE».

4.2.1 Установка PVE

Если развертывание PVE происходит в уже установленной системе на базе Десятой платформы, достаточно любым штатным способом (apt-get, aptitude, synaptic) установить пакет `pve-manager` (все необходимые компоненты при этом будут установлены автоматически):

```
# apt-get update
# apt-get install pve-manager
```

Также следует убедиться в том, что пакет systemd обновлен до версии, находящейся в репозитории P10.

4.2.2 Настройка сетевой подсистемы

На всех узлах кластера необходимо настроить Ethernet-мост.

Примечание. Мосту должно быть назначено имя vmbr0 и оно должно быть одинаково на всех узлах.

Примечание. При использовании дистрибутива Альт Сервер Виртуализации интерфейс vmbr0 создаётся и настраивается в процессе установки системы.

4.2.2.1 Настройка Ethernet-моста в командной строке

Перед настройкой Ethernet-моста (далее – моста) с помощью etcnet сначала необходимо убедиться, что установлен пакет `bridge-utils`. Etcnet использует утилиту `brctl` для настройки моста, и, если утилита не установлена, то при перезапуске системы сеть станет недоступна. Если интерфейсы, входящие в состав моста, являются единственными физически подключенными и настройка моста происходит с удаленного узла через эти интерфейсы, то требуется соблюдать осторожность, т.к. эти интерфейсы перестанут быть доступны. В случае ошибки в конфигурации потребуется физический доступ к серверу. Для страховки, перед перезапуском сервиса network можно открыть еще одну консоль и запустить там, например, команду: `sleep 500 && reboot`.

Для настройки Ethernet-моста с именем vmbr0, следует выполнить следующие команды:

```
# mkdir /etc/net/ifaces/vmbr0
# cp /etc/net/ifaces/enp0s3/* /etc/net/ifaces/vmbr0/
# rm -f /etc/net/ifaces/enp0s3/{i,r}*
# cat <<EOF > /etc/net/ifaces/vmbr0/options
BOOTPROTO=static
CONFIG_WIRELESS=no
CONFIG_IPV4=yes
HOST='enp0s3'
ONBOOT=yes
TYPE=bri
EOF
```

Имя интерфейса, обозначенного здесь как `enp0s3`, следует указать в соответствии с реальной конфигурацией сервера.

IP-адрес для интерфейса будет взят из `ipv4address`.

В опции HOST файла `options` нужно указать те интерфейсы, которые будут входить в мост. Если в него будут входить интерфейсы, которые до этого имели IP-адрес (например, `enp0s3`), то этот адрес должен быть удален (например, можно закомментировать содержимое файла `/etc/net/ifaces/enp0s3/ipv4address`).

Для того, чтобы изменения вступили в силу необходим перезапуск сервиса `network`:

```
# systemctl restart network
```

При старте сети сначала поднимаются интерфейсы, входящие в мост, затем сам мост (автоматически).

4.2.2.2 *Настройка Ethernet-моста в веб-интерфейсе*

При установленных пакетах alterator-net-eth и alterator-net-bridge, для настройки Ethernet-моста можно воспользоваться веб-интерфейсом центра управления системой.

Примечание. Должен также быть установлен пакет alterator-fbi и запущены сервисы ahttpd и alteratord:

```
# apt-get install alterator-fbi
# systemctl start ahttpd
# systemctl start alteratord
```

Веб-интерфейс доступен по адресу https://ip-address:8080.

Для настройки Ethernet-моста необходимо выполнить следующие действия:

1) в группе «Сеть» выбрать пункт «Ethernet-интерфейсы»;
2) удалить IP-адрес и шлюз по умолчанию (Рис. 53) и нажать кнопку «Создать сетевой мост»;
3) в открывшемся окне (Рис. 54), задать имя моста vmbr0, выбрать сетевой интерфейс в списке доступных интерфейсов («Available interfaces»), переместить его в список «Участники» («Members») и нажать кнопку «Ok»;
4) настроить сетевой интерфейс vmbr0: ввести имя компьютера, задать IP-адрес и нажать кнопку «Добавить», ввести адрес шлюза по умолчанию и DNS-сервера (Рис. 55).

Настройка сети в веб-интерфейсе

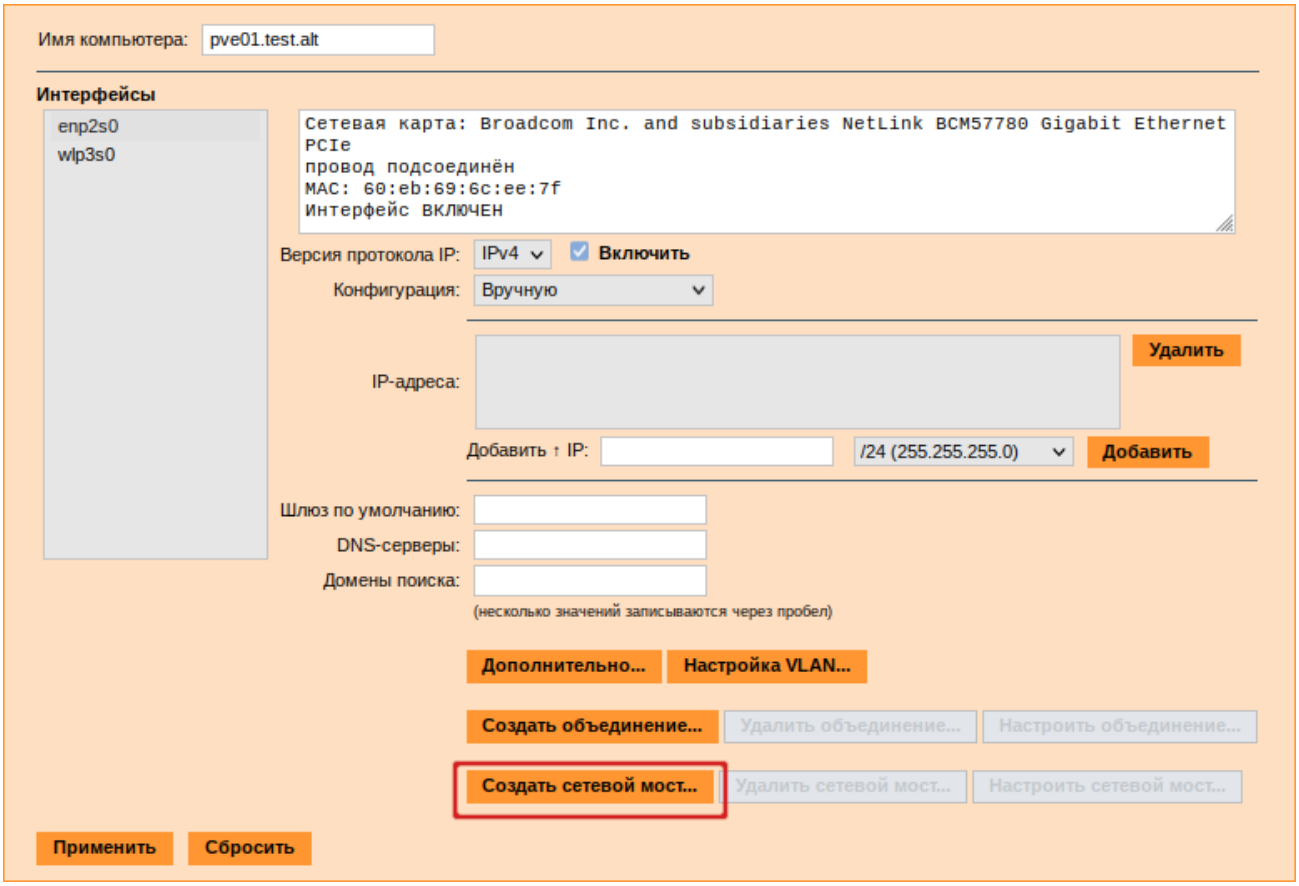


Рис. 53

Выбор сетевого интерфейса

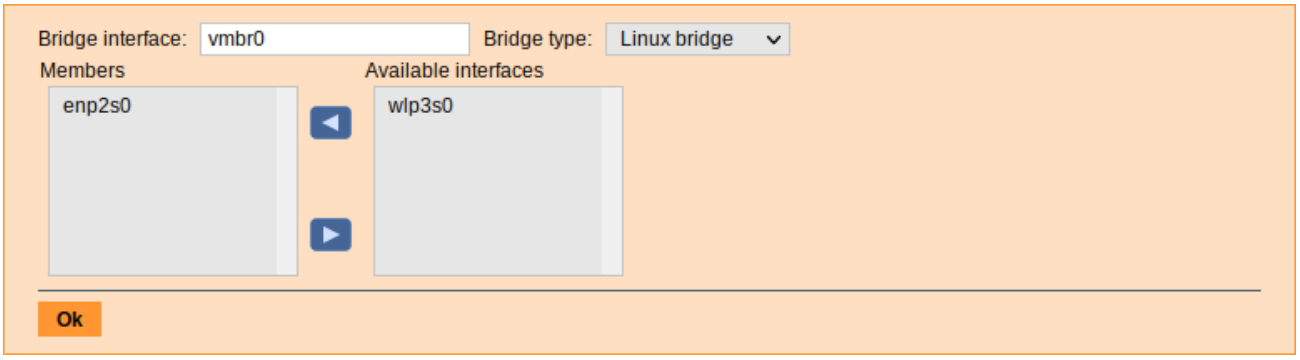


Рис. 54

Настройка параметров сетевого интерфейса vmbr0

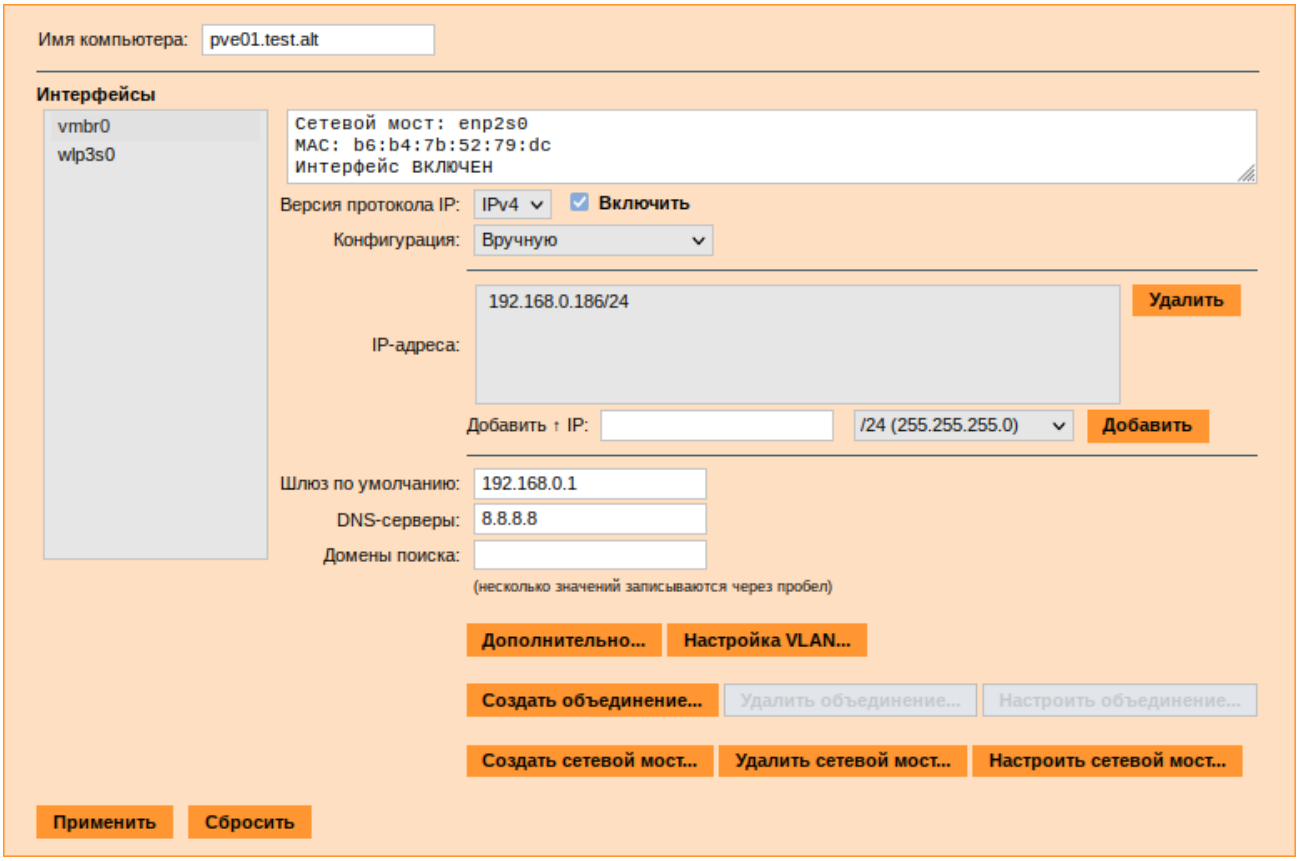


Рис. 55

4.3 Создание кластера PVE

Рекомендации:

- все узлы должны иметь возможность подключаться друг к другу через UDP порты 5404 и 5405;
- дата и время должны быть синхронизированы;
- между узлами используется SSH туннель на 22 TCP порту;

- если необходимо обеспечение высокой доступности (High Availability), необходимо иметь как минимум три узла для организации кворума. На всех узлах должна быть установлена одна версия PVE;
- рекомендуется использовать выделенный сетевой адаптер для трафика кластера, особенно если используется общее хранилище.

PVE кластер может состоять из двух и более серверов.

Кластер не создается автоматически на только что установленном узле PVE. В настоящее время создание кластера может быть выполнено либо в консоли (вход через ssh), либо в веб-интерфейсе («Датацентр» → «Кластер»).

Примечание. PVE при создании кластера включает парольную аутентификацию для root в sshd. В целях повышения безопасности, после включения всех серверов в кластер, рекомендуется отключить парольную аутентификацию для root в sshd:

```
# control sshd-permit-root-login without_password
# systemctl restart sshd
```

При добавлении в кластер нового сервера, можно временно включить парольную аутентификацию:

```
# control sshd-permit-root-login enabled
# systemctl restart sshd
```

А после того как сервер будет добавлен, снова отключить.

Кластеры используют ряд определенных портов (Табл. 2) для различных функций. Важно обеспечить доступность этих портов и отсутствие их блокировки межсетевыми экранами.

Таблица 2 – Используемые порты

Порт	Функция
TCP 8006	Веб-интерфейс PVE
TCP 5900-5999	Доступ к консоли VNC
TCP 3128	Доступ к консоли SPICE
TCP 22	SSH доступ
UDP 5404, 5405	Широковещательный CMAN для применения настроек кластера

4.3.1 Настройка узлов кластера

PVE должен быть установлен на всех узлах. Следует убедиться, что каждый узел установлен с окончательным именем хоста и IP-конфигурацией. Изменение имени хоста и IP-адреса невозможно после создания кластера.

Необходимо обеспечить взаимно однозначное прямое и обратное преобразование имен для всех узлов кластера. Крайне желательно использовать DNS, в крайнем случае, можно обойтись соответствующими записями в локальных файлах `/etc/hosts`:

```
# echo "192.168.0.186 pve01.test.alt pve01" >> /etc/hosts
# echo "192.168.0.90 pve02.test.alt pve02" >> /etc/hosts
# echo "192.168.0.70 pve03.test.alt pve03" >> /etc/hosts
```

Примечание. В PVE это можно сделать в панели управления: выбрать узел, перейти в «Система» → «Hosts», добавить все узлы, которые будут включены в состав кластера (Рис. 56).

Редактирование записей в файле /etc/hosts

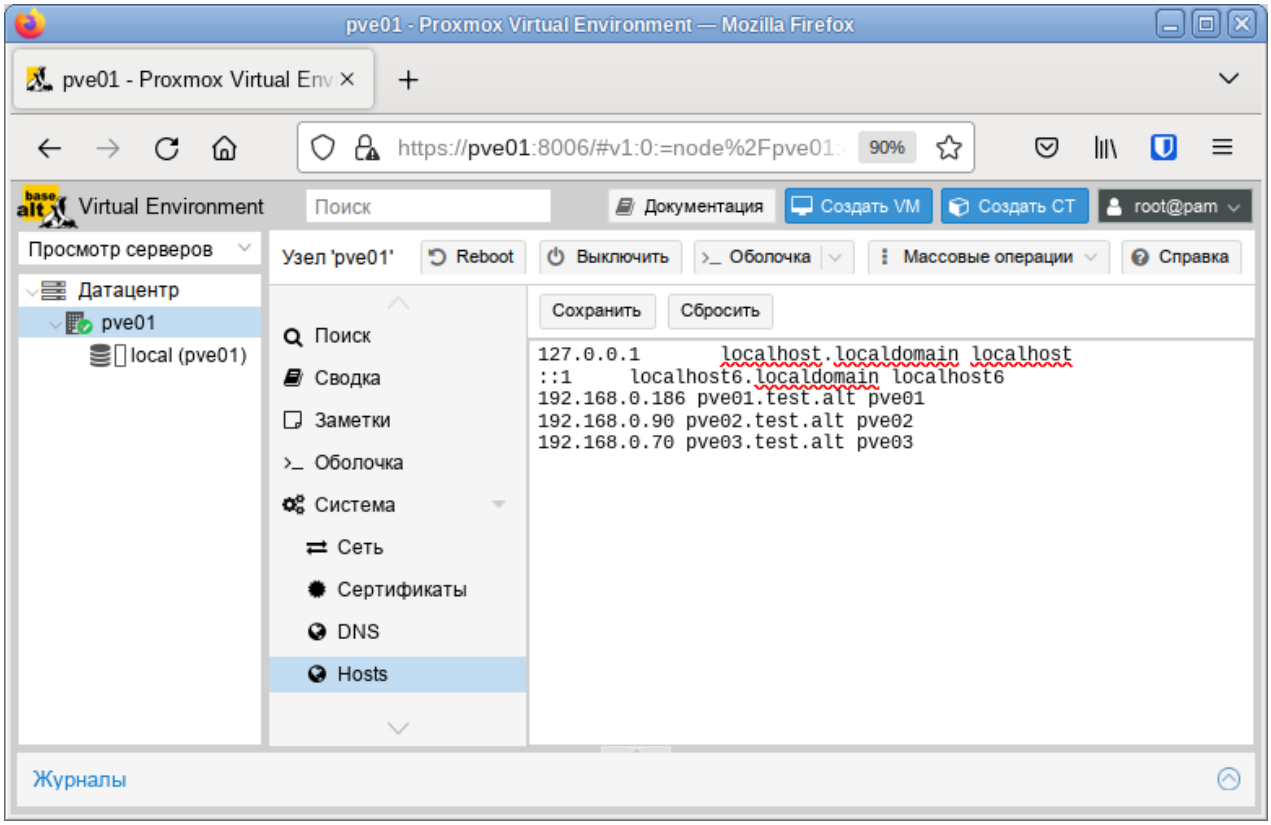


Рис. 56

Примечание. Имя машины не должно присутствовать в файле `/etc/hosts` разрешающимся в 127.0.0.1.

4.3.2 Создание кластера в веб-интерфейсе

Для создания кластера необходимо выполнить следующие действия:

1) в панели управления любого узла кластера выбрать «Датацентр» → «Кластер» и нажать кнопку «Создать кластер» (Рис. 57);
2) в открывшемся окне, задать название кластера, выбрать IP-адрес интерфейса, на котором узел кластера будет работать, и нажать кнопку «Создать» (Рис. 58);
3) при успешном создании кластера будет выведена соответствующая информация (Рис. 59).

Создание кластера в веб-интерфейсе

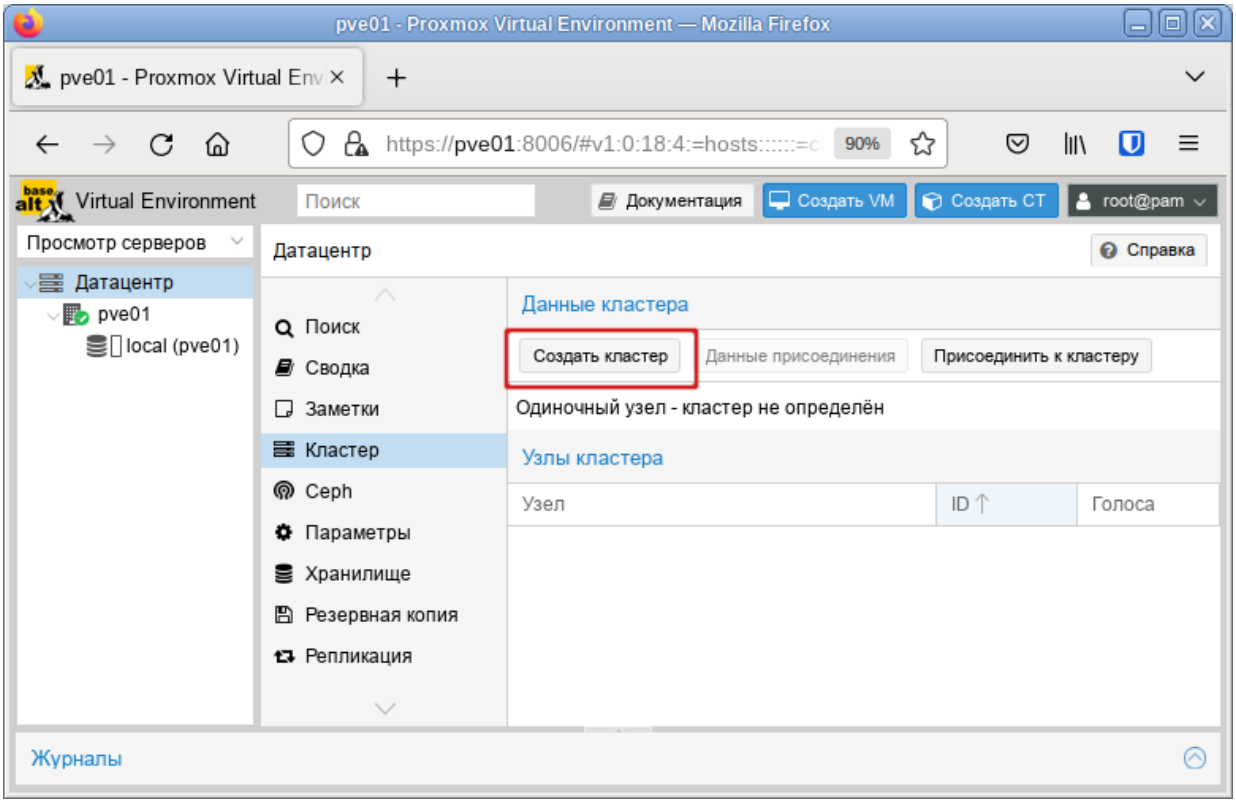


Рис. 57

Создание кластера в веб-интерфейсе. Название кластера

Рис. 58

Информация о создании кластера

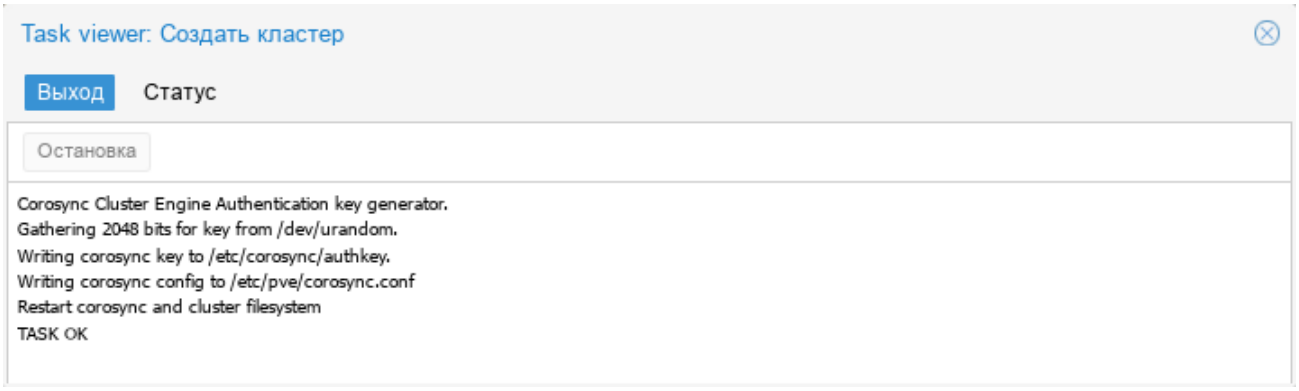


Рис. 59

Для добавления узла в кластер необходимо выполнить следующие действия:

1) в панели управления узла, на котором был создан кластер, выбрать «Датацентр» → «Кластер» и нажать кнопку «Данные присоединения» (Рис. 60);
2) в открывшемся окне, нажав кнопку «Копировать данные» (Рис. 61), скопировать данные присоединения;
3) перейти в панель управления узла, который следует присоединить к кластеру. Выбрать пункт«Датацентр» → «Кластер» и нажать кнопку «Присоединить к кластеру» (Рис. 62);
4) в поле «Данные» вставить данные присоединения, поля «Адрес сервера» и «Отпечаток» при этом будут заполнены автоматически. В поле «Пароль» ввести пароль пользователя root первого узла (Рис. 63) и нажать кнопку «Join <имя кластера>» («Присоединение»);
5) через несколько минут, после завершения репликации всех настроек, узел будет подключен к кластеру (Рис. 64).

Создание кластера в веб-интерфейсе. Получить данные присоединения

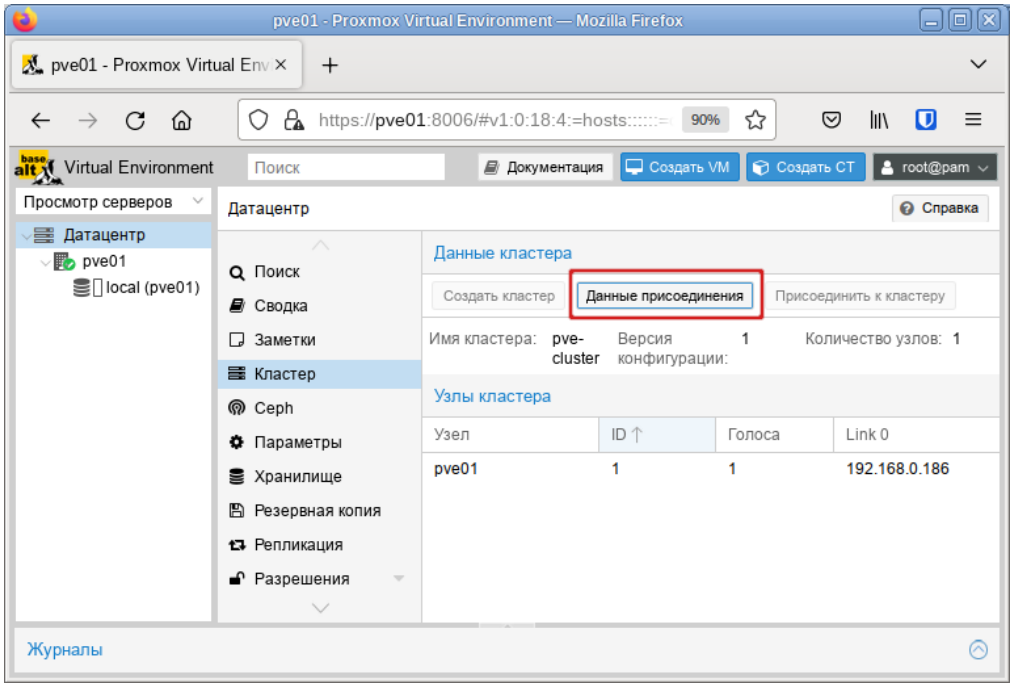


Рис. 60

Создание кластера в веб-интерфейсе. Данные присоединения

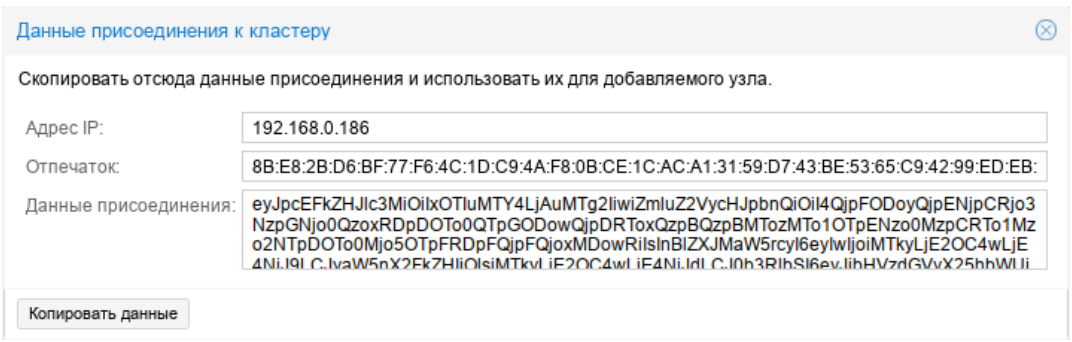


Рис. 61

Узел, который следует присоединить к кластеру

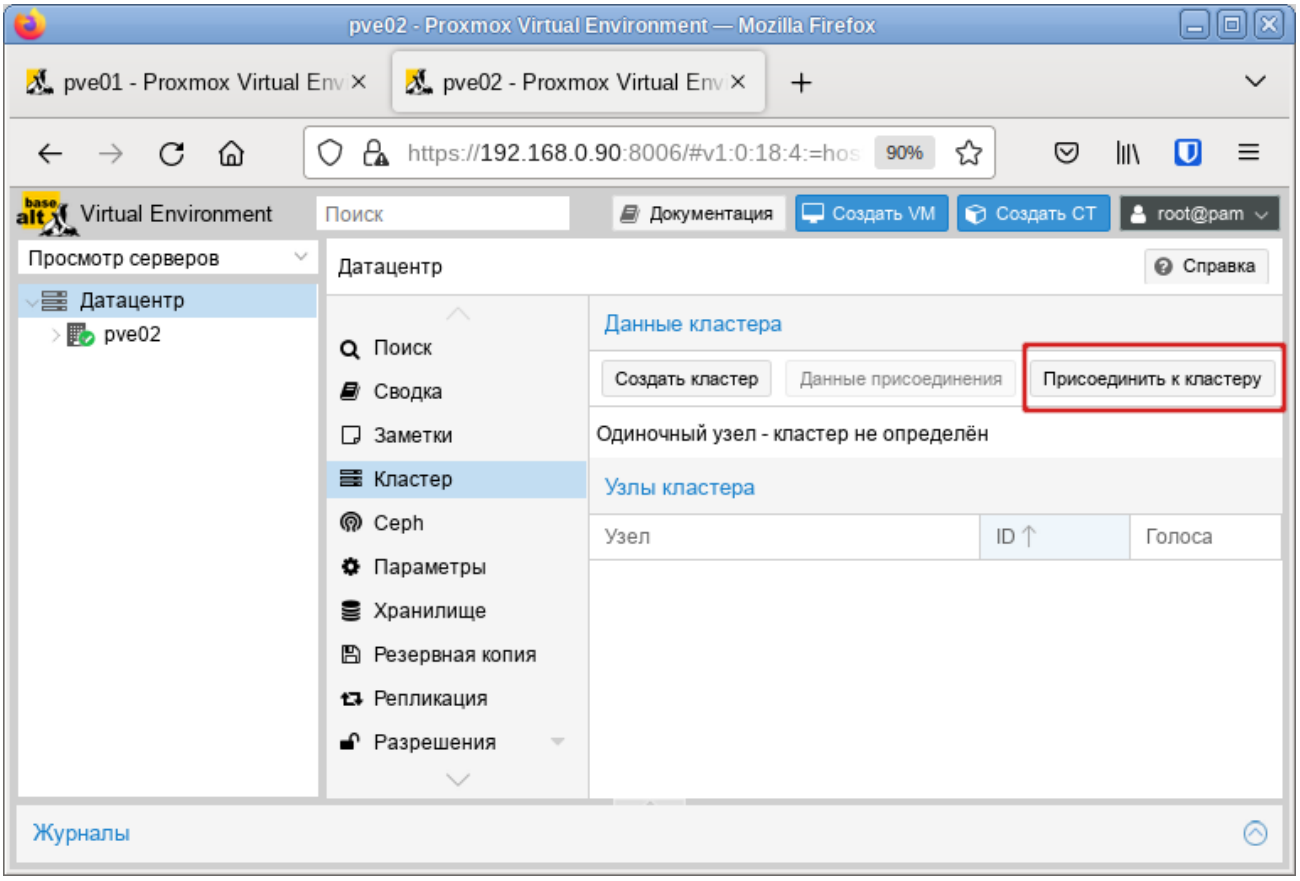


Рис. 62

Присоединение узла к кластеру

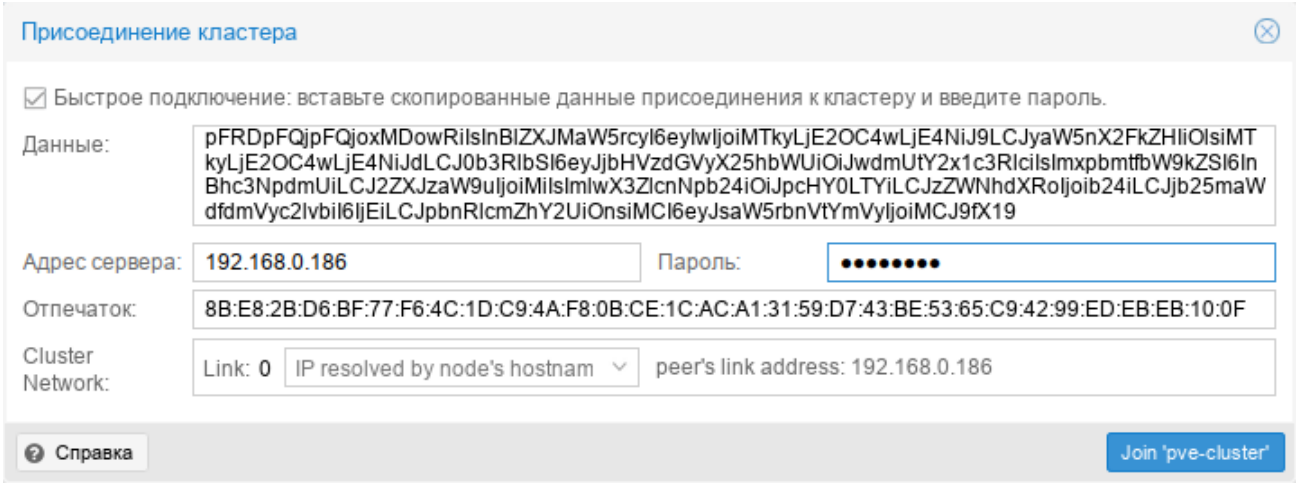


Рис. 63

Узлы кластера в веб-интерфейсе

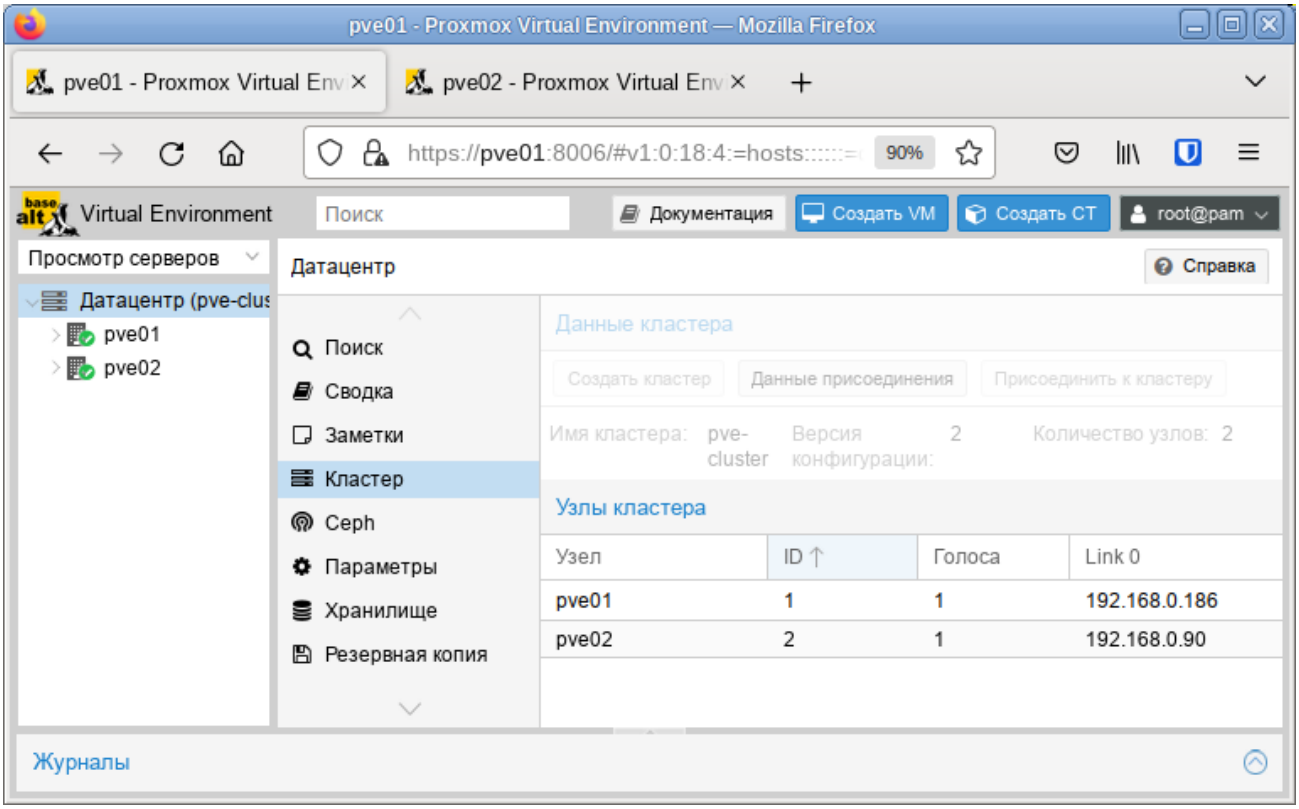


Рис. 64

Сразу после инициализации кластера в пределах каждого из узлов доступно одно локальное хранилище данных (Рис. 65).

Узлы кластера и локальные хранилища данных

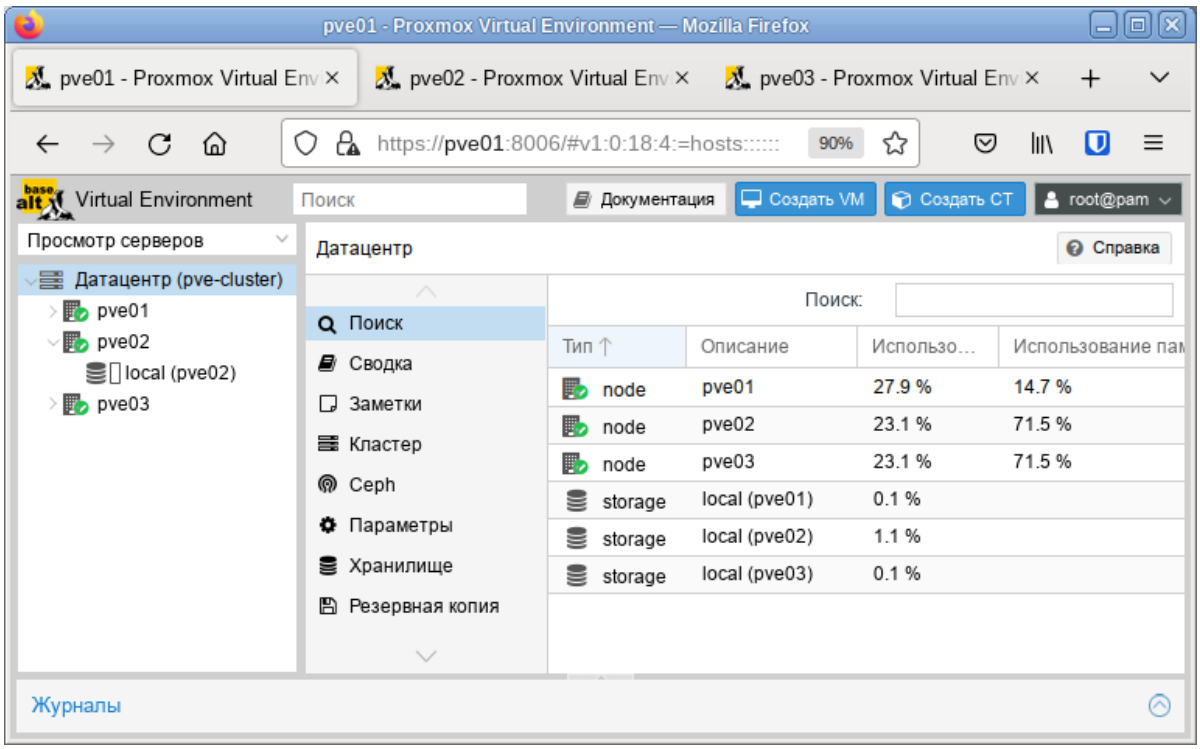


Рис. 65

4.3.3 Создание кластера в консоли

Команда, создания кластера:

```
# pvecm create <cluster_name>
```

На «головном» узле кластера необходимо выполнить команду инициализации конкретного кластера PVE, в данном примере – «pve-cluster»:

```
# systemctl start pve-cluster
# pvecm create pve-cluster
```

Проверка:

```
# pvecm status
Cluster information
-------------------
Name:             pve-cluster
Config Version:   1
Transport:        knet
Secure auth:      on

Quorum information
------------------
Date:             Thu Dec 22 10:02:31 2022
Quorum provider:  corosync_votequorum
Nodes:            1
Node ID:          0x00000001
Ring ID:          1.d6
Quorate:          Yes

Votequorum information
----------------------
Expected votes:   1
Highest expected: 1
Total votes:      1
Quorum:           1
Flags:            Quorate

Membership information
----------------------
```

```
    Nodeid      Votes Name
0x00000001          1 192.168.0.186 (local)
```

Команда создания кластера создает файл настройки `/etc/pve/corosync.conf`. По мере добавления узлов в кластер файл настройки будет автоматически пополняться информацией об узлах.

Команда, для добавления узла в кластер:

```
# pvecm add <existing_node_in_cluster>
```

где `existing_node_in_cluster` – адрес уже добавленного узла (рекомендуется указывать самый первый).

Для добавления узлов в кластер, необходимо на «подчиненных» узлах выполнить команде:

```
# pvecm add pve01
```

где `pve01` – имя или IP-адрес «головного» узла.

При добавлении узла в кластер, потребуется ввести пароль администратора главного узла кластера:

```
# pvecm add pve01
Please enter superuser (root) password for 'pve01': ***
Establishing API connection with host 'pve01'

Login succeeded.
Request addition of this node
Join request OK, finishing setup locally
stopping pve-cluster service
backup old database to '/var/lib/pve-cluster/backup/config-
1625747072.sql.gz'
waiting for quorum...OK
(re)generate node files
generate new node certificate
merge authorized SSH keys and known hosts
generated new node certificate, restart pveproxy and pvedaemon
services
successfully added node 'pve03' to cluster.
```

После добавления узлов в кластер, файл настройки кластера в `/etc/pve/corosync.conf` должен содержать информацию об узлах кластера.

На всех узлах кластера должны быть запущены и добавлены в список служб, запускаемых при старте узла, службы:

```
# systemctl start lxc lxc-net lxc-monitord pve-lxc-syscalld pvedaemon
pve-firewall pvestatd pve-ha-lrm pve-ha-crm spiceproxy pveproxy
# systemctl enable corosync lxc lxc-net lxc-monitord pve-lxc-syscalld
pve-cluster pvedaemon pve-firewall pvestatd pve-ha-lrm pve-ha-crm
spiceproxy pveproxy pve-guests
```

4.3.4 Удаление узла из кластера

Перед удалением узла из кластера необходимо переместить все ВМ с этого узла, а также убедиться, что нет никаких локальных данных или резервных копий, которые необходимо сохранить.

Для удаления узла из кластера необходимо выполнить следующие шаги:

1) войти в узел кластера не подлежащий удалению;

2) ввести команду `pvecm nodes`, чтобы определить идентификатор узла, который следует удалить:

```
# pvecm nodes
Membership information
----------------------
    Nodeid      Votes Name
         1          1 pve01 (local)
         2          1 pve02
         3          1 pve03
```

3) выключить подлежащий удалению узел (в данном случае pve02);

4) удалить узел из кластера, выполнив команду:

```
# pvecm delnode pve02
```

5) проверить, что узел удален (команда отобразит список узлов кластера без удаленного узла):

```
# pvecm nodes
Membership information
----------------------
    Nodeid      Votes Name
         1          1 pve01 (local)
         3          1 pve03
```

Если необходимо вернуть удаленный узел обратно в кластер, следует выполнить следующие действия:

- переустановить PVE на этом узле (это гарантирует, что все секретные ключи кластера/ssh и любые данные конфигурации будут уничтожены);

– присоединиться к кластеру.

4.3.5 Кластерная файловая система PVE (pmxcfs)

Кластерная файловая система PVE (pmxcfs) – это файловая система на основе базы данных для хранения файлов конфигурации виртуальных окружений, реплицируемая в режиме реального времени на все узлы кластера с помощью `corosync`. Эта файловая система используется для хранения всех конфигурационных файлов связанных с PVE. Хотя файловая система хранит все данные в базе данных на диске, копия данных находится в оперативной памяти, что накладывает ограничение на максимальный размер данных, который в настоящее время составляет 30 МБ.

Преимущества файловой системы pmxcfs:

– мгновенная репликация и обновление конфигурации на все узлы в кластере;
– исключается вероятность дублирования идентификаторов виртуальных машин;
– в случае развала кворума в кластере, файловая система становится доступной только для чтения.

Все файлы и каталоги принадлежат пользователю root и имеют группу www-data. Только root имеет права на запись, но пользователи из группы www-data могут читать большинство файлов. Файлы в каталогах `/etc/pve/priv/` и `/etc/pve/nodes/${NAME}/priv/` доступны только root.

Файловая система смонтирована в `/etc/pve/`.

4.4 Системы хранения

Образы ВМ могут храниться в одном или нескольких локальных хранилищах или в общем (совместно используемом) хранилище, например NFS или iSCSI (NAS, SAN). Ограничений нет, можно настроить столько хранилищ, сколько необходимо.

В кластерной среде PVE наличие общего хранилища не является обязательным, однако оно делает управление хранением более простой задачей. Преимущества общего хранилища:

– миграция ВМ в реальном масштабе времени;
– плавное расширение пространства хранения с множеством узлов;
– централизованное резервное копирование;
– многоуровневое кэширование данных;
– централизованное управление хранением.

4.4.1 Типы хранилищ в PVE

Существует два основных типа хранилищ:

– файловые хранилища – хранят данные в виде файлов. Технологии хранения на уровне файлов обеспечивают доступ к полнофункциональной файловой системе (POSIX). В целом они

более гибкие, чем любое хранилище на уровне блоков, и позволяют хранить контент любого типа;

– блочное хранилище – позволяет хранить большие необработанные образы. Обычно в таких хранилищах невозможно хранить другие файлы (ISO-образы, резервные копии, и т.д.). Большинство современных реализаций хранилищ на уровне блоков поддерживают моментальные снимки и клонирование. RADOS и GlusterFS являются распределенными системами, реплицирующими данные хранилища на разные узлы.

Хранилищами данных удобно управлять через веб-интерфейс. В качестве бэкенда хранилищ PVE может использовать:

– сетевые файловые системы, в том числе распределенные: NFS, CEPH, GlusterFS;
– локальные системы управления дисковыми томами: LVM, ZFS;
– удаленные (iSCSI) и локальные дисковые тома;
– локальные дисковые каталоги.

Доступные типы хранилищ приведены в табл. 3.

Т а б л и ц а 3 – Доступные типы хранилищ

Хранилище	PVE тип	Уровень	Общее (shared)	Снимки (snapshots)
ZFS (локальный)	zfspool	файл	нет	да
Каталог	dir	файл	нет	нет (возможны в формате qcow2)
BTRFS	btrfs	файл	нет	да
NFS	nfs	файл	да	нет (возможны в формате qcow2)
CIFS	cifs	файл	да	нет (возможны в формате qcow2)
GlusterFS	glusterfs	файл	да	нет (возможны в формате qcow2)
CephFS	cephfs	файл	да	да
LVM	lvm	блок	нет	нет
LVM-thin	lvmthin	блок	нет	да
iSCSI/kernel	iscsi	блок	да	нет
iSCSI/libiscsi	iscsidirect	блок	да	нет
Ceph/RBD	rbd	блок	да	да
ZFS over iSCSI	zfs	блок	да	да
Proxmox Backup	pbs	файл/блок	да	-

4.4.2 Конфигурация хранилища

Вся связанная с PVE информация о хранилищах хранится в файле `/etc/pve/storage.cfg`. Поскольку этот файл находится в `/etc/pve/`, он автоматически распространяется на все узлы кластера. Таким образом, все узлы имеют одинаковую конфигурацию хранилища.

Совместное использование конфигурации хранилища имеет смысл для общего хранилища, поскольку одно и то же «общее» хранилище доступно для всех узлов. Но также полезно для локальных типов хранения. В этом случае такое локальное хранилище доступно на всех узлах, но оно физически отличается и может иметь совершенно разное содержимое.

Каждое хранилище имеет <тип> и уникально идентифицируется своим <STORAGE_ID>. Конфигурация хранилища выглядит следующим образом:

```
<type>: <STORAGE_ID>
        <property> <value>
        <property> <value>
        ...
```

Строка `<type>: <STORAGE_ID>` определяет хранилище, затем следует список свойств.

Пример файла `/etc/pve/storage.cfg`:

```
# cat /etc/pve/storage.cfg
dir: local
     path /var/lib/vz
     content images,rootdir,iso,snippets,vztmpl
     maxfiles 0
nfs: nfs-storage
     export /export/storage
     path /mnt/nfs-vol
     server 192.168.0.105
     content images,iso,backup,vztmpl
     options vers=3,nolock,tcp
```

В данном случае файл содержит описание специального хранилища local, которое ссылается на каталог `/var/lib/vz` и NFS хранилище nfs-storage.

Некоторые параметры являются общими для разных типов хранилищ (табл. 4).

Т а б л и ц а 4 – Параметры хранилищ

Свойство	Описание
nodes	Список узлов кластера, где хранилище можно использовать/доступно. Можно использовать это свойство, чтобы ограничить доступ к хранилищу
content	Хранилище может поддерживать несколько типов содержимого. Это свойство указывает, для чего используется это хранилище. Доступные опции: – images – образы виртуальных дисков; – rootdir – данные контейнеров; – vztmpl – шаблоны контейнеров; – backup – резервные копии (vzdump);

	– iso – ISO-образы; – snippets – файлы сниппетов
shared	Пометить хранилище как общее
disable	Отключить хранилище
maxfiles	Устарело, следует использовать свойство prune-backups. Максимальное количество файлов резервных копий на ВМ
prune-backups	Варианты хранения резервных копий
format	Формат образа по умолчанию (raw|qcow2|vmdk)

4.4.3 Работа с хранилищами в PVE

4.4.3.1 Использование командной строки

Утилита `pvesm` (PVE Storage Manager), позволяет выполнять общие задачи управления хранилищами.

Команды добавления (подключения) хранилища:

```
# pvesm add <TYPE> <STORAGE_ID> <OPTIONS>
# pvesm add dir <STORAGE_ID> --path <PATH>
# pvesm add nfs <STORAGE_ID> --path <PATH> --server <SERVER> --export <EXPORT>
# pvesm add lvm <STORAGE_ID> --vgname <VGNAME>
# pvesm add iscsi <STORAGE_ID> --portal <HOST[:PORT]> --target <TARGET>
```

Отключить хранилище:

```
# pvesm set <STORAGE_ID> --disable 1
```

Включить хранилище:

```
# pvesm set <STORAGE_ID> --disable 0
```

Для того чтобы изменить/установить опции хранилища можно, выполнить команды:

```
# pvesm set <STORAGE_ID> <OPTIONS>
# pvesm set <STORAGE_ID> --shared 1
# pvesm set local --format qcow2
# pvesm set <STORAGE_ID> --content iso
```

Удалить хранилище (при этом никакие данные не удаляются, удаляется только конфигурация хранилища):

```
# pvesm remove <STORAGE_ID>
```

Команда выделения тома:

```
# pvesm alloc <STORAGE_ID> <VMID> <name> <size> [--format <raw|qcow2>]
```

Выделить том 4 ГБ в локальном хранилище (имя будет сгенерировано):

```
# pvesm alloc local <VMID> '' 4G
```

Освободить место (уничтожает все данные тома):

```
# pvesm free <VOLUME_ID>
```

Вывести список хранилищ:

```
# pvesm status
```

Вывести список содержимого хранилища:

```
# pvesm list <STORAGE_ID> [--vmid <VMID>]
```

4.4.3.2 *Добавление хранилища в веб-интерфейсе PVE*

Хранилища, которые могут быть добавлены в веб-интерфейсе PVE (Рис. 66):

- Локальные хранилища:
  - Каталог – хранение на существующей файловой системе;
  - LVM – локальные устройства, такие как, FC, DRBD и т.д.;
  - ZFS;
  - BTRFS;
- Сетевые хранилища:
  - LVM – сетевая поддержка с iSCSI target;
  - NFS;
  - CIFS;
  - GlusterFS;
  - iSCSI;
  - CephFS;
  - RBD;
  - ZFS over iSCSI.

Выбор типа добавляемого хранилища

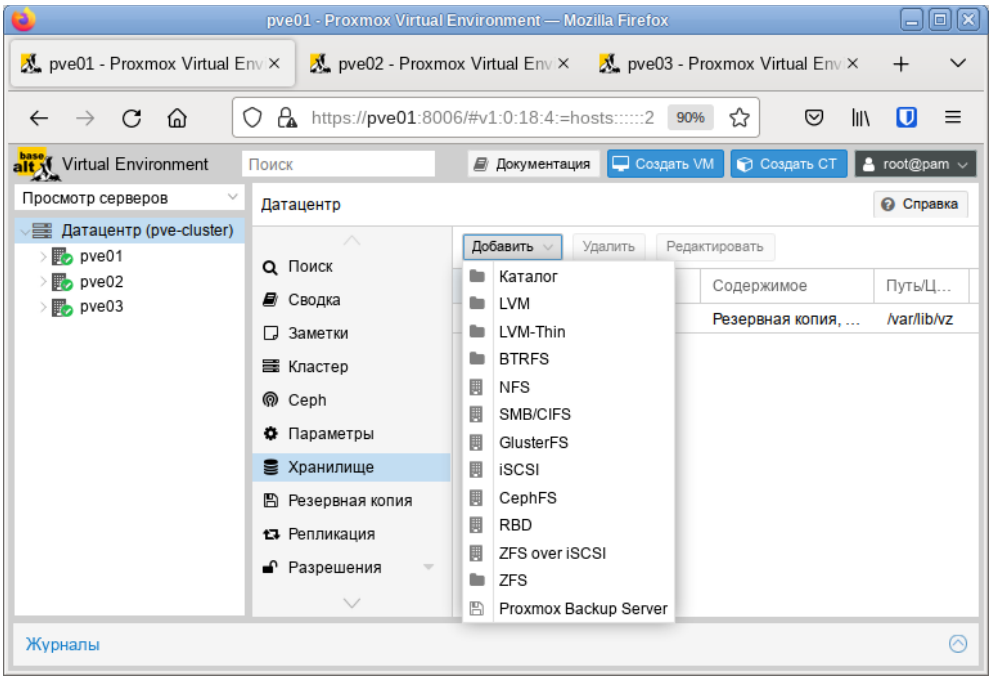


Рис. 66

При создании каждому хранилищу данных присваивается роль или набор ролей. Например, хранение контейнеров, образов виртуальных дисков, файлов .iso и так далее. Список возможных ролей зависит от бэкенда хранилища. Например, для NFS или каталога локальной файловой системы доступны любые роли или наборы ролей (*Рис. 67*), а хранилища на базе CEPH можно использовать только для хранения ISO-образов или шаблонов контейнеров.

Выбор ролей для хранилища

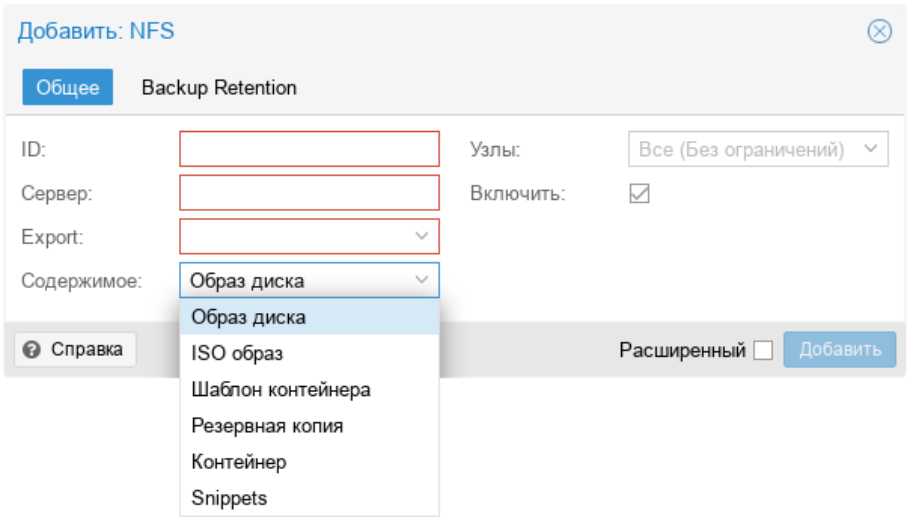


Рис. 67

4.4.3.3 Каталог (Directory)

PVE может использовать локальные каталоги или локально смонтированные общие ресурсы для организации хранилища. Каталог – это файловое хранилище, поэтому в нем можно хранить данные любого типа, например образы виртуальных дисков, контейнеры, шаблоны, ISO-образы или файлы резервных копий. Для хранения данных разного типа, используется предопределенная структура каталогов (табл. 5). Эта структура используется на всех файловых хранилищах.

Т а б л и ц а 5 – Структура каталогов.

Тип данных	Подкаталог
Образы дисков ВМ	images/<VMID>/
ISO-образы	template/iso/
Шаблоны контейнеров	template/cache/
Резервные копии	dump/
Snippets	snippets/

Для создания нового хранилища типа «Каталог» необходимо выбрать «Датацентр» → «Хранилище» («Datacenter» → «Storage»), нажать кнопку «Добавить» («Add») и в выпадающем меню выбрать пункт «Каталог» («Directory») (Рис. 66). На Рис. 68 показан диалог создания храни-

лища local-iso типа «Каталог» для хранения ISO-образов и шаблонов контейнеров, которое будет смонтировано в каталог `/mnt/iso`.

Добавление хранилища «Каталог»

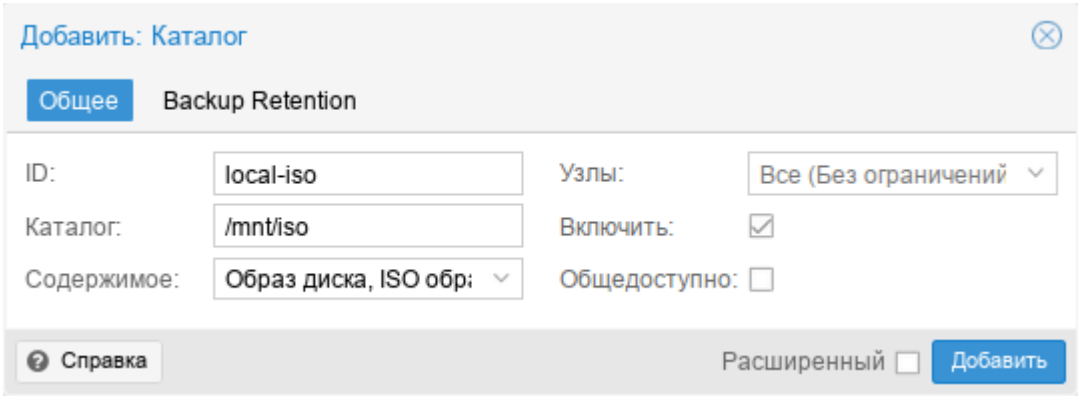


Рис. 68

Данное хранилище поддерживает все общие свойства хранилищ и дополнительно свойство `path` для указания каталога. Это должен быть абсолютный путь к файловой системе.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
dir: backup
     path /mnt/backup
     content backup
     prune-backups keep-last=7
     shared 0
```

Данная конфигурация определяет пул хранения резервных копий. Этот пул может использоваться для хранения последних 7 резервных копий на виртуальную машину. Реальный путь к файлам резервных копий – `/mnt/backup/dump/...`.

Хранилище «Каталог» использует четко определенную схему именования образов ВМ:

```
VM-<VMID>-<NAME>.<FORMAT>
```

где:

<VMID> – ID виртуальной машины;

<NAME> – произвольное имя (ascii) без пробелов. По умолчанию используется disk-[N], где [N] заменяется целым числом.

<FORMAT> – определяет формат образа (raw|qcow2|vmdk).

Пример:

```
# ls /var/lib/vz/images/101
vm-101-disk-0.qcow2  vm-101-disk-1.qcow2
```

При создании шаблона ВМ все образы дисков ВМ переименовываются, чтобы указать, что они теперь доступны только для чтения и могут использоваться в качестве базового образа для клонов:

`base-<VMID>-<NAME>.<FORMAT>`

4.4.3.4 NFS

Хранилище NFS аналогично хранению каталогов и файлов на диске, с дополнительным преимуществом совместного хранения и миграции в реальном времени. Свойства хранилища NFS во многом совпадают с хранилищем типа «Каталог». Структура каталогов и соглашение об именах файлов также одинаковы. Основным преимуществом является то, что можно напрямую настроить свойства сервера NFS, и общий ресурс будет монтироваться автоматически.

Данное хранилище поддерживает все общие свойства хранилищ кроме флага `shared`, который всегда установлен. Кроме того, для настройки NFS используются следующие свойства:

- server – IP-адрес сервера или DNS-имя. Предпочтительнее использовать IP-адрес вместо DNS-имени (чтобы избежать задержек при поиске DNS);
- export – совместный ресурс с сервера NFS (список можно просмотреть, выполнив команду `pvesm scan nfs <server>`);
- path – локальная точка монтирования (по умолчанию `/mnt/pve/<STORAGE_ID>/`);
- options – параметры монтирования NFS.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
nfs: nfs-storage
     export /export/storage
     path /mnt/pve/nfs-storage
     server 192.168.0.105
     content images,iso,backup,vztmpl
     options vers=3,nolock,tcp
```

Примечание. Для возможности монтирования NFS хранилища должен быть запущен nfs-client:

```
# systemctl enable --now nfs-client.target
```

На *Рис. 69* показано присоединение хранилища NFS с именем nfs-storage с удаленного сервера 192.168.0.105.

После ввода IP-адреса удаленного сервера, доступные ресурсы будут автоматически просканированы и будут отображены в выпадающем списке «Export». В данном примере обнаруженная в блоке диалога точка монтирования – `/export/storage`.

Пример добавления хранилища NFS в командной строке с помощью утилиты `pvesm`:

```
# pvesm add nfs nfs-storage --path /mnt/pve/nfs-storage --server 192.168.0.105 --options vers=3,nolock,tcp --export /export/storage --content images,iso,vztmpl,backup
```

Создание хранилища NFS

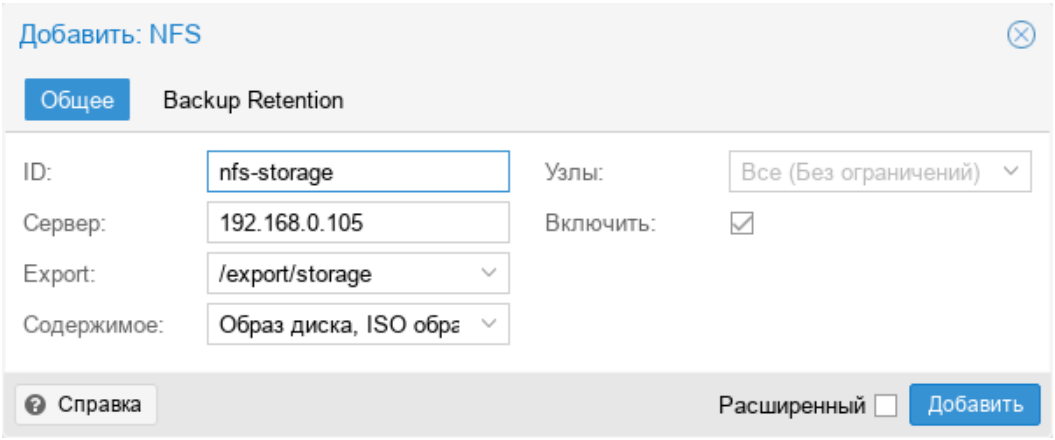


Рис. 69

Получить список совместных ресурсов с сервера NFS:

```
# pvesm nfsscan <server>
```

4.4.3.5 BTRFS

Свойства хранилища BTRFS во многом совпадают с хранилищем типа Каталог. Основное отличие состоит в том, с этим типом хранилища диски в формате raw будут помещены в subvolume, для возможности создания снимков (снапшотов) и поддержки автономной миграции хранилища с сохранением снимков.

Примечание. BTRFS учитывает флаг O_DIRECT при открытии файлов, что означает, что ВМ не должны использовать режим кеширования none, иначе возникнут ошибки контрольной суммы.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
btrfs: btrfs-storage
   path /mnt/data
   content rootdir,images
   nodes pve02
   prune-backups keep-all=1
```

На *Рис. 70* показан диалог создания хранилища brtfs-storage типа BTRFS для хранения образов дисков и контейнеров.

Пример добавления хранилища BTRFS в командной строке с помощью утилиты `pvesm`:

```
# pvesm add btrfs btrfs-storage --path /mnt/data/btrfs-storage --is_mountpoint / --content images,rootdir
```

Создание хранилища BTRFS

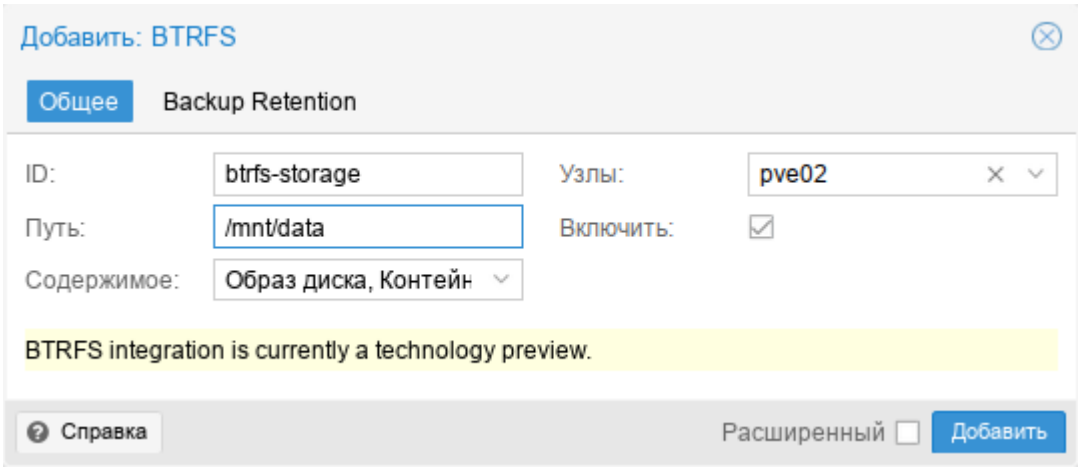


Рис. 70

4.4.3.6 SMB/CIFS

Хранилище SMB/CIFS расширяет хранилище типа «Каталог», поэтому ручная настройка монтирования CIFS не требуется.

Примечание. Для возможности просмотра общих ресурсов на каждом узле кластера необходимо установить пакет `samba-client`.

Данное хранилище поддерживает все общие свойства хранилищ кроме флага `shared`, который всегда установлен. Кроме того, для настройки CIFS используются следующие свойства:

- server – IP-адрес сервера или DNS-имя. Предпочтительнее использовать IP-адрес вместо DNS-имени (чтобы избежать задержек при поиске DNS);
- share – совместный ресурс с сервера CIFS (список можно просмотреть, выполнив команду `pvesm scan cifs <server>`);
- username – имя пользователя для хранилища CIFS (необязательно, по умолчанию «guest»);
- password – пароль пользователя (необязательно). Пароль будет сохранен в файле, доступном только для чтения root-пользователю (`/etc/pve/priv/<STORAGE_ID>.cred`);
- domain – устанавливает домен пользователя (рабочую группу) для этого хранилища (необязательно);
- smbversion – версия протокола SMB (необязательно, по умолчанию 3);
- path – локальная точка монтирования (по умолчанию `/mnt/pve/<STORAGE_ID>/`).

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
cifs: newCIFS
      path /mnt/pve/newCIFS
      server 192.168.0.105
      share smb_data
```

Получить список совместных ресурсов с сервера CIFS можно, выполнив команду:

```
# pvesm cifsscan <server> [--username <username>] [--password]
```

Команда добавления общего ресурса в качестве хранилища:

```
# pvesm add cifs <storagename> --server <server> --share <share> [--username <username>] [--password]
```

На *Рис. 71* показано присоединение хранилища SMB/CIFS с именем newCIFS с удаленного сервера 192.168.0.105.

Добавление CIFS хранилища

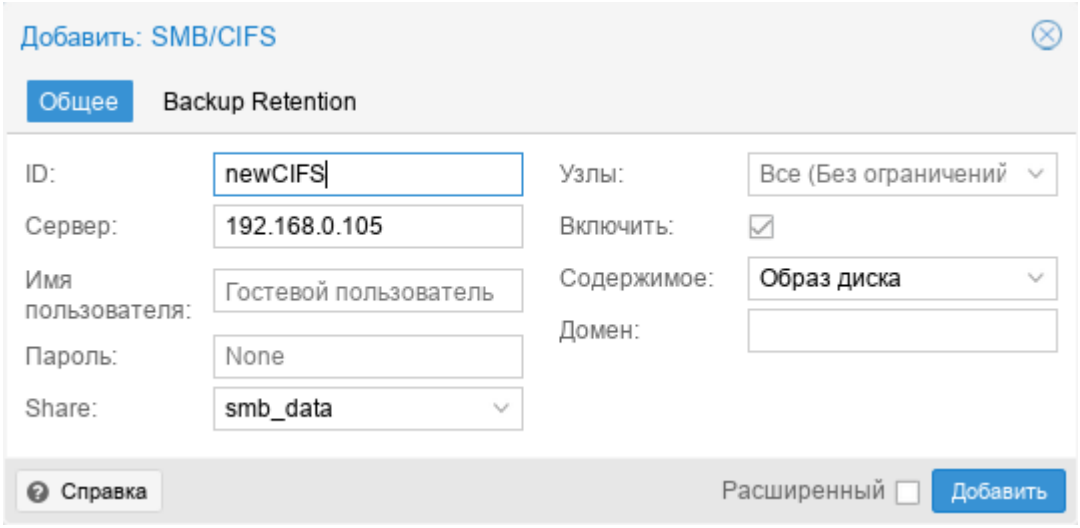


Рис. 71

После ввода IP-адреса удаленного сервера, доступные ресурсы будут автоматически просканированы и будут отображены в выпадающем списке «Share».

Примечание. При создании CIFS хранилища в веб-интерфейсе, PVE предполагает, что удаленный сервер поддерживает CIFS v3. В веб-интерфейсе нет возможности указать версию CIFS, поэтому в случае, если удалённый сервер поддерживает версии CIFS отличные от v3, создать хранилище можно в командной строке, например:

```
# pvesm add cifs newCIFS --server 192.168.0.105 --share smb_data --smbversion 2.1
```

4.4.3.7 GlusterFS

GlusterFS – это масштабируемая сетевая файловая система. Система использует модульную конструкцию, работает на аппаратном оборудовании и может обеспечить высокодоступное корпоративное хранилище при низких затратах. Такая система способна масштабироваться до нескольких петабайт и может обрабатывать тысячи клиентов.

Данное хранилище поддерживает все общие свойства хранилищ, и дополнительно используются следующие свойства:

– server – IP-адрес или DNS-имя сервера GlusterFS;

– server2 – IP-адрес или DNS-имя резервного сервера GlusterFS;
– volume – том GlusterFS;
– transport – транспорт GlusterFS: tcp, unix или rdma.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
glusterfs: gluster-01
            server 192.168.0.105
            server2 192.168.0.110
            volume glustervol
            content images,iso
```

На *Рис. 72* показано присоединение хранилища GlusterFS с именем gluster-01 с удаленного сервера 192.168.0.105.

Создание хранилища GlusterFS

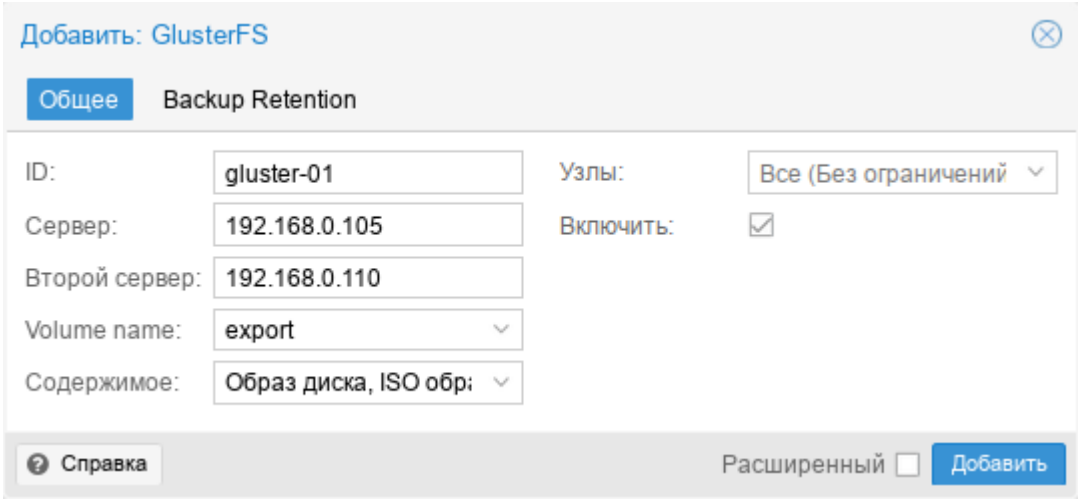


Рис. 72

4.4.3.8 Локальный ZFS

Примечание. Для работы с локальным ZFS хранилищем должен быть установлен модуль ядра `kernel-modules-zfs-std-def`. Включить модуль:

```
# modprobe zfs
```

Чтобы не вводить эту команду после перезагрузки, следует раскомментировать строку: `#zfs` в файле `/etc/modules-load.d/zfs.conf`.

Локальный ZFS позволяет получить доступ к локальным пулам ZFS (или файловым системам ZFS внутри таких пулов). Данное хранилище поддерживает все общие свойства хранилищ, кроме того, для настройки ZFS используются следующие свойства:

– pool – пул/файловая система ZFS;
– blocksize – размер блока;
– sparse – использовать тонкую инициализацию ZFS.

Пул ZFS поддерживает следующие типы RAID:

- RAID-0 (Single Disk) – размер такого пула – сумма емкостей всех дисков. RAID0 не добавляет избыточности, поэтому отказ одного диска делает том не пригодным для использования (минимально требуется один диск);
- пул RAID-1 (Mirror) – данные зеркалируются на все диски (минимально требуется два диска);
- пул RAID-10 – сочетание RAID0 и RAID1 (минимально требуется четыре диска);
- пул RAIDZ-1 – вариация RAID-5, одинарная четность (минимально требуется три диска);
- пул RAIDZ-2 – вариация на RAID-5, двойной паритет (минимально требуется четыре диска);
- пул RAIDZ-3 – разновидность RAID-5, тройная четность (минимально требуется пять дисков).

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
zfspool: vmdata
        pool vmdata
        content images,rootdir
        mountpoint /vmdata
        nodes pve03
```

Возможные типы содержимого: `rootdir` (данные контейнера), `images` (образ виртуального диска в формате raw или subvol).

Используется следующая схема именования образов дисков ВМ:

- vm-<VMID>-<NAME> – образ ВМ;
- base-<VMID>-<NAME> – шаблон образа ВМ (только для чтения);
- subvol-<VMID>-<NAME> – файловая система ZFS для контейнеров.

4.4.3.8.1 Создание локального хранилища ZFS в веб-интерфейсе

Для создания локального хранилища ZFS в веб-интерфейсе, следует выбрать узел, на котором будет создано хранилище, в разделе «Диски» («Disks») выбрать пункт «ZFS» и нажать кнопку «Создать: ZFS» (*Рис. 73*).

В открывшемся окне (*Рис. 74*) следует задать параметры ZFS хранилища: имя хранилища, выбрать диски, уровень RAID и нажать кнопку «Создать».

Статус пула можно просмотреть выбрав его в списке и нажав кнопку «Подробно» (Рис. 75).

Добавление ZFS хранилища

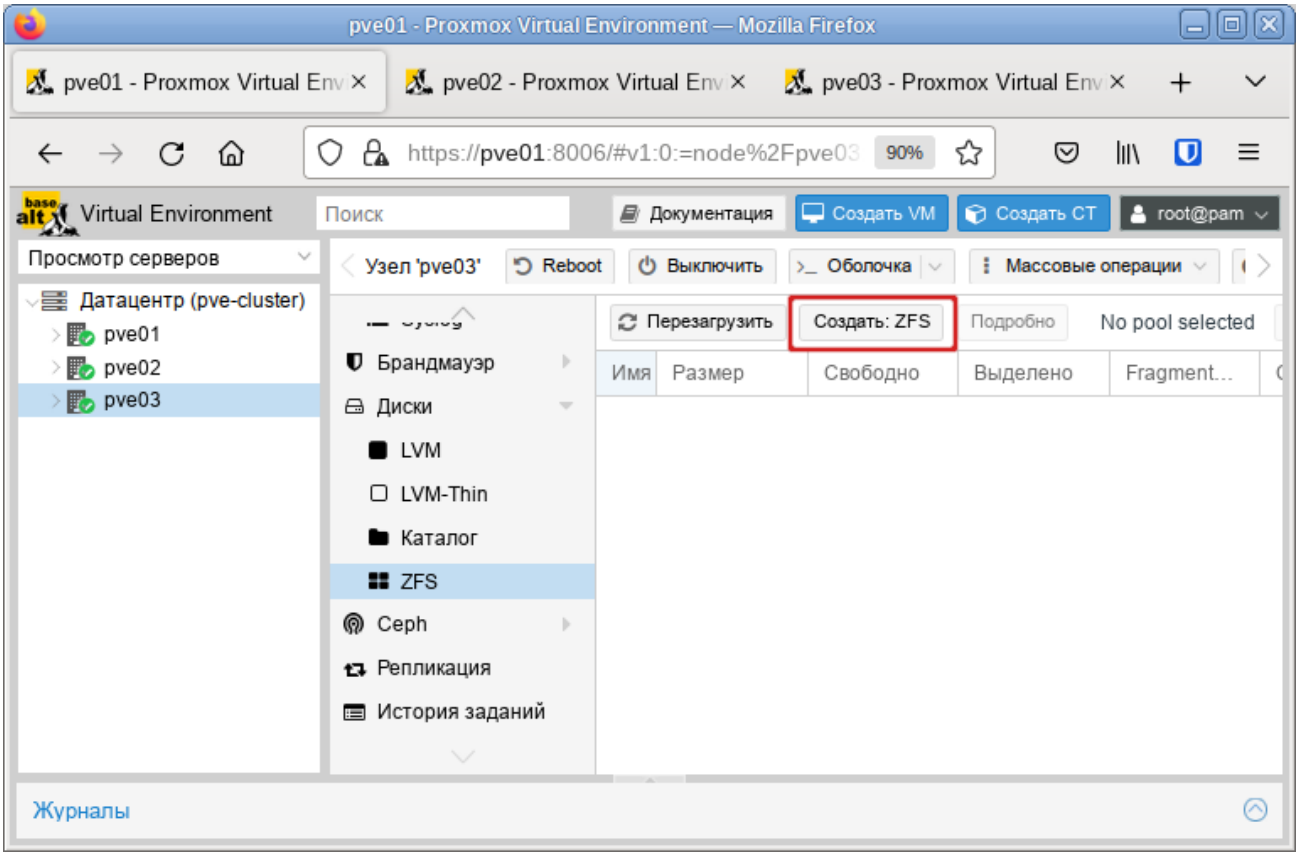


Рис. 73

Параметры ZFS хранилища

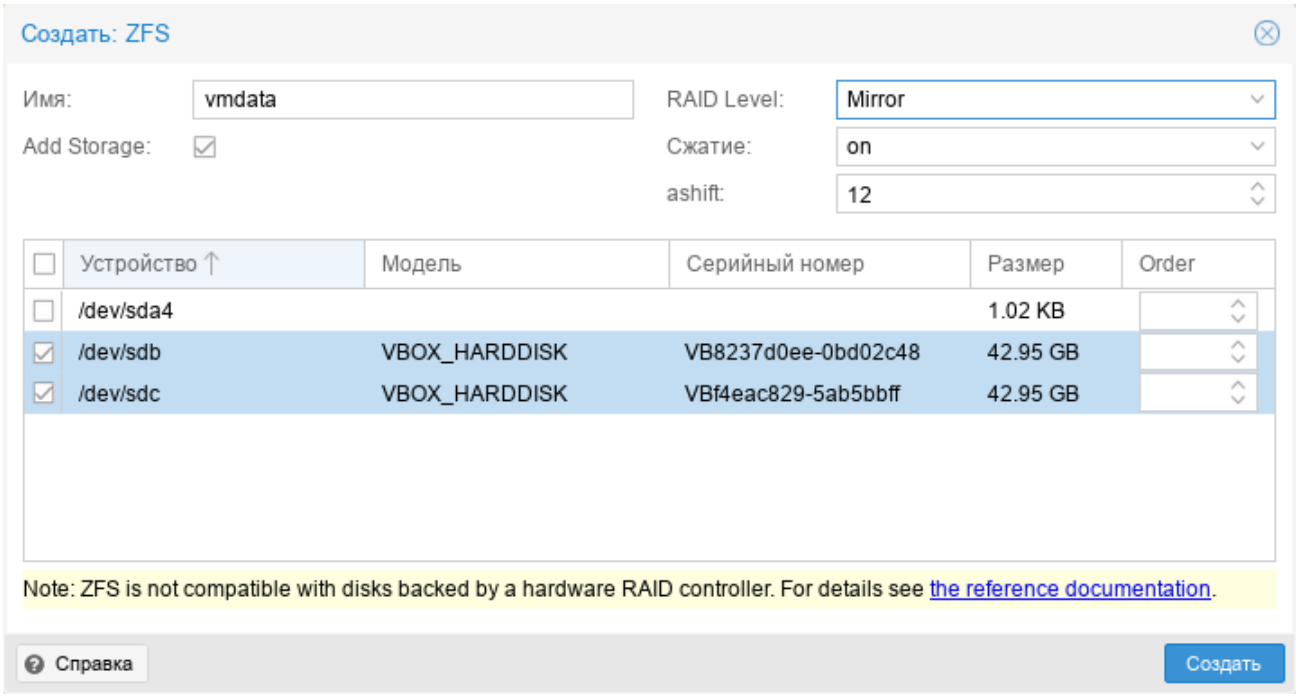


Рис. 74

Локальное ZFS хранилище

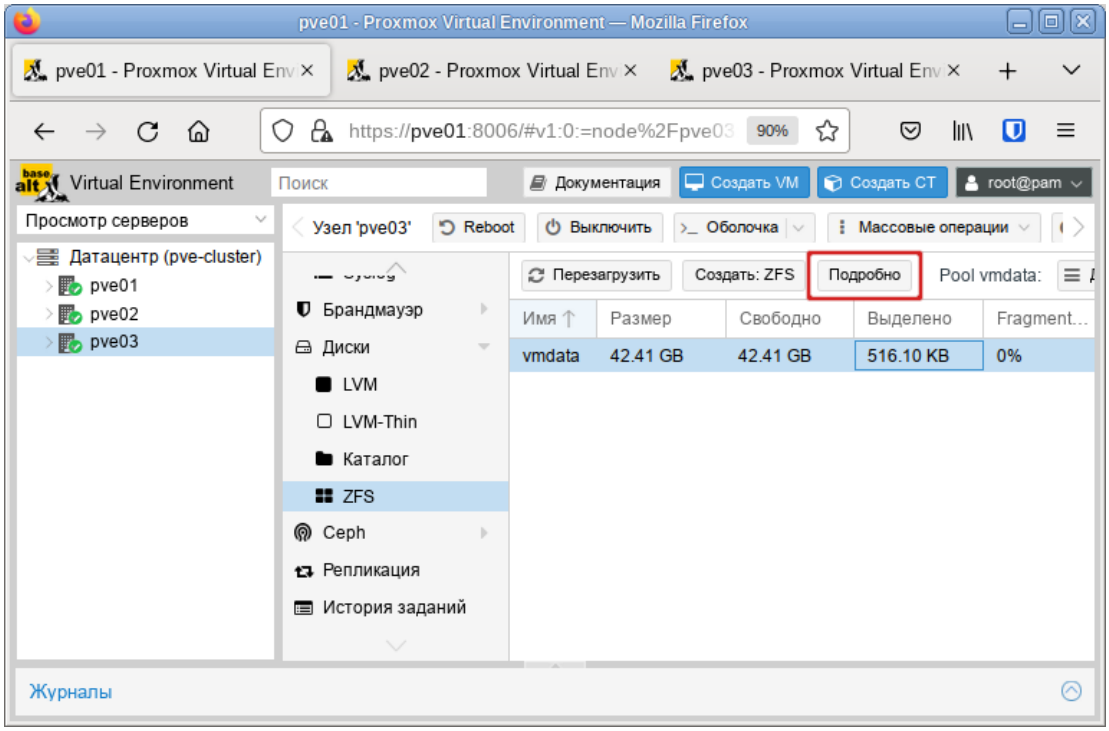


Рис. 75

Для того чтобы внести изменения в настройки ZFS хранилища следует выбрать «Датацентр» → «Хранилище» («Datacenter» → «Storage»), затем нужное хранилище и нажать кнопку «Редактировать» (*Рис. 76*). В открывшемся окне (*Рис. 77*) можно изменить тип содержимого контейнера, включить/отключить хранилище, включить дисковое резервирование.

Выбор хранилища для редактирования

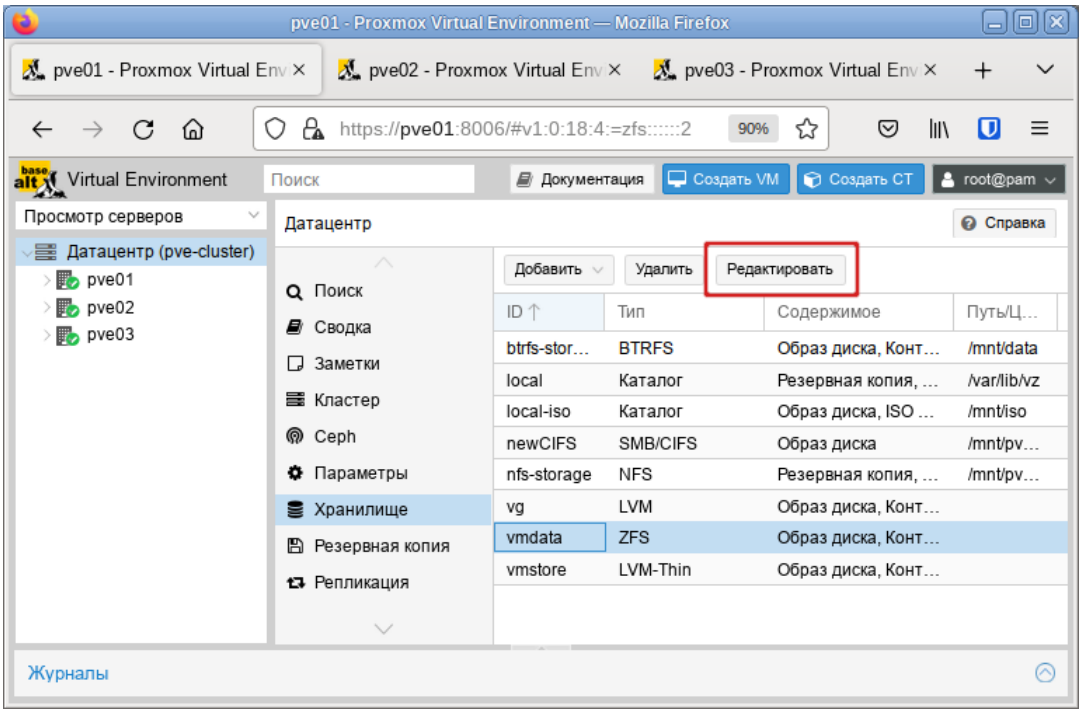


Рис. 76

Редактирование ZFS хранилища

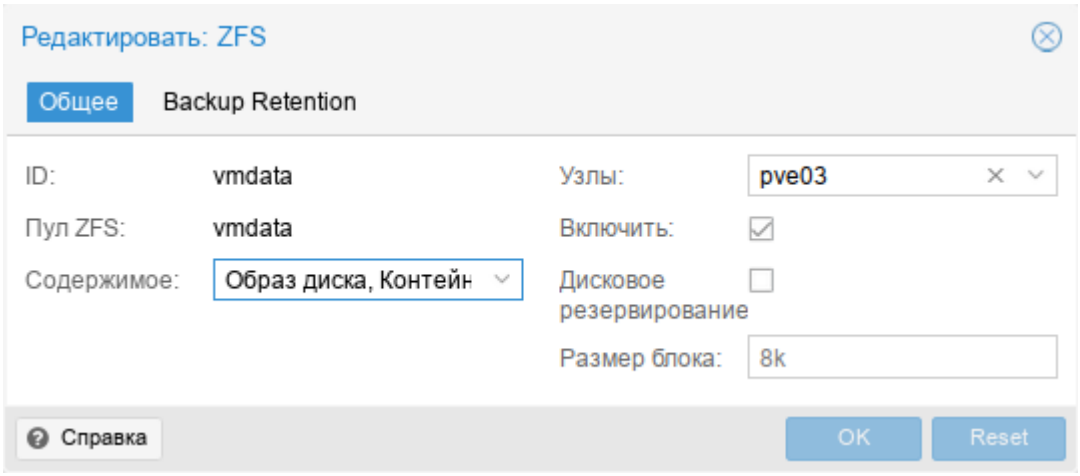


Рис. 77

4.4.3.8.2 Администрирование ZFS

Основными командами для управления ZFS являются zfs и zpool.

Для создания нового пула необходим как минимум один пустой диск.

Создание нового пула RAID-0 (минимум 1 диск):

```
# zpool create -f -o ashift=12 <pool> <device1> <device2>
```

Создание нового пула RAID-1 (минимум 2 диска):

```
# zpool create -f -o ashift=12 <pool> mirror <device1> <device2>
```

Создание нового пула RAID-10 (минимум 4 диска):

```
# zpool create -f -o ashift=12 <pool> mirror <device1> <device2>
mirror <device3> <device4>
```

Создание нового пула RAIDZ-1 (минимум 3 диска):

```
# zpool create -f -o ashift=12 <pool> raidz1 <device1> <device2>
<device3>
```

Создание нового пула RAIDZ-2 (минимум 4 диска):

```
# zpool create -f -o ashift=12 <pool> raidz2 <device1> <device2>
<device3> <device4>
```

Смена неисправного устройства:

```
# zpool replace -f <pool> <old device> <new device>
```

Включить сжатие:

```
# zfs set compression=on <pool>
```

Получить список доступных ZFS файловых систем:

```
# pvesm zfsscan
```

Пример создания RAID1(mirror) с помощью zfs:

```
# zpool create -f vmdata mirror sdb sdc
```

Просмотреть созданные в системе пулы:

```
# zpool list
NAME     SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG   CAP  DEDUP  HEALTH  ALTROOT
vmdata  17,5G   492K  17,5G        -         -     0%    0%  1.00x  ONLINE  -
```

Просмотреть статус пула:

```
# zpool status
  pool: vmdata
 state: ONLINE
  scan: none requested
config:

      NAME        STATE     READ WRITE CKSUM
      vmdata      ONLINE       0     0     0
        mirror-0  ONLINE       0     0     0
          sdb     ONLINE       0     0     0
          sdc     ONLINE       0     0     0

errors: No known data errors
```

4.4.3.9 LVM

LVM (Logical Volume Management) это система управления дисковым пространством. Позволяет логически объединить несколько дисковых пространств (физические тома) в одно, и уже из этого пространства (дисковой группы или группы томов – VG), можно выделять разделы (логические тома – LV), доступные для работы.

Использование LVM групп обеспечивает лучшую управляемость. Логические тома можно легко создавать/удалять/перемещать между физическими устройствами хранения. Если база хранения для группы LVM доступна на всех PVE узлах (например, ISCSI LUN) или репликах (например, DRBD), то все узлы имеют доступ к образам ВМ, и возможна live-миграция.

Данное хранилище поддерживает все общие свойства хранилищ, кроме того, для настройки LVM используются следующие свойства:

- vgname – имя группы томов LVM (должно указывать на существующую группу томов);
- base – базовый объем;
- saferemove – обнуление данных при удалении LV. При удалении тома это гарантирует, что все данные будут удалены;
- saferemove_throughput – очистка пропускной способности (значение параметра `cstream -t`).

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
lvm: vg
```

```
vgname vg
content rootdir,images
nodes pve03
shared 0
```

Возможные типы содержимого: `rootdir` (данные контейнера), `images` (образ виртуального диска в формате raw).

4.4.3.9.1 Создание локального хранилища LVM в веб-интерфейсе

Примечание. Для создания локального LVM хранилища в веб-интерфейсе необходимо чтобы в системе имелся хотя бы один пустой жесткий диск.

Для создания локального LVM хранилища в веб-интерфейсе, следует выбрать узел, на котором будет создано хранилище, в разделе «Диски» («Disks») выбрать пункт «LVM» и нажать кнопку «Создать: Volume Group» (*Рис. 78*). В открывшемся окне (*Рис. 79*) следует выбрать диск, задать имя группы томов, отметить пункт «Add Storage» (если этот пункт не отмечен будет создана только группа томов) и нажать кнопку «Создать».

Пункт «LVM» в разделе «Диски»

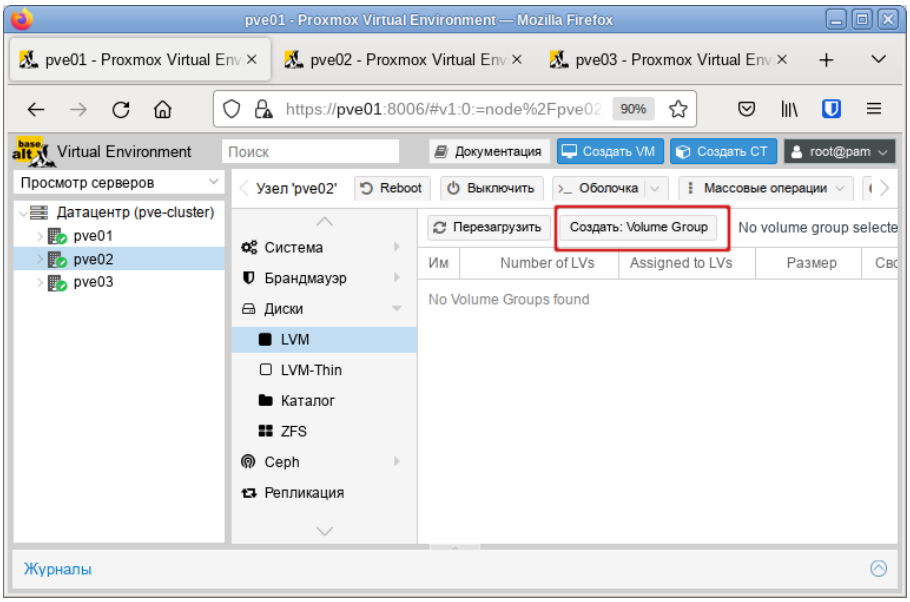


Рис. 78

Создание группы томов

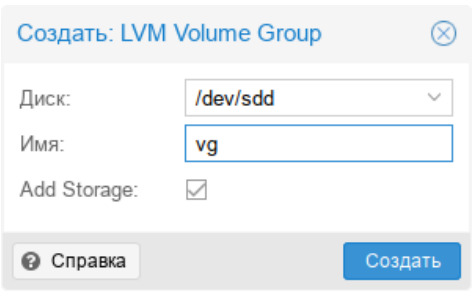


Рис. 79

Для того чтобы внести изменения в настройки LVM хранилища следует выбрать «Дата-центр» → «Хранилище» («Datacenter» → «Storage»), затем нужное хранилище и нажать кнопку «Редактировать». В открывшемся окне (*Рис. 80*) можно изменить тип содержимого контейнера, включить/отключить хранилище.

Редактирование LVM хранилища

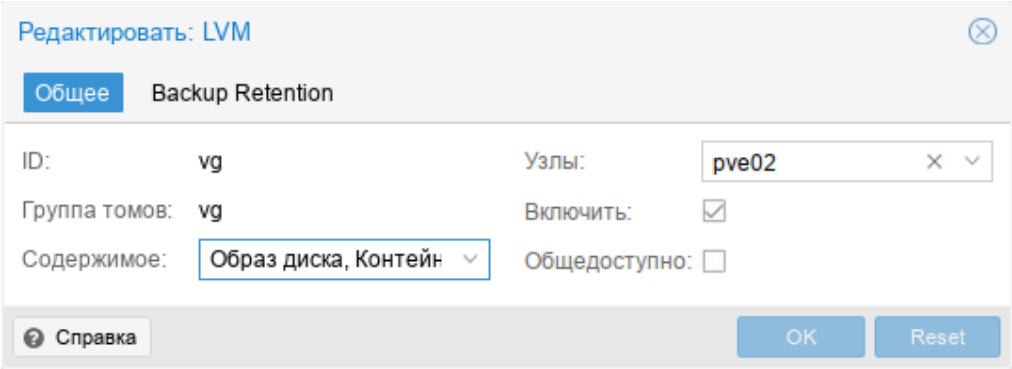


Рис. 80

4.4.3.9.2 Создание хранилища LVM в командной строке

Пример создания LVM хранилища на пустом диске /dev/sdd:

1) создать физический том (PV):

```
# pvcreate /dev/sdd
  Physical volume "/dev/sdd" successfully created.
```

2) создать группу томов (VG) с именем vg:

```
# vgcreate vg /dev/sdd
  Volume group "vg" successfully created
```

3) создать логические тома (LV):

```
# lvcreate -n lv01 -L 10G vg
  Logical volume "lv01" created.
# lvcreate -n lv02 -L 5G vg
  Logical volume "lv02" created.
```

4) показать информацию о физических томах:

```
# pvs
PV         VG         Fmt  Attr PSize   PFree
  /dev/sdd   vg         lvm2 a--  <18,00g <3,00g
```

5) показать информацию о группах томов:

```
# vgs
  VG         #PV #LV #SN Attr   VSize   VFree
  vg           1   2   0 wz--n- <18,00g <3,00g
```

6) показать информацию о логических томах:

```
# lvs
  LV        VG        Attr       LSize  Pool Origin Data%  Meta%  Move
Log Cpy%Sync Convert
  lv01      vg        -wi-a----- 10,00g
  lv02      vg        -wi-a-----  5,00g
```

7) получить список доступных PVE групп томов:

```
# pvesm lvmscan
vg
```

8) создать LVM хранилище с именем myspace:

```
# pvesm add lvm myspace --vgname vg --nodes pve03
```

4.4.3.10 LVM-thin

LVM-thin (thin provision) – это возможность использовать какое-либо внешнее блочное устройство в режиме только для чтения как основу для создания новых логических томов LVM. Такие разделы при создании уже будут выглядеть так, будто они заполнены данными исходного блочного устройства. Операции с томами изменяются налету таким образом, что чтение данных выполняется с исходного блочного устройства (или с тома, если данные уже отличаются), а запись – на том.

Такая возможность может быть полезна, например, при создании множества однотипных ВМ или для решения других аналогичных задач, т.е. задач, где нужно получить несколько изменяемых копий одних и тех же исходных данных.

Данное хранилище поддерживает все общие свойства хранилищ, кроме того, для настройки LVM-thin используются следующие свойства:

- vgname – имя группы томов LVM (должно указывать на существующую группу томов);
- thinpool – название тонкого пула LVM.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
lvmthin: vmstore
        thinpool vmstore
        vgname vmstore
        content rootdir,images
        nodes pve03
```

Возможные типы содержимого: `rootdir` (данные контейнера), `images` (образ виртуального диска в формате raw).

LVM thin является блочным хранилищем, но полностью поддерживает моментальные снимки и клоны. Новые тома автоматически инициализируются с нуля.

Тонкие пулы LVM не могут совместно использоваться несколькими узлами, поэтому их можно использовать только в качестве локального хранилища.

4.4.3.10.1 Создание локального хранилища LVM-Thin в веб-интерфейсе

Примечание. Для создания локального LVM-Thin хранилища в веб-интерфейсе необходимо чтобы в системе имелся хотя бы один пустой жесткий диск.

Для создания локального LVM-Thin хранилища в веб-интерфейсе, следует выбрать узел, на котором будет создано хранилище, в разделе «Диски» («Disks») выбрать пункт «LVM-Thin» и нажать кнопку «Создать: Thinpool» (*Рис. 81*). В открывшемся окне (*Рис. 82*) следует выбрать диск, задать имя группы томов, отметить пункт «Add Storage» (если этот пункт не отмечен будет создана только группа томов) и нажать кнопку «Создать».

Пункт «LVM-Thin» в разделе «Диски»

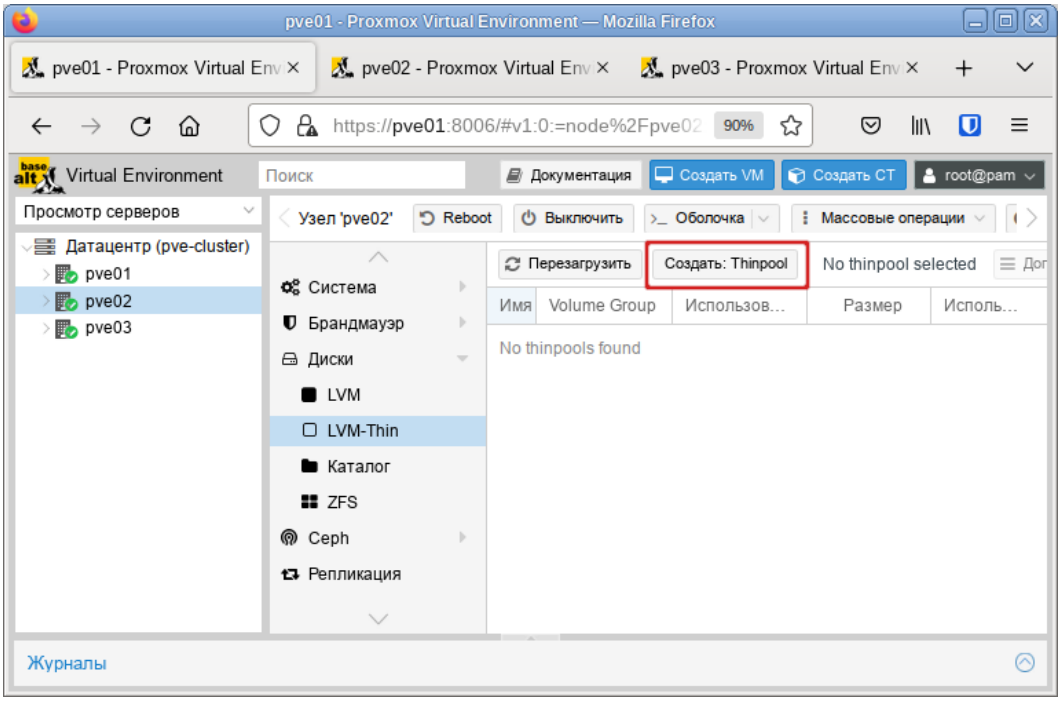


Рис. 81

Создание LVM-Thin хранилища

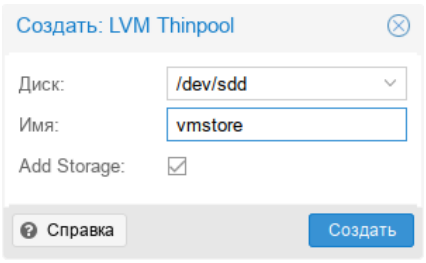


Рис. 82

Для того чтобы внести изменения в настройки LVM-Thin хранилища следует выбрать «Датацентр» → «Хранилище» («Datacenter» → «Storage»), затем нужное хранилище и нажать кнопку

«Редактировать». В открывшемся окне (*Рис. 83*) можно изменить тип содержимого контейнера, включить/отключить хранилище.

Редактирование LVM-Thin хранилища

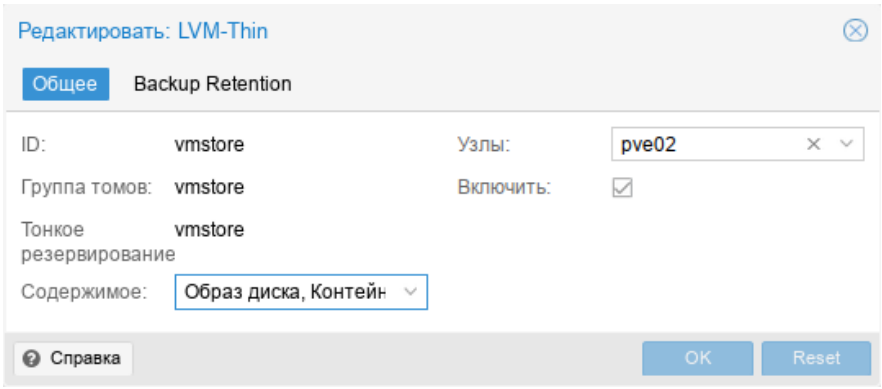


Рис. 83

4.4.3.10.2 Создание хранилища LVM-Thin в командной строке

Для создания и управления пулами LVM-Thin можно использовать инструменты командной строки.

Пул LVM-Thin должен быть создан поверх группы томов.

Команда создания нового тонкого пула LVM (размер 80 ГБ) с именем vmstore (предполагается, что группа томов LVM с именем vg уже существует):

```
# lvcreate -L 80G -T -n vmstore vg
```

Получить список доступных LVM-thin пулов в группе томов vg:

```
# pvesm lvmthinscan vg
vmstore
```

Команда создания LVM-Thin хранилища с именем vmstore на узле pve03:

```
# pvesm add lvmthin vmstore --thinpool vmstore --vgname vg --nodes pve03
```

4.4.3.11 iSCSI

iSCSI (Internet Small Computer System Interface) – широко применяемая технология, используемая для подключения к серверам хранения.

Данное хранилище поддерживает все общие свойства хранилищ, и дополнительно используются следующие свойства:

– portal – IP-адрес или DNS-имя сервера iSCSI;
– target – iSCSI target.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
iscsi: test1-iSCSI
        portal 192.168.0.105
```

```
target iqn.2021-7.local.omv:test
content images
```

Возможные типы содержимого: `images` (образ виртуального диска в формате raw).

iSCSI является типом хранилища блочного уровня и не предоставляет интерфейса управления. Поэтому обычно лучше экспортировать одно большое LUN и установить LVM поверх этого LUN.

Примечание. Для работы с устройством, подключенным по интерфейсу iSCSI, на всех узлах необходимо выполнить команду (должен быть установлен пакет open-iscsi):

```
# systemctl enable --now iscsid
```

На *Рис. 84* показано добавление адресата iSCSI с именем test1-iSCSI, который настроен на удаленном узле 192.168.0.105. Параметр «Использовать LUN напрямую» («Use LUNs directly») – разрешение/запрет прямого применения LUN (параметр должен быть установлен на запрет, разрешение может привести к потере данных).

*Добавление хранилища «*iSCSI*»*

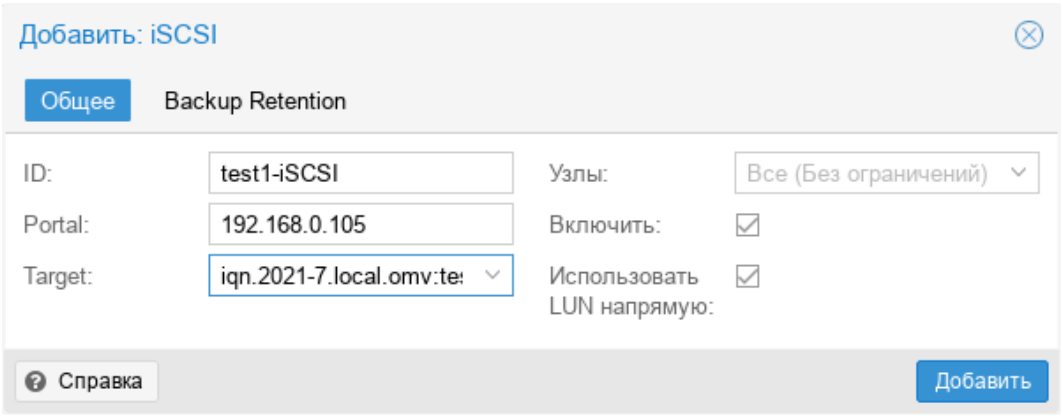


Рис. 84

Посмотреть доступные для подключения iSCSI target-ы:

```
# pvesm scan iscsi <HOST[:PORT]>
```

4.4.3.12 Ceph RBD

Хранилище RBD (Rados Block Device) предоставляется распределенной системой хранения Ceph. По своей архитектуре Ceph является распределенной системой хранения. Хранилище RBD может содержать только форматы образов `.raw`.

Данное хранилище поддерживает все общие свойства хранилищ, и дополнительно используются следующие свойства:

- monhost – список IP-адресов демона монитора (только если Ceph не работает на кластере PVE);
- pool – название пула Ceph (rbd);

– username – идентификатор пользователя Ceph (только если Ceph не работает на кластере PVE);
– subdir – подкаталог CephFS для монтирования (по умолчанию /);
– fuse – доступ к CephFS через FUSE (по умолчанию 0).

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
rbd: new
     content images
     krbd 0
     monhost 192.168.0.105
     pool rbd
     username admin
```

Возможные типы содержимого: `rootdir` (данные контейнера), `images` (образ виртуального диска в формате raw).

На *Рис. 85* показано добавление хранилища RBD.

*Добавление хранилища «*RBD*»*

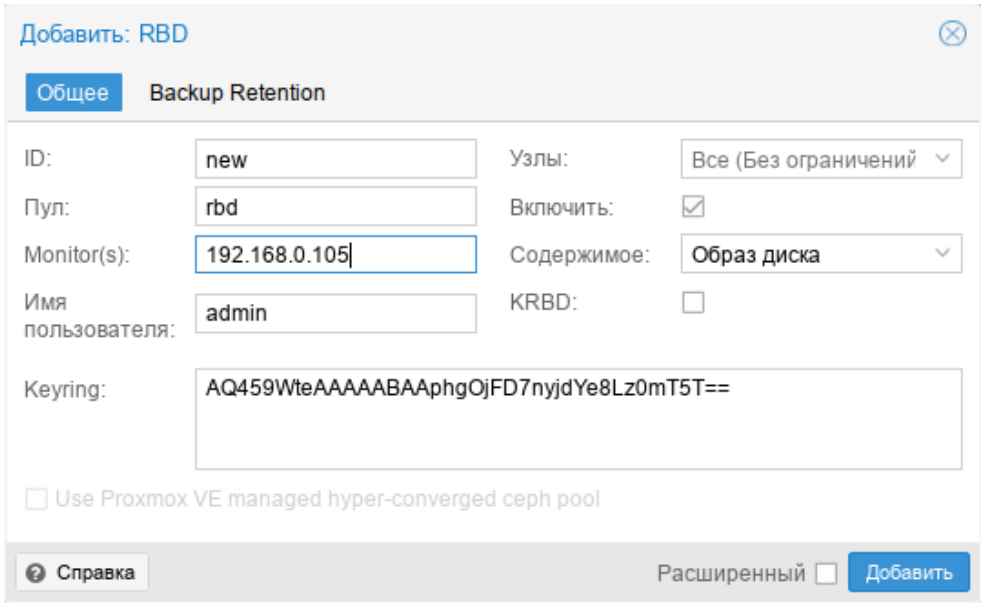


Рис. 85

Если используется аутентификация cephx, которая включена по умолчанию, необходимо предоставить связку ключей из внешнего кластера Ceph.

При настройке хранилища в командной строке, предварительно следует сделать доступным файл, содержащий связку ключей. Один из способов – скопировать файл из внешнего кластера Ceph непосредственно на один из узлов PVE. Например, скопировать файл в каталог /root узла:

```
# scp <external cephserver>:/etc/ceph/ceph.client.admin.keyring
/root/rbd.keyring
```

Команда настройки внешнего хранилища RBD:

```
# pvesm add rbd <name> --monhost "10.1.1.20 10.1.1.21 10.1.1.22" --
content images --keyring /root/rbd.keyring
```

При настройке внешнего хранилища RBD в графическом интерфейсе, связку ключей можно указать в поле «Keyring».

Связка ключей будет храниться в файле /etc/pve/priv/ceph/<STORAGE_ID>.keyring.

4.4.3.13 CephFS

CephFS реализует POSIX-совместимую файловую систему, использующую кластер хранения Ceph для хранения своих данных. Поскольку CephFS основывается на Ceph, он разделяет большинство свойств, включая избыточность, масштабируемость, самовосстановление и высокую доступность.

Данное хранилище поддерживает все общие свойства хранилищ, и дополнительно используются следующие свойства:

- monhost – список IP-адресов демона монитора (только если Ceph не работает на кластере PVE);
- path – локальная точка монтирования (по умолчанию используется `/mnt/pve/<STORAGE_ID>/`);
- username – идентификатор пользователя (только если Ceph не работает на кластере PVE);
- subdir – подкаталог CephFS для монтирования (по умолчанию /);
- fuse – доступ к CephFS через FUSE (по умолчанию 0).

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
cephfs: cephfs-external
     content backup,images
     monhost 192.168.0.105
     path /mnt/pve/cephfs-external
     username admin
```

Возможные типы содержимого: vztmpl (шаблон контейнера), iso (ISO-образ), backup (резервная копия), snippets (сниппеты).

На *Рис. 86* показано добавление хранилища CephFS.

Примечание. Получить список доступных cephfs, для указания в поле «FS Name», можно с помощью команды:

```
# ceph fs ls
```

Если используется аутентификация cephx, которая включена по умолчанию, необходимо указать ключ из внешнего кластера Ceph.

Добавление хранилища «CephFS»

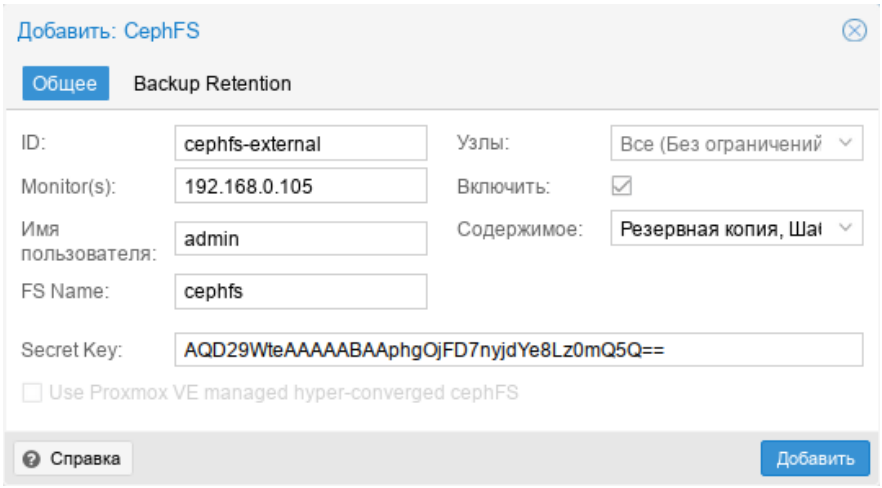


Рис. 86

При настройке хранилища в командной строке, предварительно следует сделать файл с ключом доступным. Один из способов – скопировать файл из внешнего кластера Ceph непосредственно на один из узлов PVE. Например, скопировать файл в каталог /root узла:

```
# scp <external cephserver>:/etc/ceph/cephfs.secret
/root/cephfs.secret
```

Команда настройки внешнего хранилища CephFS:

```
# pvesm add cephfs <name> --monhost "10.1.1.20 10.1.1.21 10.1.1.22" --
content backup --keyring /root/cephfs.secret
```

При настройке внешнего хранилища CephFS в графическом интерфейсе, связку ключей можно указать в поле «Secret Key».

Связка ключей будет храниться в файле /etc/pve/priv/ceph/<STORAGE_ID>.secret.

Ключ можно получить из кластера Ceph (как администратор Ceph), выполнив команду:

```
# ceph auth get-key client.userid > cephfs.secret
```

4.4.3.14 Proxmox Backup

Proxmox Backup – позволяет напрямую интегрировать сервер резервного копирования Proxmox в PVE.

Серверная часть поддерживает все общие свойства хранилищ, кроме флага «общее» («shared»), который всегда установлен. Кроме того, для Proxmox Backup доступны следующие специальные свойства:

– server – IP-адрес или DNS-имя сервера резервного копирования;

– username – имя пользователя на сервере резервного копирования (например, root@pam, backup_u@pbs);

- password – пароль пользователя. Значение будет сохранено в файле `/etc/pve/priv/storage/<STORAGE-ID>.pw`, доступном только суперпользователю;
- datastore– идентификатор хранилища на сервере резервного копирования;
- fingerprint – отпечаток TLS-сертификата API Proxmox Backup Server. Требуется, если сервер резервного копирования использует самоподписанный сертификат. Отпечаток можно получить в веб-интерфейсе сервера резервного копирования или с помощью команды `proxmox-backup-manager cert info`;
- encryption-key – ключ для шифрования резервной копии. Ключ будет сохранен в файле `/etc/pve/priv/storage/<STORAGE-ID>.enc`, доступном только суперпользователю;
- master-pubkey – открытый ключ RSA, используемый для шифрования резервного ключа шифрования в рамках задачи резервного копирования. Зашифрованная копия будет добавлена к резервной копии и сохранена на сервере резервного копирования для целей восстановления.

Пример файла конфигурации (`/etc/pve/storage.cfg`):

```
pbs: pbs_backup
        datastore store2
        server 192.168.0.123
        content backup
        fingerprint 42:5d:29:20:…:d1:be:bc:c0:c0:a9:9b:b1:a8:1b
        prune-backups keep-all=1
        username root@pam
```

На *Рис. 87* показано добавление хранилища Proxmox Backup с именем pbs_backup с удаленного сервера 192.168.0.123.

Добавление хранилища «Proxmox Backup»

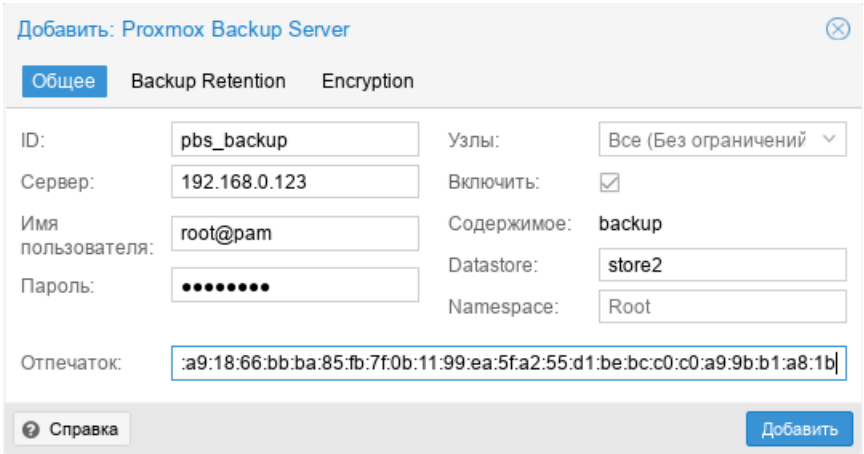


Рис. 87

Добавление хранилища Proxmox Backup в командной строке:

```
# pvesm add pbs pbs_backup --server 192.168.0.123 --datastore store2 -
-username root@pam --fingerprint 42:5d:29:…:c0:a9:b1:a8:1b --password
```

4.5 Управление ISO-образами и шаблонами LXC

Для загрузки ISO-образов и шаблонов LXC в хранилище PVE следует выполнить следующие шаги:

1) выбрать хранилище;

2) перейти на вкладку «ISO Images» для загрузки ISO-образов (Рис. 88) или на вкладку «CT Templates» для загрузки шаблонов LXC. Если загруженных файлов не существует, будет доступна только кнопка «Загрузить» («Upload»);

3) для загрузки образа (шаблона) с локального компьютера следует нажать кнопку «Загрузить» («Upload»). В открывшемся окне необходимо нажать кнопку «Выбрать файл…» («Select File...»), выбрать файл с ISO-образом и нажать кнопку «Загрузить» («Upload») (Рис. 89). Здесь же можно выбрать алгоритм и указать контрольную сумму. В этом случае после загрузки образа будет проверена его контрольная сумма;

4) для загрузки образа (шаблона) с сервера следует нажать кнопку «Download from URL». В открывшемся окне указать ссылку на образ (шаблон), нажать кнопку «Query URL», для того чтобы получить метаинорфмацию о файле, нажать кнопку «Загрузка» («Upload») для старта загрузки файла в хранилище (Рис. 90).

Локальное хранилище. Вкладка «ISO Images»

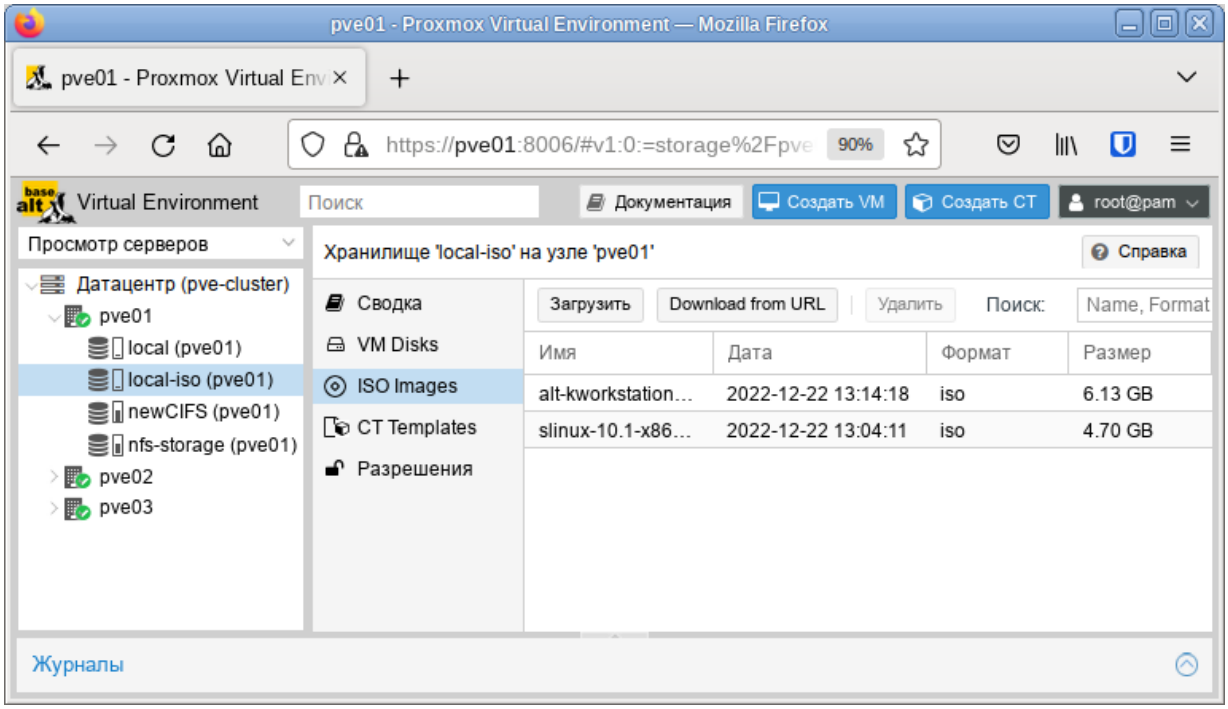


Рис. 88

Выбор образа

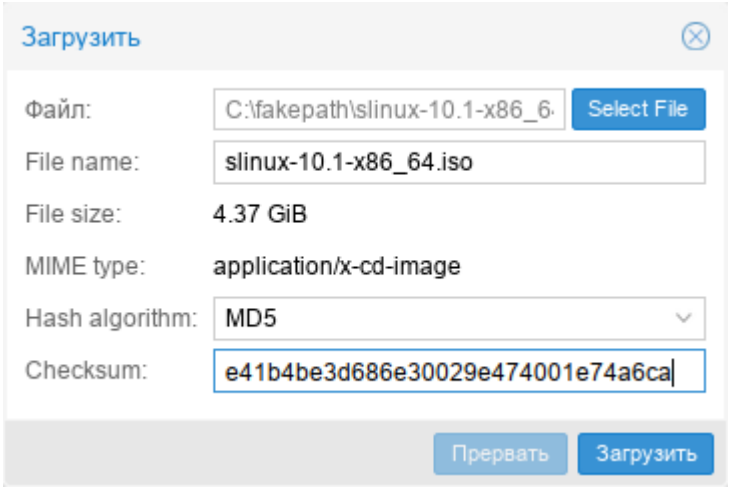


Рис. 89

Выбор образа для загрузки файла с сервера

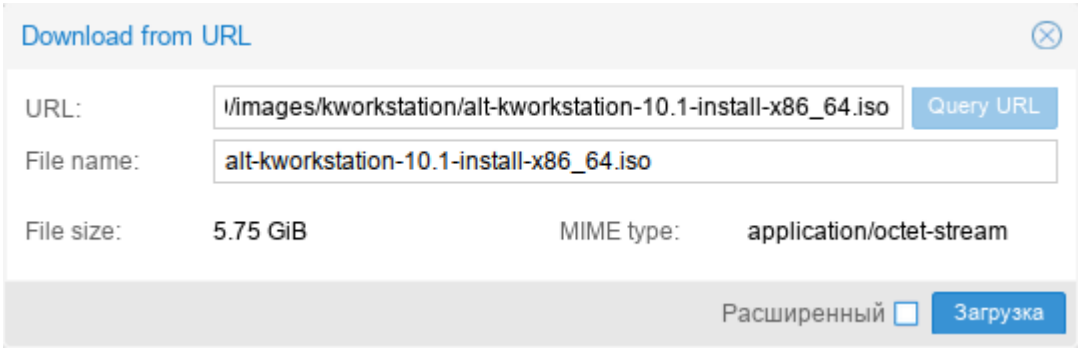


Рис. 90

Для удаления ISO-образа или шаблона LXC следует выбрать файл из списка в хранилище (Рис. 88) и нажать кнопку «Удалить» («Remove»).

ISO-образы и шаблоны LXC можно также загрузить, используя интерфейс командной строки.

Если используются только локальные хранилища, то ISO-образы и шаблоны необходимо загрузить на все узлы в кластере. Если есть общее хранилище, то можно хранить все образы в одном месте, таким образом, сохраняя пространство локальных хранилищ.

В таблице 6 показаны каталоги для локального хранилища. В таблице 7 показаны каталоги для всех других хранилищ.

Т а б л и ц а 6 – Каталоги локального хранилища

Каталог	Тип шаблона
`/var/lib/vz/template/iso`	ISO-образы
`/var/lib/vz/template/cache`	Шаблоны контейнеров LXC

Т а б л и ц а 7 – Каталоги общих хранилищ

Каталог	Тип шаблона
`/mnt/pve/<storage_name>/template/iso`	ISO-образы
`/mnt/pve/<storage_name>/template/cache`	Шаблоны контейнеров LXC

4.6 Виртуальные машины на базе KVM

4.6.1 Создание виртуальной машины на базе KVM

Прежде чем создать в интерфейсе PVE виртуальную машину (ВМ), необходимо определиться со следующими моментами:

- откуда будет загружен инсталлятор ОС, которая будет установлена внутрь ВМ;
- на каком физическом узле будет выполняться процесс гипервизора kvm;
- в каком хранилище данных будут располагаться образы дисков ВМ.

Все остальные параметры ВМ относятся к конфигурации виртуального компьютера и могут быть определены по ходу процесса создания ВМ (PVE пытается выбрать разумные значения по умолчанию для ВМ).

Чтобы установить ОС на ВМ, расположенную на этом узле, нужно обеспечить возможность загрузки инсталлятора на этой ВМ. Для этого необходимо загрузить ISO-образ инсталлятора в хранилище данных выбранного физического узла или общее хранилище. Это можно сделать через веб-интерфейс (Рис. 88).

Для создания ВМ необходимо нажать кнопку «Создать VM» («Create VM»), расположенную в правом верхнем углу веб-интерфейса PVE (Рис. 91).

Кнопка «Создать VM»

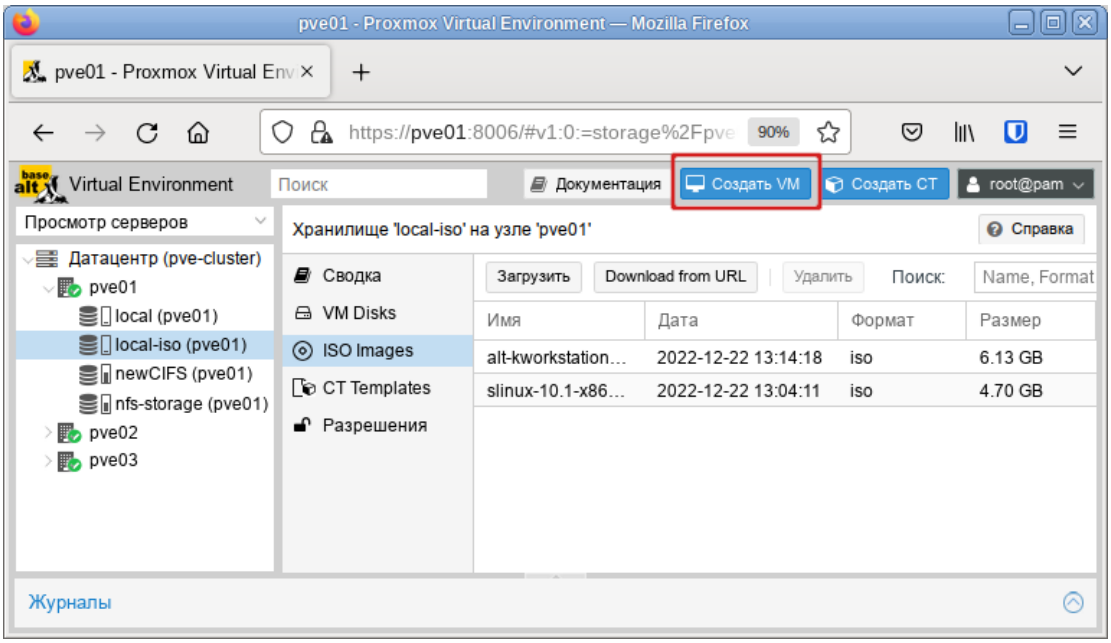


Рис. 91

Процесс создания ВМ оформлен в виде «мастера», привычного для пользователей систем управления ВМ.

На вкладке «Общее» («General») необходимо указать (Рис. 92):

– «Узел» («Node») – физический сервер, на котором будет работать ВМ;
– «VM ID» – идентификатор ВМ в численном выражении. Одно и то же значение идентификатора не может использоваться более чем для одной машины. Поле идентификатора ВМ заполняется автоматически инкрементально: первая созданная ВМ, по умолчанию будет иметь VM ID со значением 100, следующая 101 и так далее;
– «Имя» («Name») – текстовая строка названия ВМ;
– «Пул ресурсов» («Resource Pool») – логическая группа ВМ. Чтобы иметь возможность выбора, пул должен быть предварительно создан.

Вкладка «Общее»

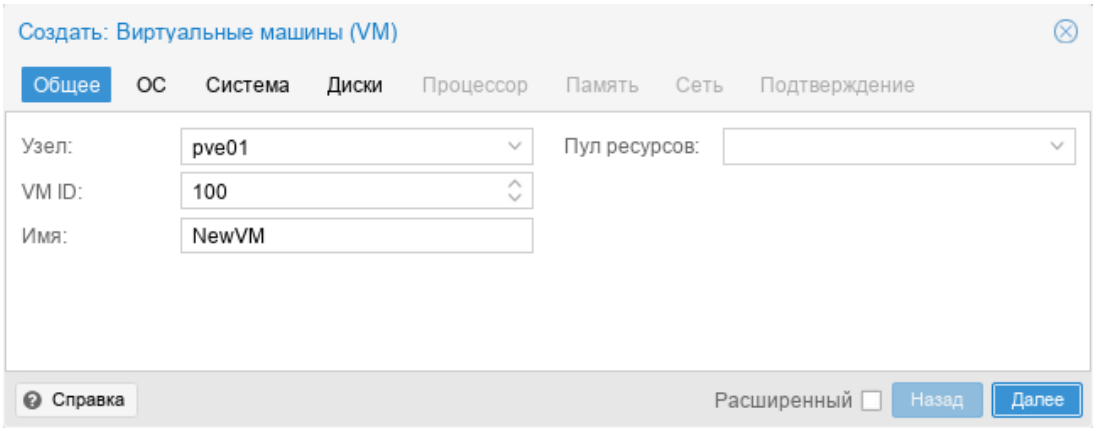


Рис. 92

Примечание. Настроить диапазон, из которого выбираются новые VM ID при создании ВМ или контейнера можно, выбрав на вкладке «Датацентр» → «Параметры» пункт «Next Free VMID Range» (Рис. 93). Установка нижнего значения («Lower») равным верхнему («Upper») полностью отключает автоподстановку VM ID.

На вкладке «ОС» необходимо указать источник установки ОС и тип ОС (*Рис. 95*).

Настройка диапазона VM ID

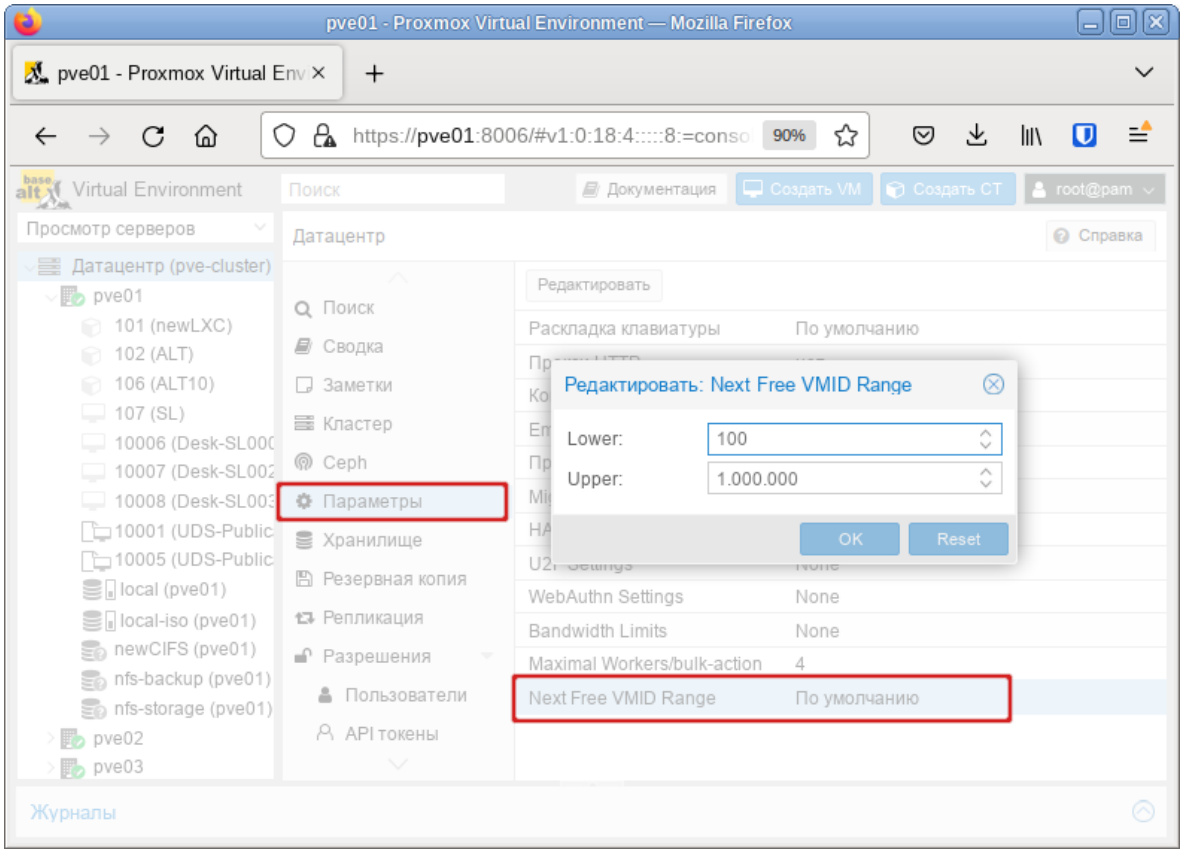


Рис. 93

Вкладка «ОС»

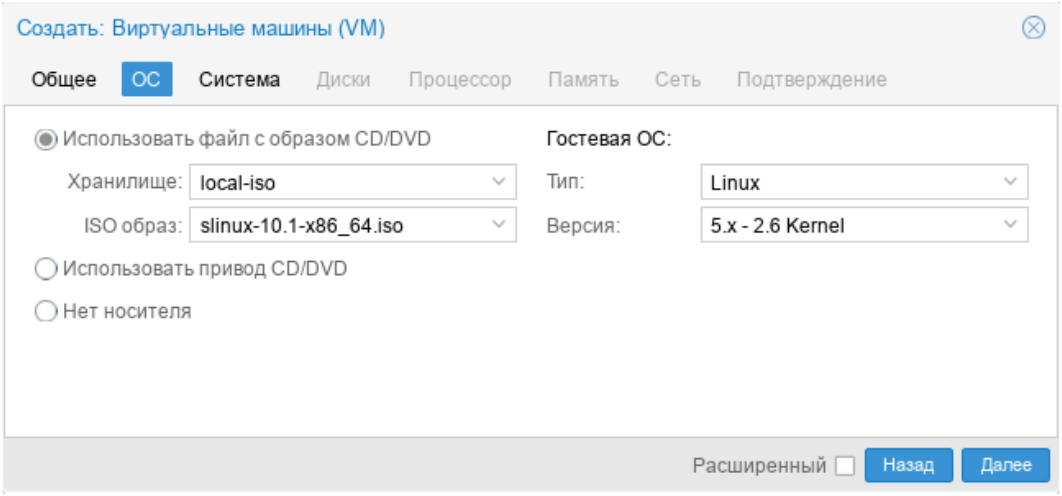


Рис. 94

В качестве источника установки ОС («OS») можно указать:

– «Использовать файл с образом CD/DVD» («Use CD/DVD disc image file») – использовать уже загруженный в хранилище ISO-образ (Рис. 95);

– «Использовать привод CD/DVD» («Use physical CD/DVD Drive») – использовать физический диск хоста PVE;

– «Нет носителя» («Do not use any media») – не использовать ISO-образ или физический носитель.

Выбор типа гостевой ОС при создании ВМ позволяет PVE оптимизировать некоторые параметры низкого уровня.

Выбор ISO-образа

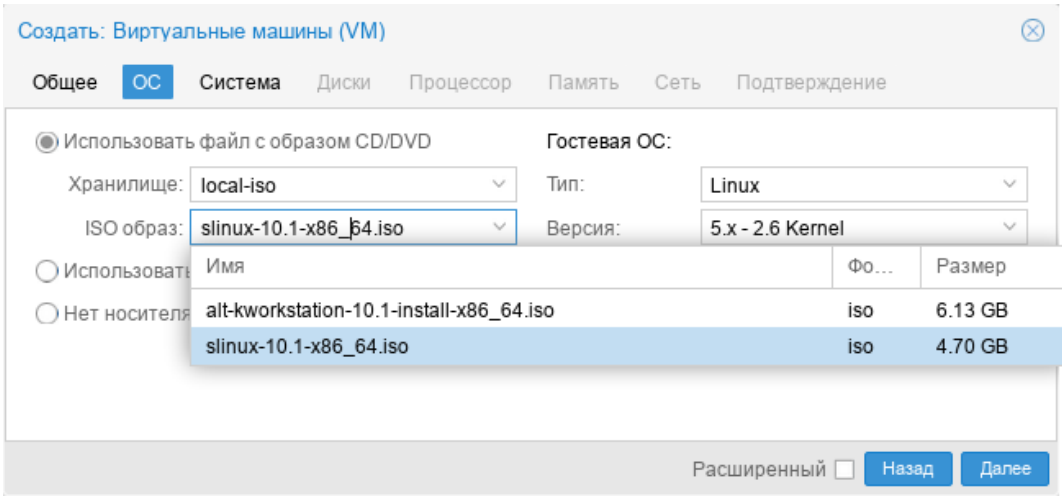


Рис. 95

На следующем этапе (вкладка «Система») можно выбрать видеокарту, контроллер SCSI, указать нужно ли использовать агент QEMU (Рис. 96).

Вкладка «Система»

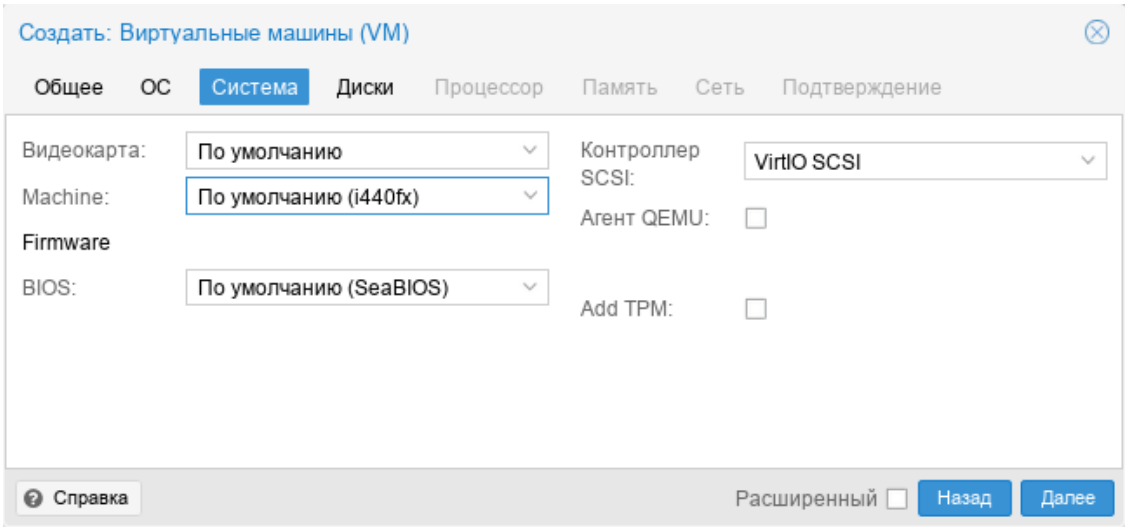


Рис. 96

Подробнее о выборе видеокарты см. «Настройки дисплея».

PVE позволяет загружать ВМ с разными прошивками (SeaBIOS и OVMF). Прошивку OVMF следует выбирать, если планируется использовать канал PCIe. При выборе прошивки OVMF (*Рис. 97*) для сохранения порядка загрузки, должен быть добавлен диск EFI (см. «BIOS и UEFI»).

Выбор прошивки OVMF

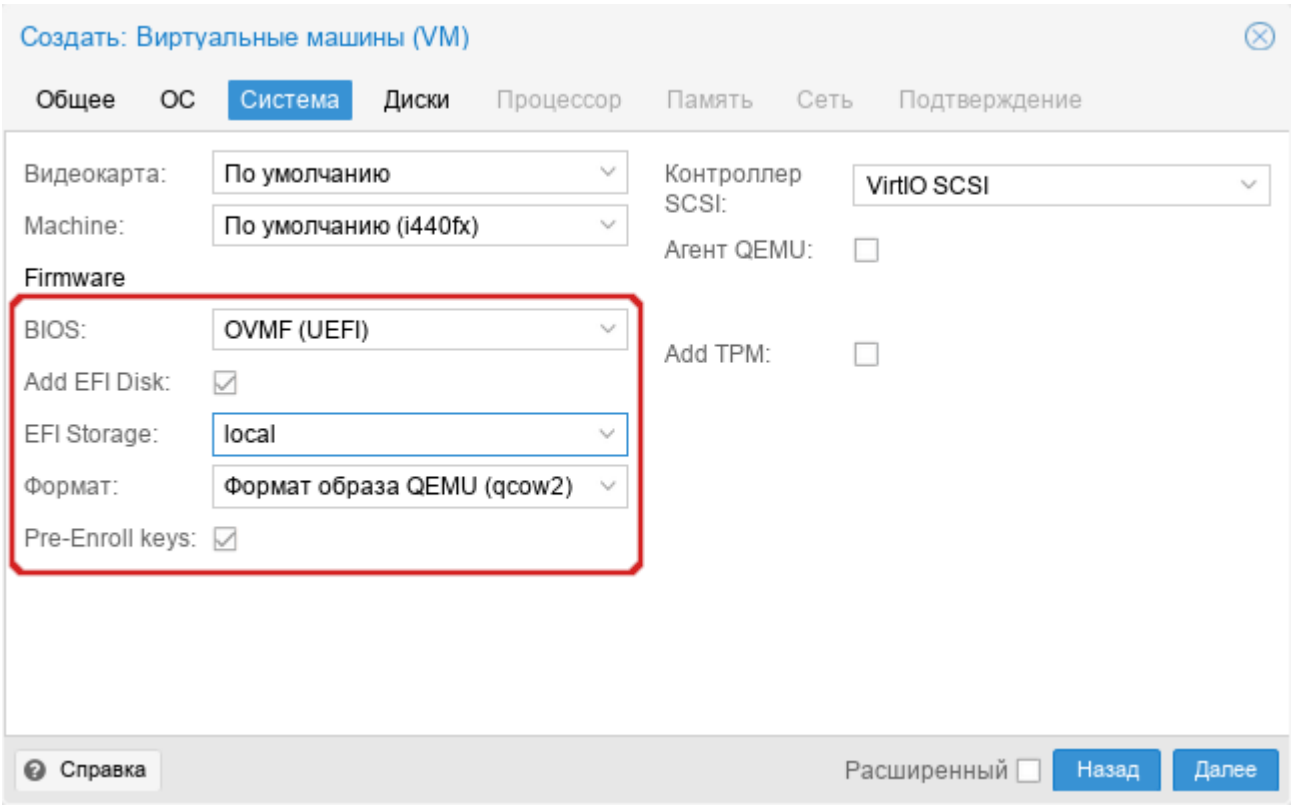


Рис. 97

Тип машины ВМ определяет аппаратную компоновку виртуальной материнской платы ВМ. Доступно два варианта набора микросхем: Intel 440FX (по умолчанию) и Q35 (предоставляет виртуальную шину PCIe).

Вкладка «Диски» («Disks») содержит следующие настройки (Рис. 98):

– «Шина/Устройство» («Bus/Device») – тип устройства виртуального диска. Допустимые значения: «IDE», «SATA», «VirtIO Block» и «SCSI» (по умолчанию). Можно также указать идентификатор устройства;
– «Хранилище» («Storage») – выбор хранилища для размещения виртуального диска (выбор хранилища определяет возможный формат образа диска);
– «Размер диска» («Disk size») (GiB) – размер виртуального диска в гигабайтах;
– «Формат» («Format») – выбирается формат образа виртуального диска. Доступные значения: «Несжатый образ диска (raw)», «Формат образа QEMU (qcow2)» и «Формат образа Vmware (vmdk)». Формат образа RAW является полностью выделяемым (thick-provisioned), т.е. выделяется сразу весь объем образа. QEMU и VMDK поддерживают динамичное выделение пространства (thin-provisioned), т.е. объем растет по мере сохранения данных на виртуальный диск;

– «Кэш» («Cache») – выбор метода кэширования виртуальной машины. По умолчанию выбирается работа без кэширования. Доступные значения: «Direct sync», «Write through», «Write back» и «Writeback (не безопасно)» и «Нет кэша»;
– «Отклонить» («Discard») – если эта опция активирована и если гостевая ОС поддерживает TRIM, то это позволит очищать неиспользуемое пространство образа виртуального диска и соответственно сжимать образ диска.

Вкладка «Жесткий диск»

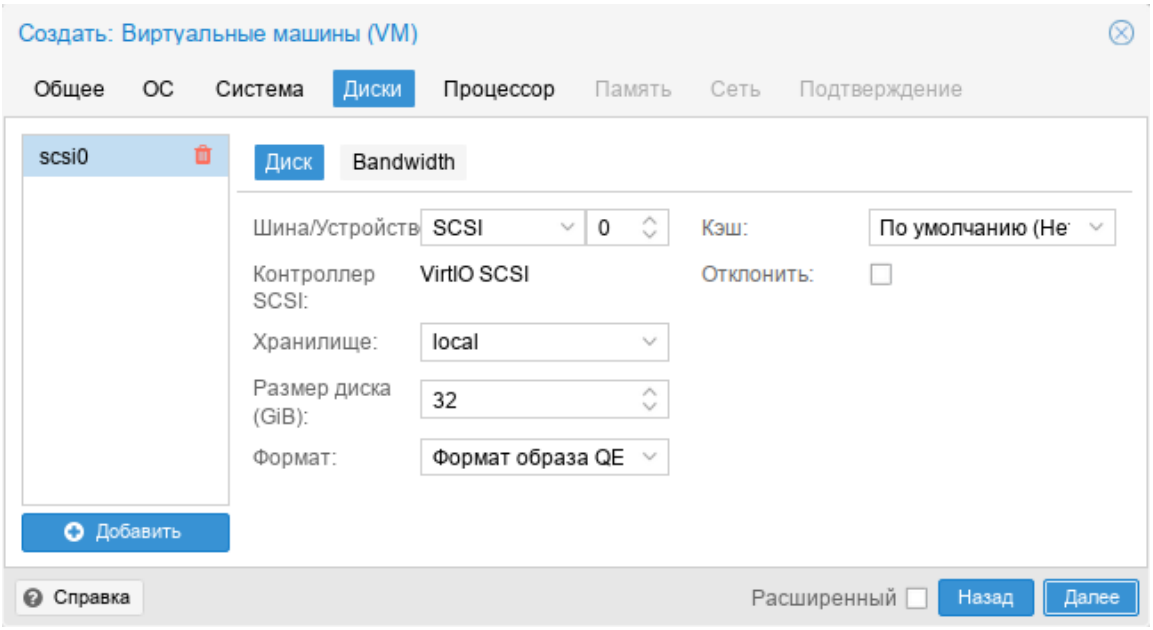


Рис. 98

В мастере создания ВМ можно добавить несколько дисков (*Рис. 99*) (кнопка «Добавить»). Максимально можно добавить: 31 диск SCSI, 16 – VirtIO, 6 – SATA, 4 – IDE.

Вкладка «Жесткий диск». Создание нескольких дисков

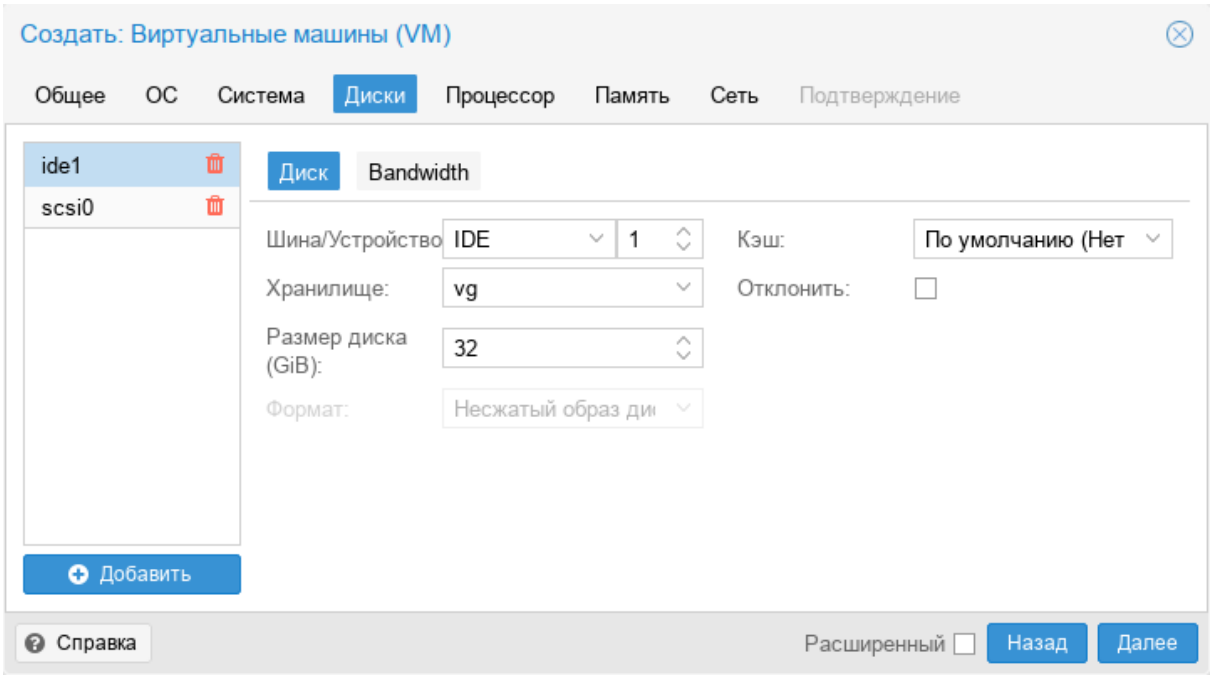


Рис. 99

В разделе «Bandwidth» (*Рис. 100*) можно задать максимальную скорость чтения/записи с диска (в мегабайтах в секунду или в операциях в секунду).

Скорость чтения/записи с диска

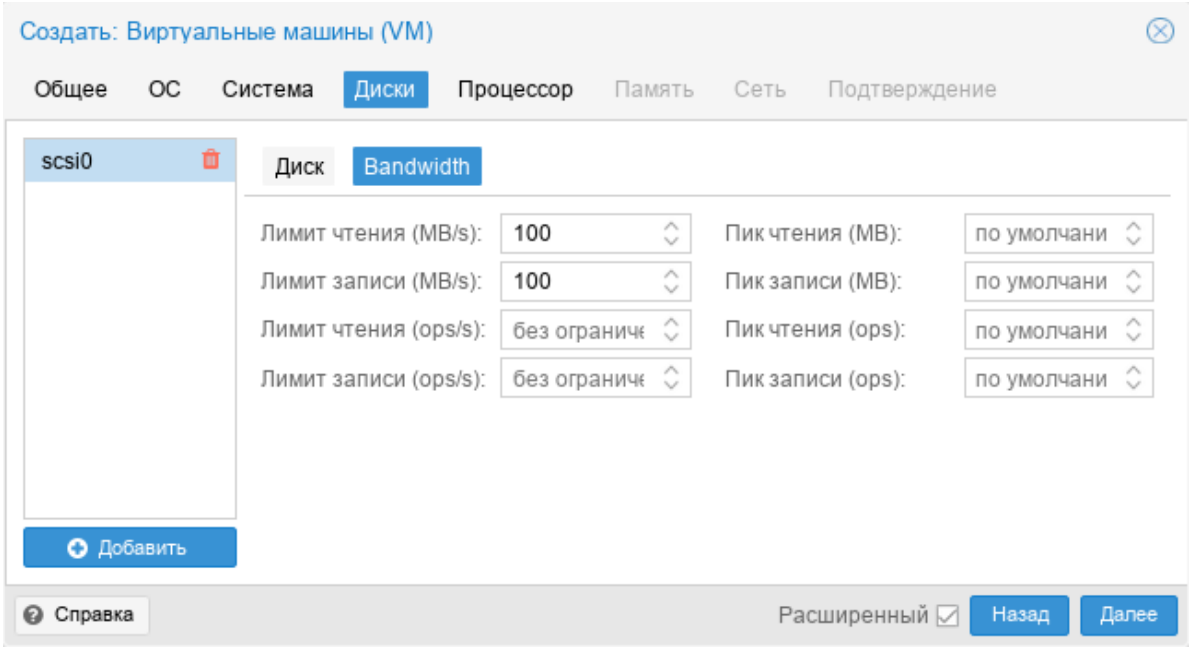


Рис. 100

Примечание. SCSI и VirtIO дискам может быть добавлен атрибут read-only (*Рис. 101*) (отметка «Только для чтения»).

Отметка «Только для чтения»

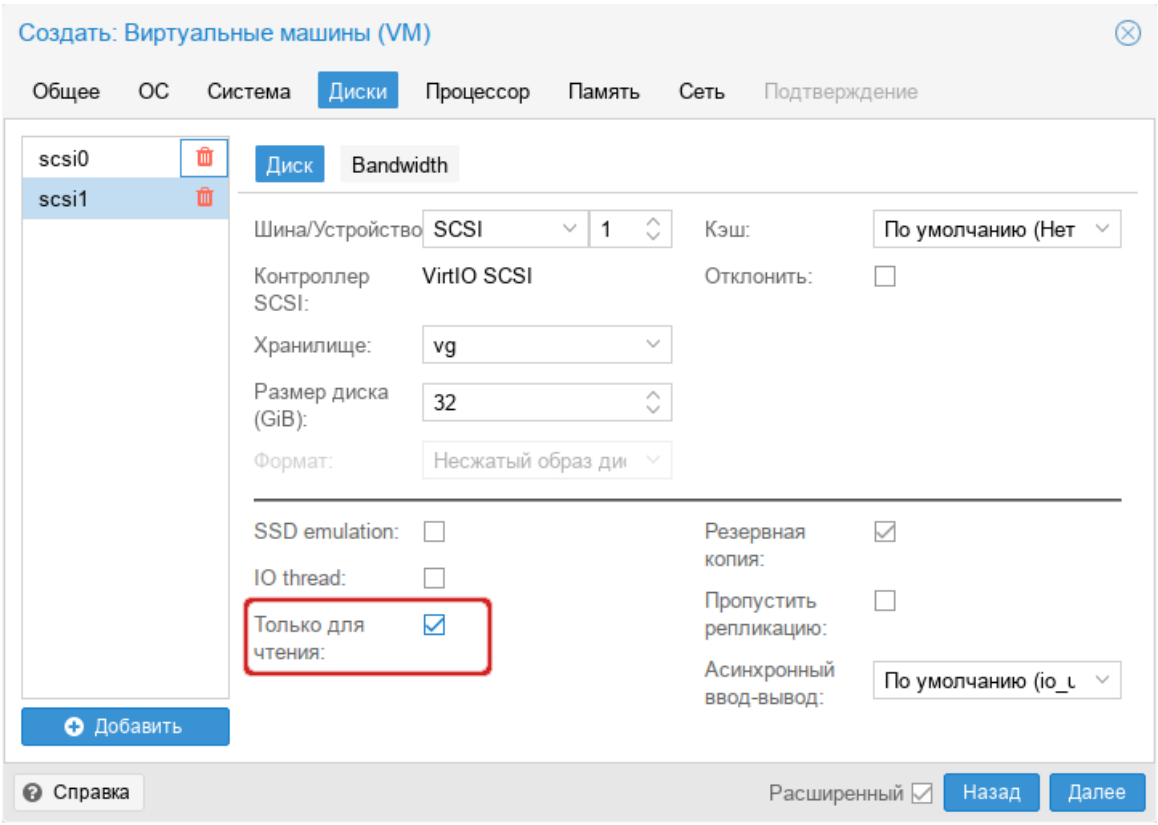


Рис. 101

На следующем этапе настраивается процессор (CPU) (Рис. 102):

– «Сокеты» («Sockets») – число сокетов ЦПУ для ВМ;

– «Ядра» («Cores») – число ядер для ВМ;

– «Тип» («Type») – тип процессора.

Вкладка «Процессор»

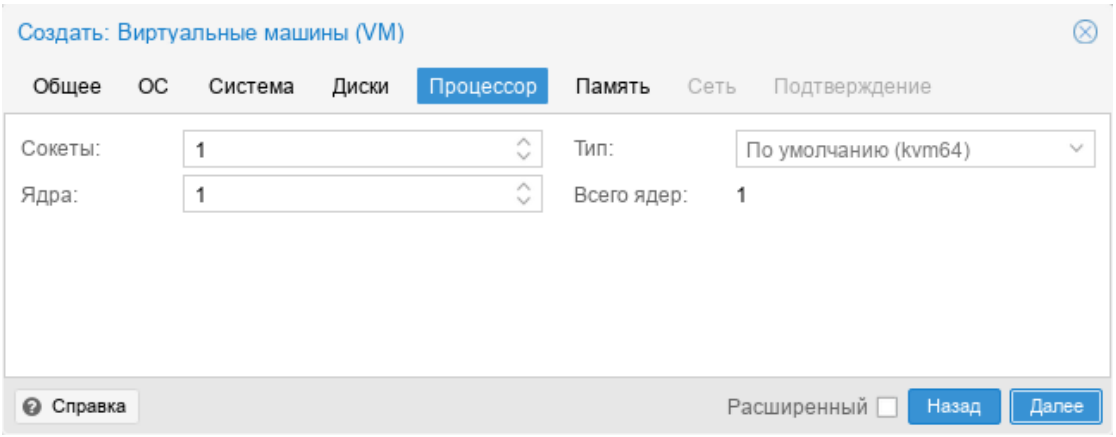


Рис. 102

На вкладке «Память» («Memory») (Рис. *103*) необходимо указать объем оперативной памяти выделяемой ВМ.

Вкладка «Память»

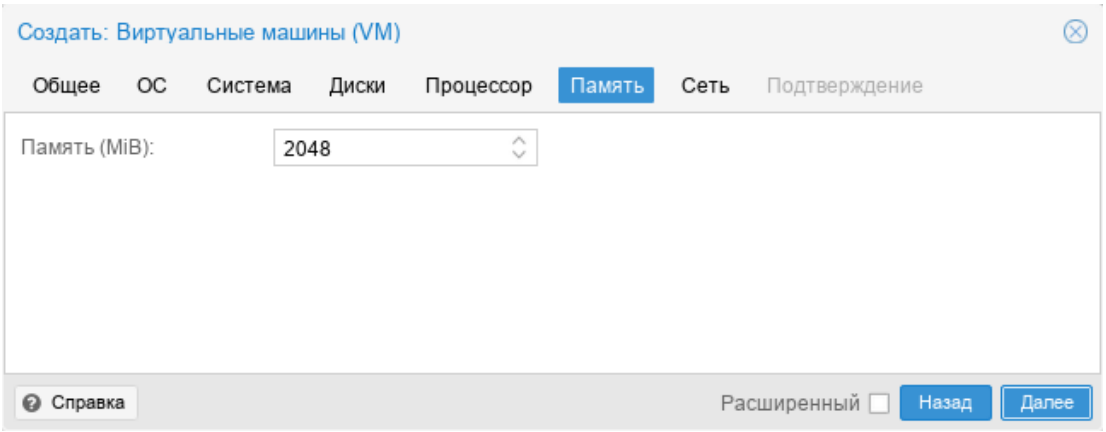


Рис. 103

Вкладка «Сеть» («Network») содержит следующие настройки (Рис. 104):

– «Нет сетевого устройства» («No network device») – выбор данного параметра пропускает шаг настройки сетевой среды;

– «Сетевой мост» («Bridge») – установка сетевого интерфейса в режиме моста. Это предпочтительный параметр для сетевой среды ВМ. В этом режиме возможно создание множества мостов с виртуальными сетями для создания изолированных сетей в одной и той же платформе, поскольку ВМ не имеют прямого доступа к реальной локальной сетевой среде;

– «Брандмауэр» («Firewall») – разрешает использование для ВМ встроенных межсетевых экранов;

– «Модель» («Model») – тип драйвера сетевого устройства. Для максимальной сетевой производительности ВМ следует выбрать пункт «VirtIO (паравиртуализированно)»;

– «Адрес MAC» («MAC address») – по умолчанию PVE автоматически создает уникальный MAC-адрес для сетевого интерфейса. Если есть такая необходимость, можно ввести пользовательский MAC-адрес вручную.

Вкладка «Сеть»

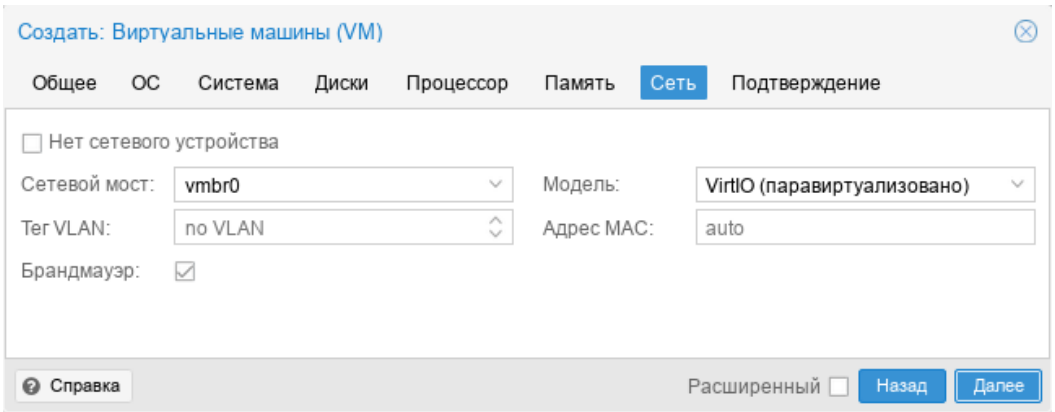


Рис. 104

Последняя вкладка «Подтверждение» («Confirm») отобразит все введенные или выбранные значения для ВМ (Рис. 105). Для создания ВМ следует нажать кнопку «Далее». Если необходимо внести изменения в параметры ВМ, можно перейти по вкладкам назад. Если отметить пункт «Start after created» ВМ будет запущена сразу после создания.

Вкладка «Подтверждение»

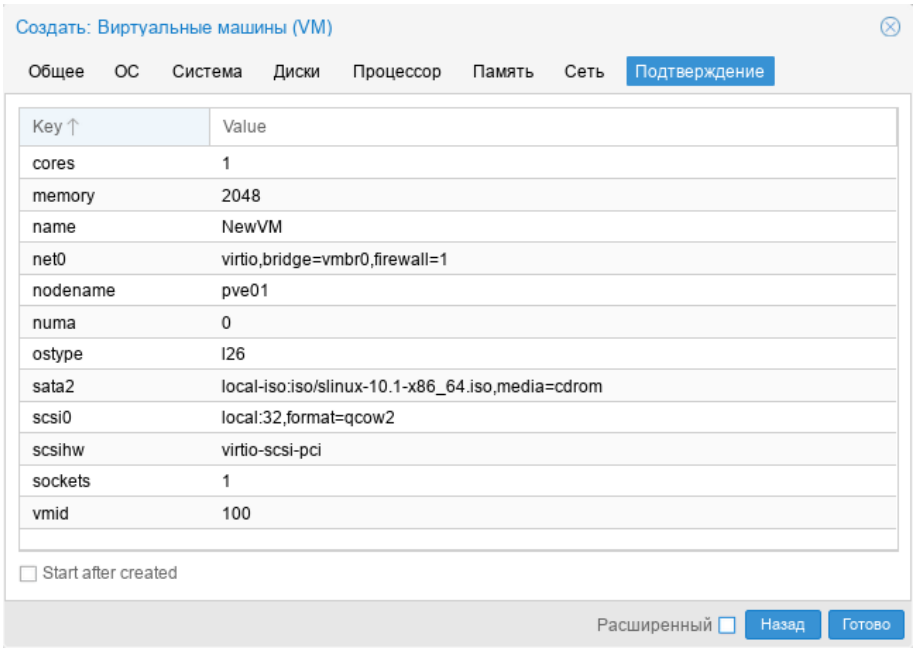


Рис. 105

4.6.2 Запуск и останов ВМ

4.6.2.1 Изменение состояния ВМ в веб-интерфейсе

Запустить ВМ можно, выбрав в контекстном меню ВМ пункт «Запуск» (*Рис. 106*), либо нажав на кнопку «Запуск» («Start») (*Рис. 107*).

Запущенная ВМ будет обозначена зеленой стрелкой на значке ВМ.

Контекстное меню ВМ

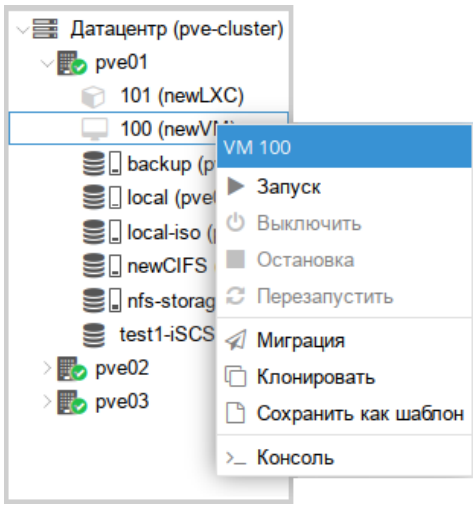


Рис. 106

Кнопки управления состоянием ВМ

Рис. 107

Запустить ВМ также можно, нажав кнопку «Start Now» в консоли гостевой машины (*Рис. 108*).

Кнопка «Start Now» в консоли ВМ

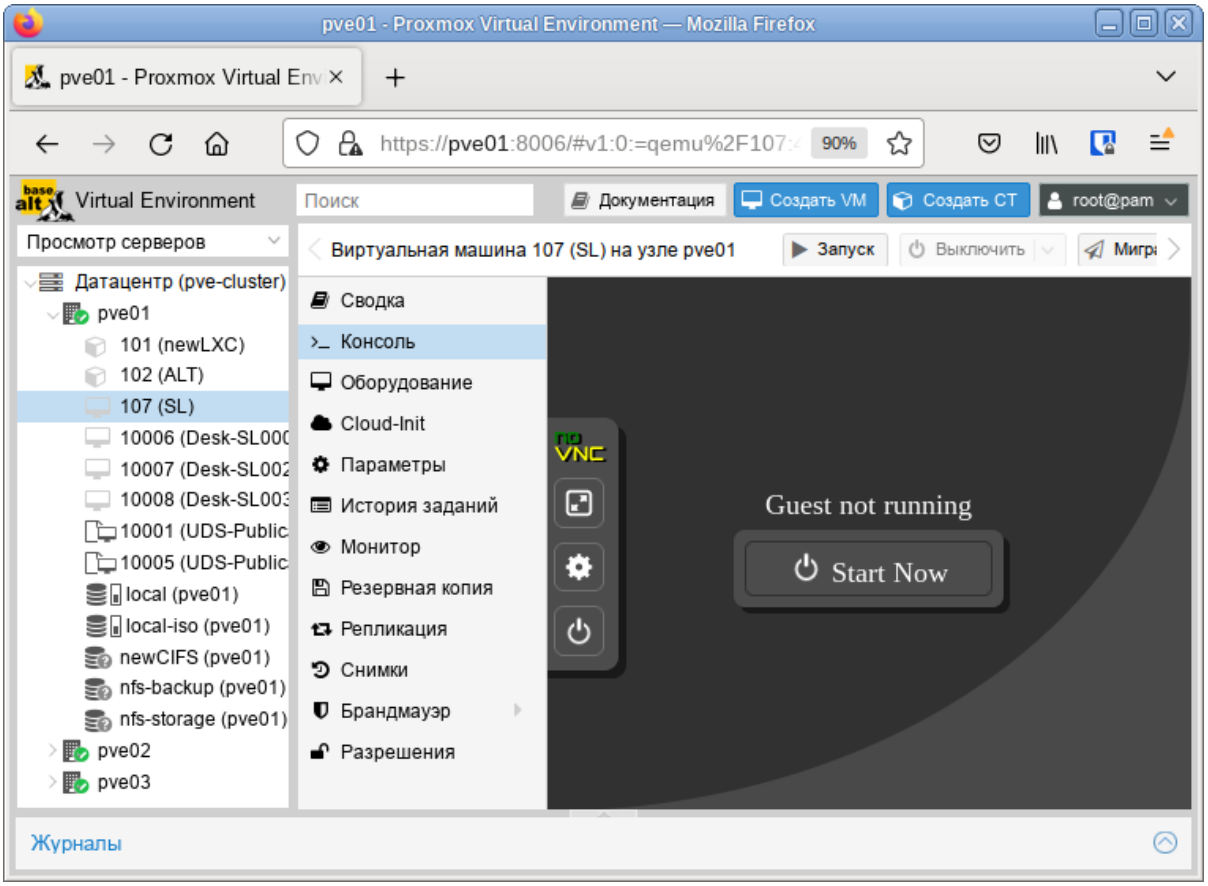


Рис. 108

Для запущенной ВМ доступны следующие действия (*Рис. 109*):

– «Пауза» («Pause») – перевод ВМ в спящий режим;
– «Hibernate» – перевод ВМ в ждущий режим;
– «Выключить» («Shutdown») – выключение ВМ;
– «Остановка» («Stop») – остановка ВМ, путем прерывания ее работы;
– «Перезапустить» («Reboot») – перезагрузка ВМ.

4.6.2.2 *Автоматический запуск ВМ*

Для того чтобы ВМ запускалась автоматически при загрузке хост-системы, необходимо отметить опцию «Запуск при загрузке» на вкладке «Параметры» ВМ в веб-интерфейсе или установить ее с помощью команды:

```
# qm set <vmid> -onboot 1
```

Контекстное меню запущенной ВМ

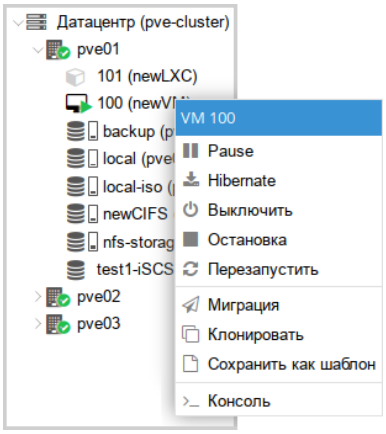


Рис. 109

Иногда необходимо точно настроить порядок загрузки ВМ, например, если одна из ВМ обеспечивает межсетевой экран или DHCP для других гостевых систем. Для настройки порядка запуска ВМ можно использовать следующие параметры (Рис. 110) (опция «Порядок запуска и выключения» на вкладке «Параметры» требуемой ВМ):

– «Порядок запуска и выключения» («Start/Shutdown order») – определяет приоритет порядка запуска. Для того чтобы ВМ запускалась первой необходимо установить этот параметр в значение 1 (для выключения используется обратный порядок: ВМ машина с порядком запуска 1 будет выключаться последней). Если несколько хостов имеют одинаковый порядок, определенный на хосте, они будут дополнительно упорядочены в порядке возрастания VMID;
– «Задержка запуска» («Startup delay») – определяет интервал (в секундах) между запуском этой ВМ и последующими запусками ВМ;
– «Задержка выключения» («Shutdown timeout») – определяет время в секундах, в течение которого PVE должен ожидать, пока ВМ не перейдет в автономный режим после команды выключения. Значение по умолчанию – 180, т.е. PVE выдаст запрос на завершение работы и подождет 180 секунд, пока машина перейдет в автономный режим. Если после истечения тайм-аута машина все еще находится в сети, она будет принудительно остановлена.

Настройка порядка запуска и выключения ВМ

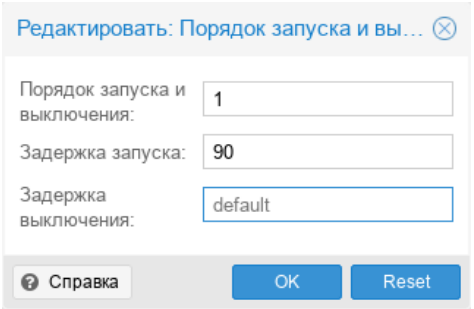


Рис. 110

Примечание. Виртуальные машины, управляемые стеком HA, не поддерживают опции запуска при загрузке и порядок загрузки. Запуск и остановку таких ВМ обеспечивает диспетчер HA.

ВМ без установленного параметра «Порядок запуска и выключения» всегда будут запускаться после тех, для которых этот параметр установлен. Кроме того, этот параметр может применяться только для ВМ, работающих на одном хосте, а не в масштабе кластера.

4.6.3 Управление ВМ с помощью qm

Если веб-интерфейс PVE недоступен, можно управлять ВМ в командной строке (используя сеанс SSH, из консоли noVNC, или зарегистрировавшись на физическом хосте).

`qm` – это инструмент для управления ВМ Qemu/KVM в PVE. Утилиту `qm` можно использовать для создания/удаления ВМ, для управления работой ВМ (запуск/остановка/приостановка/возобновление), для установки параметров в соответствующем конфигурационном файле, а также для создания виртуальных дисков.

Чтобы просмотреть доступные для управления ВМ команды можно выполнить следующую команду:

```
# qm help
```

Примеры использования утилиты `qm`:

- создать ВМ, используя ISO-файл, загруженный в локальное хранилище, с диском IDE 21 ГБ, в хранилище local-lvm:

```
# qm create 300 -ide0 local-lvm:21 -net0 e1000 -cdrom local:iso/alt-server-9.1-x86_64.iso
```

- запуск ВМ с VM ID 109:

```
# qm start 109
```

- отправить запрос на отключение, и дождаться остановки ВМ:

```
# qm shutdown 109 && qm wait 109
```

- войти в интерфейс монитора QEMU и вывести список доступных команд:

```
# qm monitor 109
qm> help
```

4.6.4 Доступ к ВМ

По умолчанию PVE предоставляет доступ к ВМ через noVNC и/или SPICE. Рекомендуется использовать их, когда это возможно.

Использование протокола SPICE позволяет задействовать множество возможностей, в том числе, проброс USB, смарт-карт, принтеров, звука, получить более тесную интеграцию с окном гостевой системы (бесшовную работу мыши, клавиатуры, динамическое переключение разреше-

ния экрана, общий с гостевой системой буфер обмена для операций копирования/вставки). Для возможности использования SPICE:

- на хосте, с которого происходит подключение, должен быть установлен клиент SPICE (например, пакет virt-viewer):
- для параметра «Дисплей» ВМ должно быть установленно значение VirtIO, SPICE (qxl) (см. «Настройки дисплея»).

При подключении к ВМ с использованием noVNC, консоль открывается во вкладке браузера (не нужно устанавливать клиентское ПО).

Для доступа к ВМ следует выбрать её в веб-интерфейсе, нажать кнопку «Консоль» («Console») и в выпадающем меню выбрать нужную консоль (*Рис. 111*).

Кнопка «Консоль»

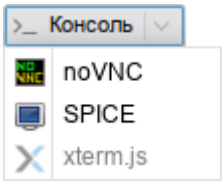


Рис. 111

Консоль noVNC также можно запустить, выбрав вкладку «Консоль» для ВМ (*Рис. 112*).

Консоль noVNC

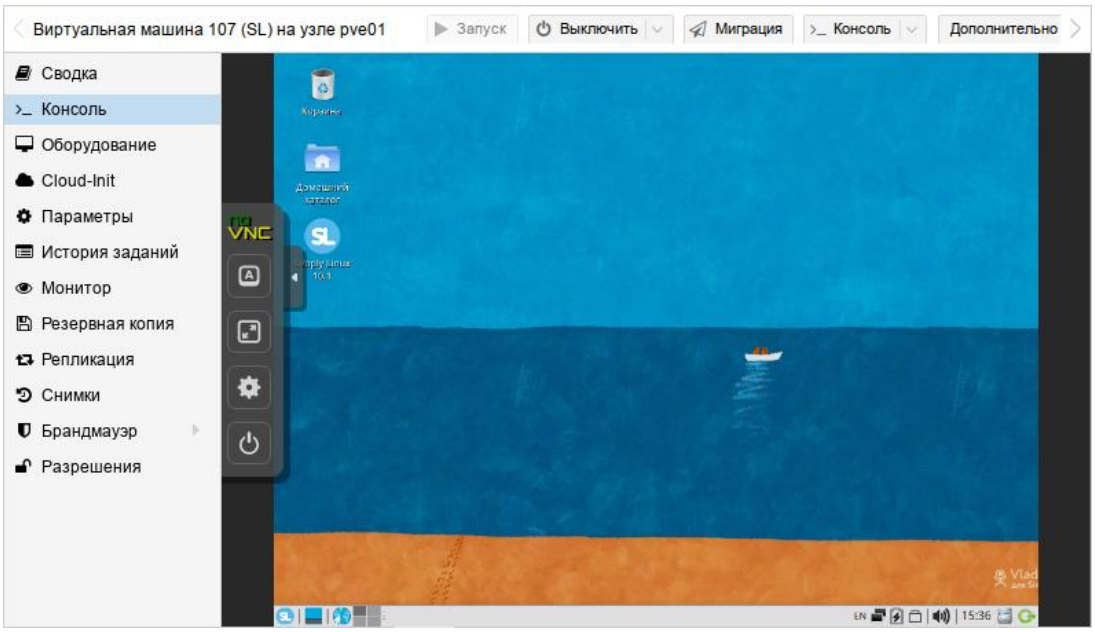


Рис. 112

Если нужен независимый от браузера доступ, можно также использовать внешний клиент VNC. Для этого в файл конфигурации ВМ `/etc/pve/qemu-server/<VMID>.conf` необходимо добавить строку с указанием номера дисплея VNC (в примере – 55):

```
args: -vnc 0.0.0.0:55
```

Или, чтобы включить защиту паролем:

```
args: -vnc 0.0.0.0:55,password=on
```

Если была включена защита паролем, необходимо установить пароль (после запуска ВМ). Пароль можно установить на вкладке «Монитор», выполнив команду:

```
set_password vnc newvnc -d vnc2
```

В данном примере, при подключении будет запрашиваться пароль: newvnc. Максимальная длина пароля VNC: 8 символов. После перезапуска ВМ указанную выше команду необходимо повторить, чтобы снова установить пароль.

Примечание. Номер дисплея VNC можно выбрать произвольно, но каждый номер должен встречаться только один раз. Служба VNC прослушивает порт 5900+номер_дисплея. Соединения noVNC используют номер дисплея 0 и последующие, поэтому во избежание конфликтов рекомендуется использовать более высокие номера.

Для подключения клиента VNC следует указать IP-адрес хоста с ВМ и порт (в приведенном выше примере – 5955).

4.6.5 Внесение изменений в ВМ

Вносить изменения в конфигурацию ВМ можно и после ее создания. Для того чтобы внести изменения в конфигурацию ВМ необходимо выбрать ВМ и перейти на вкладку «Оборудование» («Hardware»). На этой вкладке следует выбрать ресурс (Рис. 113) и нажать кнопку «Редактировать» («Edit») для выполнения изменений.

Оборудование ВМ

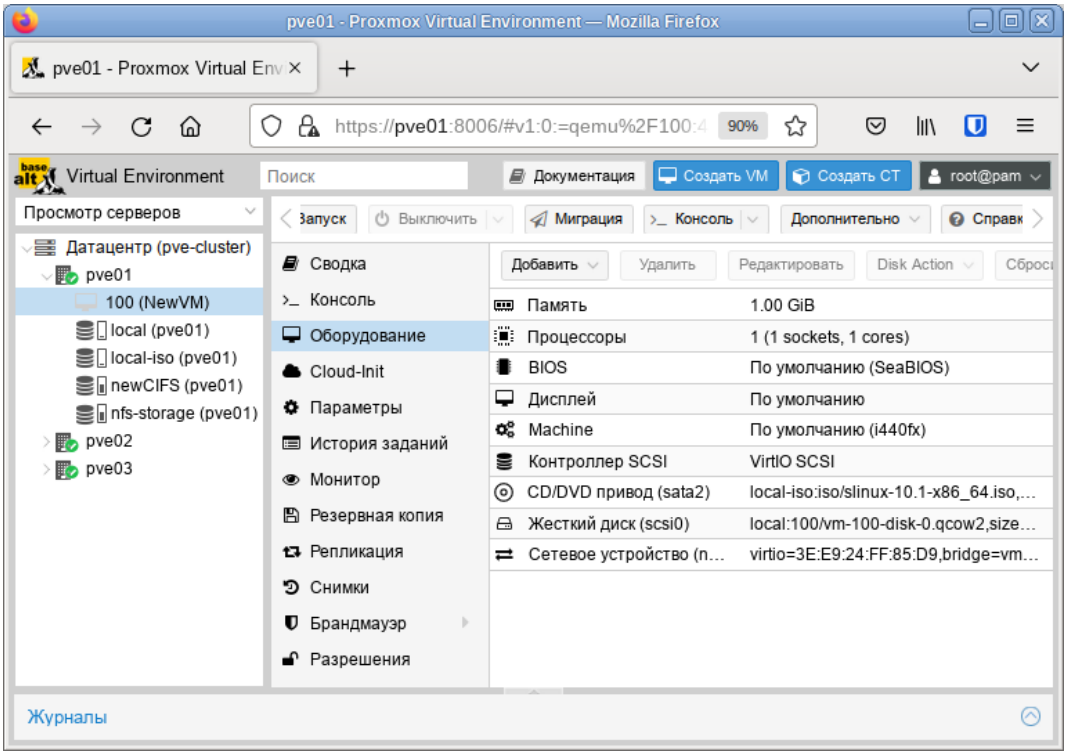


Рис. 113

Примечание. В случаях, когда изменение не может быть выполнено в горячем режиме, оно будет зарегистрировано как ожидающее изменение (*Рис. 114*) (выделяется цветом). Такие изменения будут применены только после перезагрузки ВМ.

Изменения, которые будут применены после перезапуска ВМ

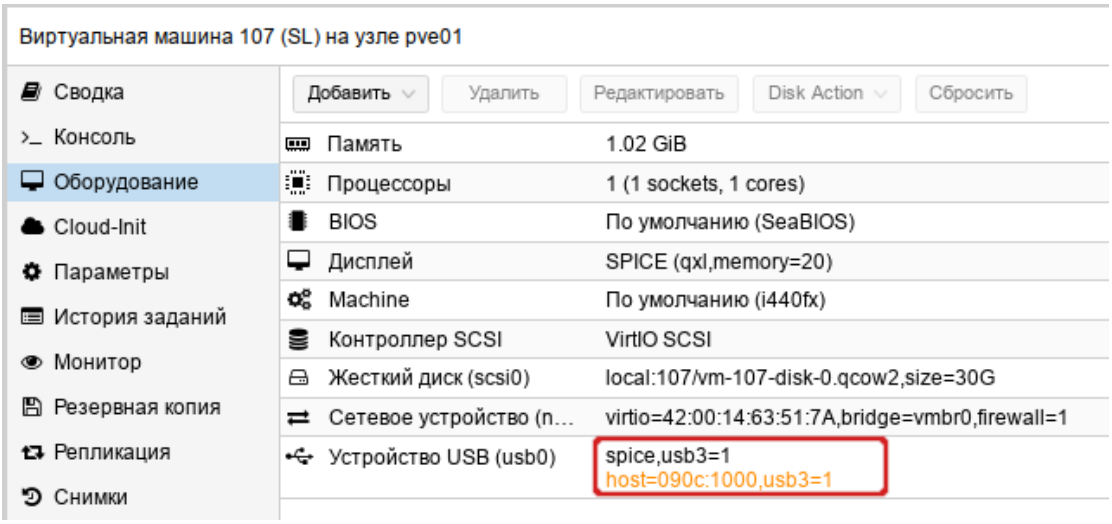


Рис. 114

4.6.5.1 Управление образами виртуальных дисков

Образ виртуального диска является файлом или группой файлов, в которых ВМ хранит свои данные.

`qemu-img` – утилита для манипулирования с образами дисков машин QEMU. `qemu-img` позволяет выполнять операции по созданию образов различных форматов, конвертировать файлы-образы между этими форматами, получать информацию об образах и объединять снимки ВМ для тех форматов, которые это поддерживают (подробнее см. раздел «Утилита qemu-img»).

Примеры, использования утилиты `qemu-img`:

– преобразование (конвертация) vmdk-образа виртуального накопителя VMware под названием test в формат qcow2:

```
# qemu-img convert -f vmdk test.vmdk -O qcow2 test.qcow2
```

– создание образа test в формате RAW, размером 40 ГБ:

```
# qemu-img create -f raw test.raw 40G
```

– изменение размера виртуального диска:

```
# qemu-img resize -f raw test.raw 80G
```

– просмотр информации об образе:

```
# qemu-img info test.raw
```

Для управления образами виртуальных дисков в веб-интерфейсе PVE необходимо выбрать ВМ и перейти на вкладку «Оборудование» («Hardware»). После выбора образа диска станут доступными кнопки (Рис. 115): «Добавить» («Add»), «Отключить» («Remove»), «Редактировать»

(«Edit»), «Изменить размер диска» («Resize»), «Переназначить диск» («Reassign Disk»), «Переместить диск» («Move Storage»).

4.6.5.1.1 Добавление виртуального диска в ВМ

Для добавления образа виртуального диска к ВМ необходимо:

1) перейти на вкладку «Оборудование» («Hardware») (Рис. 115);

2) нажать кнопку «Добавить» («Add») и выбрать в выпадающем списке пункт «Жесткий диск» («Hard Disk») (Рис. 116);

3) указать параметры жесткого диска (Рис. 117) и нажать кнопку «Добавить» («Add»).

Вкладка «Оборудование». Управление образом виртуального диска

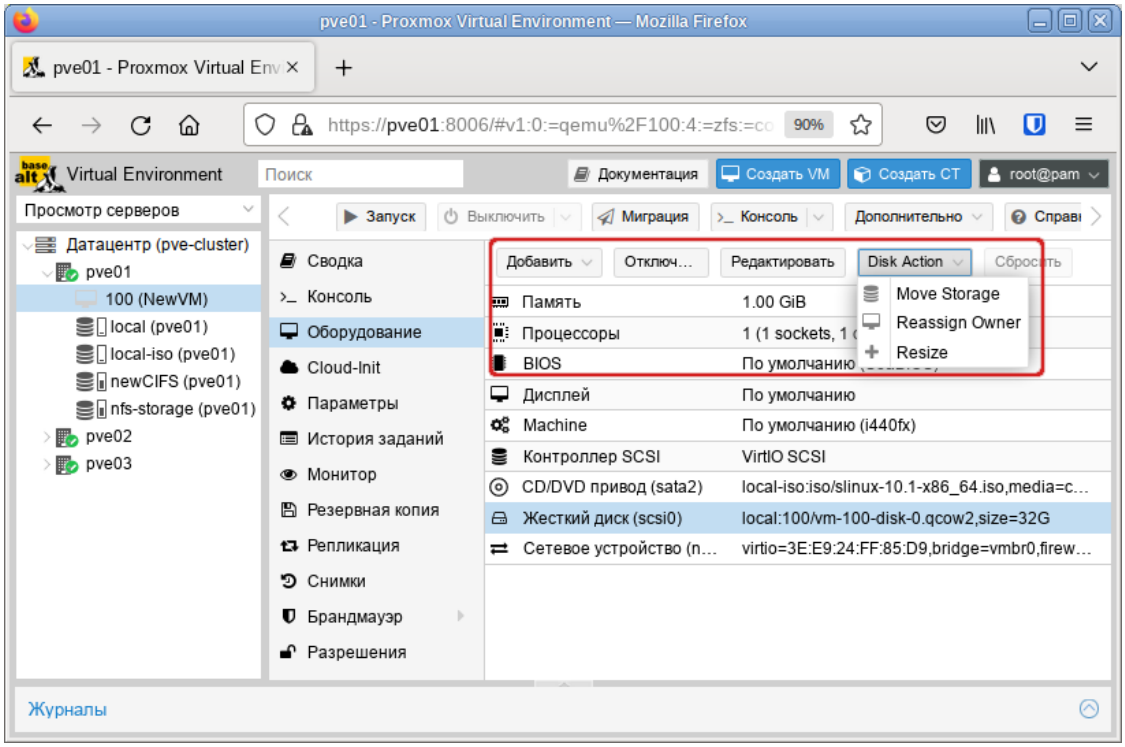


Рис. 115

Кнопка «Добавить»→«Жесткий диск»

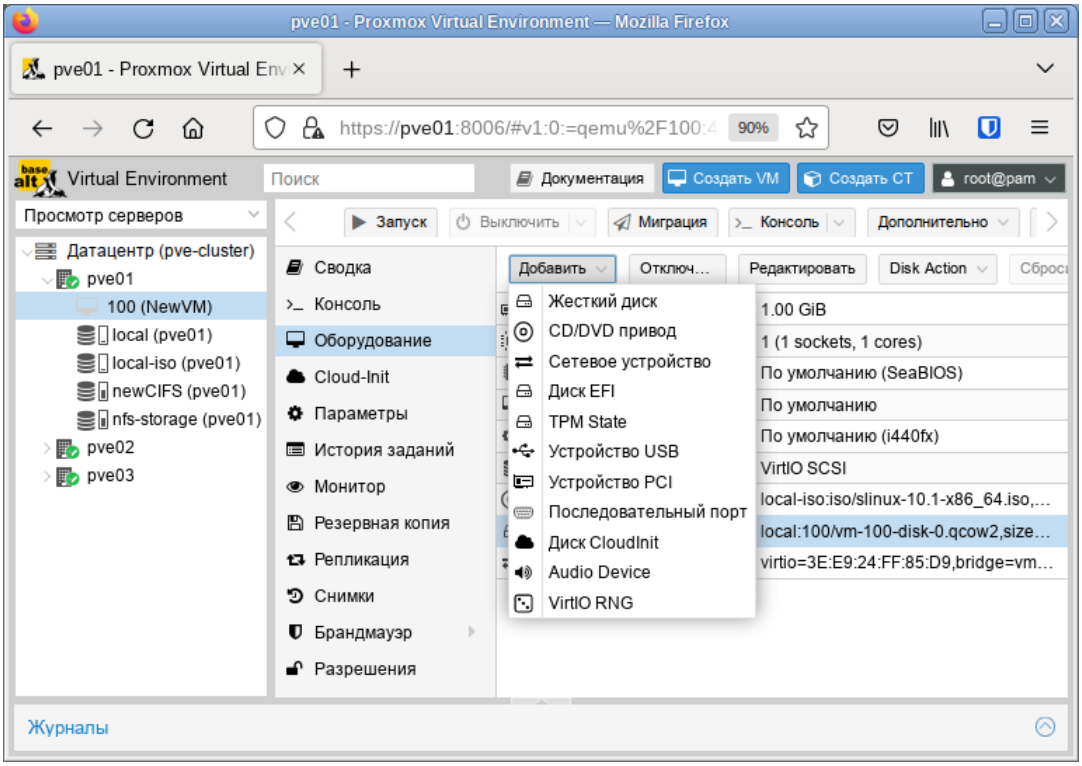


Рис. 116

Опции добавления жесткого диска

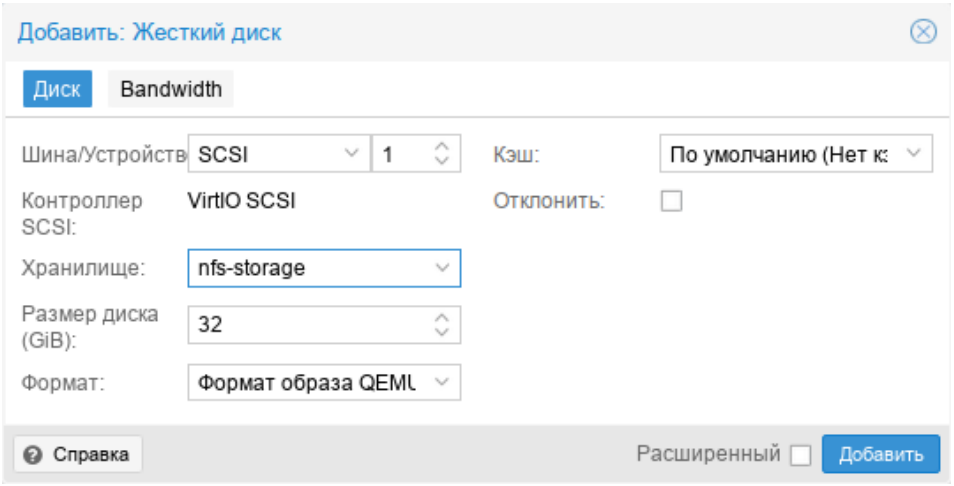


Рис. 117

4.6.5.1.2 Удаление образа виртуального диска

Для удаления образа виртуального диска необходимо:

1) перейти на вкладку «Оборудование» («Hardware») (Рис. 115);
2) выбрать образ диска ВМ;
3) нажать кнопку «Отключить» («Remove»);
4) в окне подтверждения нажать кнопку «Да» («Yes») для подтверждения действия. При этом виртуальный диск будет отсоединен от ВМ, но не удален полностью. Он будет присутствовать в списке оборудования ВМ как «Неиспользуемый диск» («Unused Disk») (Рис. 118).

«Неиспользуемый диск»

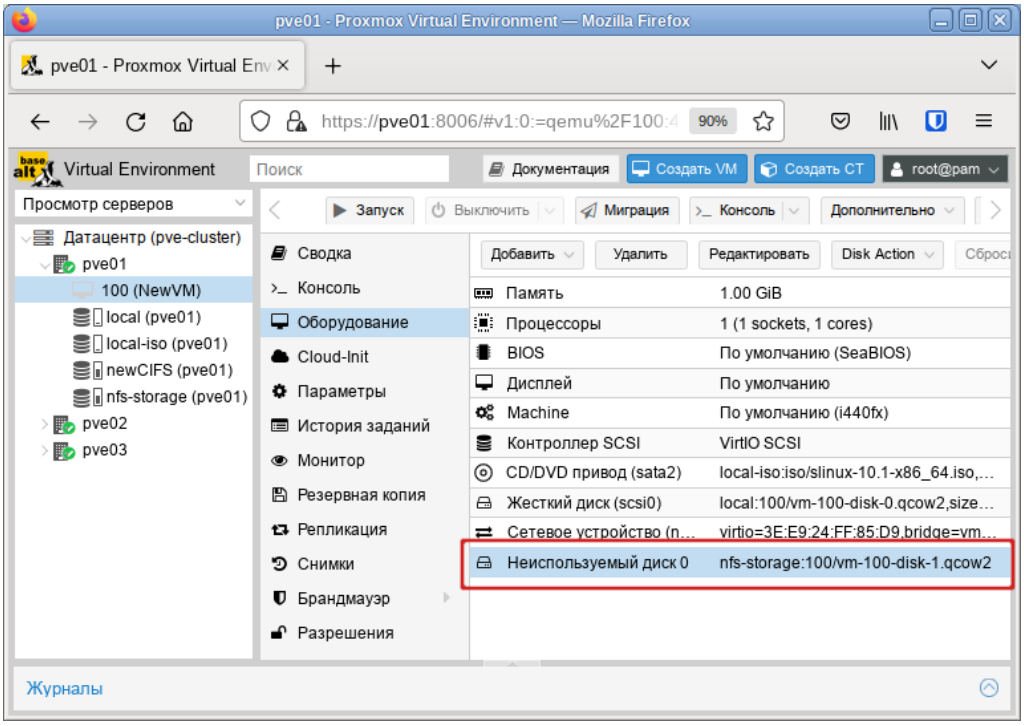


Рис. 118

Чтобы удалить образ диска окончательно, следует выбрать неиспользуемый диск и нажать кнопку «Удалить» («Remove»).

Если образ диска был отключен от ВМ по ошибке, можно повторно подключить его к ВМ, выполнив следующие действия:

1) выбрать неиспользуемый диск;
2) нажать кнопку «Редактировать» («Edit»);
3) в открывшемся диалоговом окне (Рис. 119) изменить, если это необходимо, параметры «Шина/Устройство» («Bus/Device»).
4) нажать кнопку «Добавить» («Add)» для повторного подключения образа диска.

Подключение неиспользуемого диска

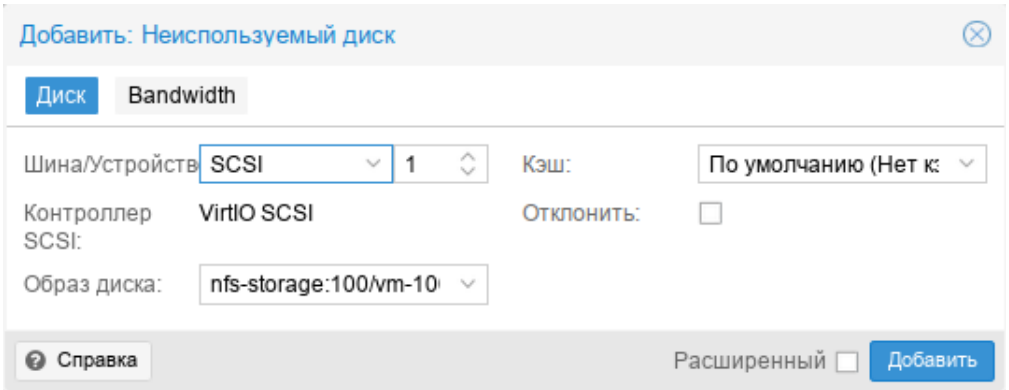


Рис. 119

4.6.5.1.3 Изменение размера диска

Функция изменения размера поддерживает только увеличение размера файла образа виртуального диска.

При изменении размера образа виртуального диска изменяется только размер файла образа виртуального диска. После изменения размера файла, разделы жесткого диска должны быть изменены внутри самой ВМ.

Для изменения размера виртуального диска необходимо:

1) перейти на вкладку «Оборудование» («Hardware») (Рис. 115);
2) выбрать образ виртуального диска.
3) нажать кнопку «Disk Action» → «Resize»;
4) в открывшемся диалоговом окне в поле «Увеличение размера (GiB)» ввести значение, на которое необходимо увеличить размер диска. Например, если размер существующего диска составляет 20 ГБ, для изменения размера диска до 30 ГБ следует ввести число 10 (*Рис. 120*);
5) нажать кнопку «Изменить размер диска» («Resize Disk») для завершения изменения размера.

Изменение размера диска

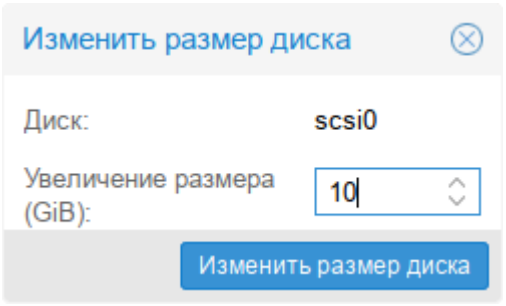


Рис. 120

Команда изменения размера виртуального диска:

```
# qm resize <vm_id> <virtual_disk> [+]<size>
```

Примечание. Если указать размер диска со знаком «+», то данное значение добавится к реальному размеру тома, без знака «+» указывается абсолютное значение. Уменьшение размера диска не поддерживается. Например, изменить размер виртуального диска до 80 ГБ:

```
# qm resize 100 scsi1 80G
```

4.6.5.1.4 Перемещение диска в другое хранилище

Образы виртуального диска могут перемещаться с одного хранилища на другое в пределах одного кластера.

Для перемещения образа диска необходимо:

1) перейти на вкладку «Оборудование» («Hardware») (Рис. 115);
2) выбрать образ диска, который необходимо переместить;

3) нажать кнопку «Disk Action» → «Move Storage»;

4) в открывшемся диалоговом окне (Рис. 121) в выпадающем меню «Целевое хранилище» («Target Storage») выбрать хранилище-получатель, место, куда будет перемещен образ виртуального диска;

5) в выпадающем меню «Формат» («Format») выбрать формат образа диска. Этот параметр полезен для преобразования образа диска из одного формата в другой;

6) отметить, если это необходимо, пункт «Удалить источник» («Delete source») для удаления образа диска из исходного хранилища после его перемещения в новое хранилище;

7) нажать кнопку «Переместить диск» («Move disk»).

Команда перемещения образа диска в другое хранилище:

```
# qm move-disk <vm_id> <virtual_disk> <storage>
```

Диалоговое окно перемещения диска

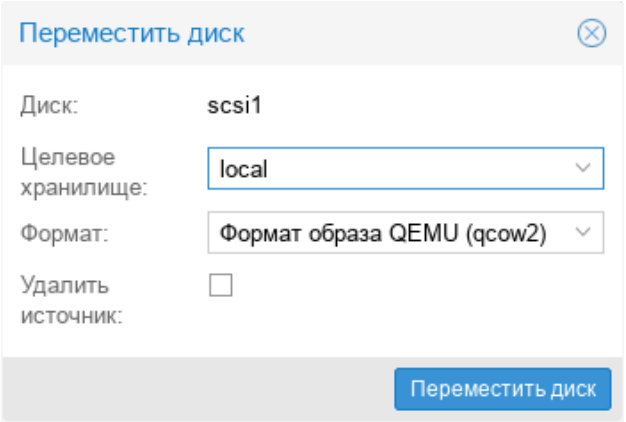


Рис. 121

4.6.5.1.5 Переназначение диска другой ВМ

При переназначении образа диска другой ВМ, диск будет удалён из исходной ВМ и подключен к целевой ВМ.

Для переназначения образа диска другой ВМ необходимо:

1) перейти на вкладку «Оборудование» («Hardware») (Рис. 115);

2) выбрать образ диска, который необходимо переназначить;

3) нажать кнопку «Disk Action» → «Reassign Disk»;

4) в открывшемся диалоговом окне (Рис. 122) в выпадающем «Target Guest» выбрать целевую ВМ (место, куда будет перемещен образ виртуального диска);

5) выбрать нужные параметры в выпадающем меню «Шина/Устройство» («Bus/Device»);

6) нажать кнопку «Переназначить диск» («Reassign Disk»).

Диалоговое окно переназначения диска

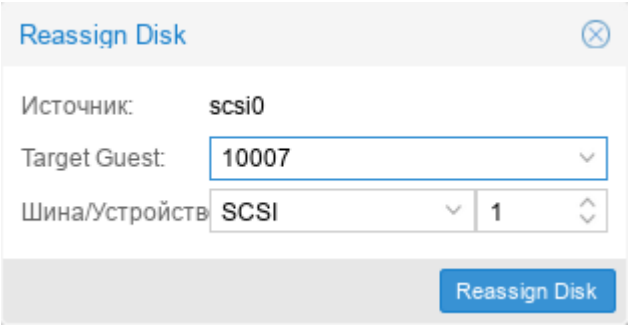


Рис. 122

Команда переназначения образа диска другой ВМ:

```
# qm move-disk <vm_id> <virtual_disk> --target-vmid <vm_id>--target-disk <virtual_disk>
```

Пример удаления образа диска scsi0 из ВМ 107 и подключение его как scsi1 к ВМ 10007:

```
# qm move-disk 107 scsi0 --target-vmid 10007--target-disk scsi1
```

4.6.5.2 Настройки дисплея

QEMU может виртуализировать разные типы оборудования VGA (Рис. 123), например:

- std («Стандартный VGA») – эмулирует карту с расширениями Bochs VBE;
- vmware («Совместимый с VMware») – адаптер, совместимый с VMWare SVGA-II;
- qxl («SPICE») – паравиртуализированная видеокарта QXL. Выбор этого параметра включает SPICE (протокол удаленного просмотра) для ВМ;
- virtio («VirtIO-GPU») – стандартный драйвер графического процессора virtio;
- virtio-gl («VirGL GPU») – виртуальный 3D-графический процессор для использования внутри ВМ, который может переносить рабочие нагрузки на графический процессор хоста.

PVE. Настройки дисплея

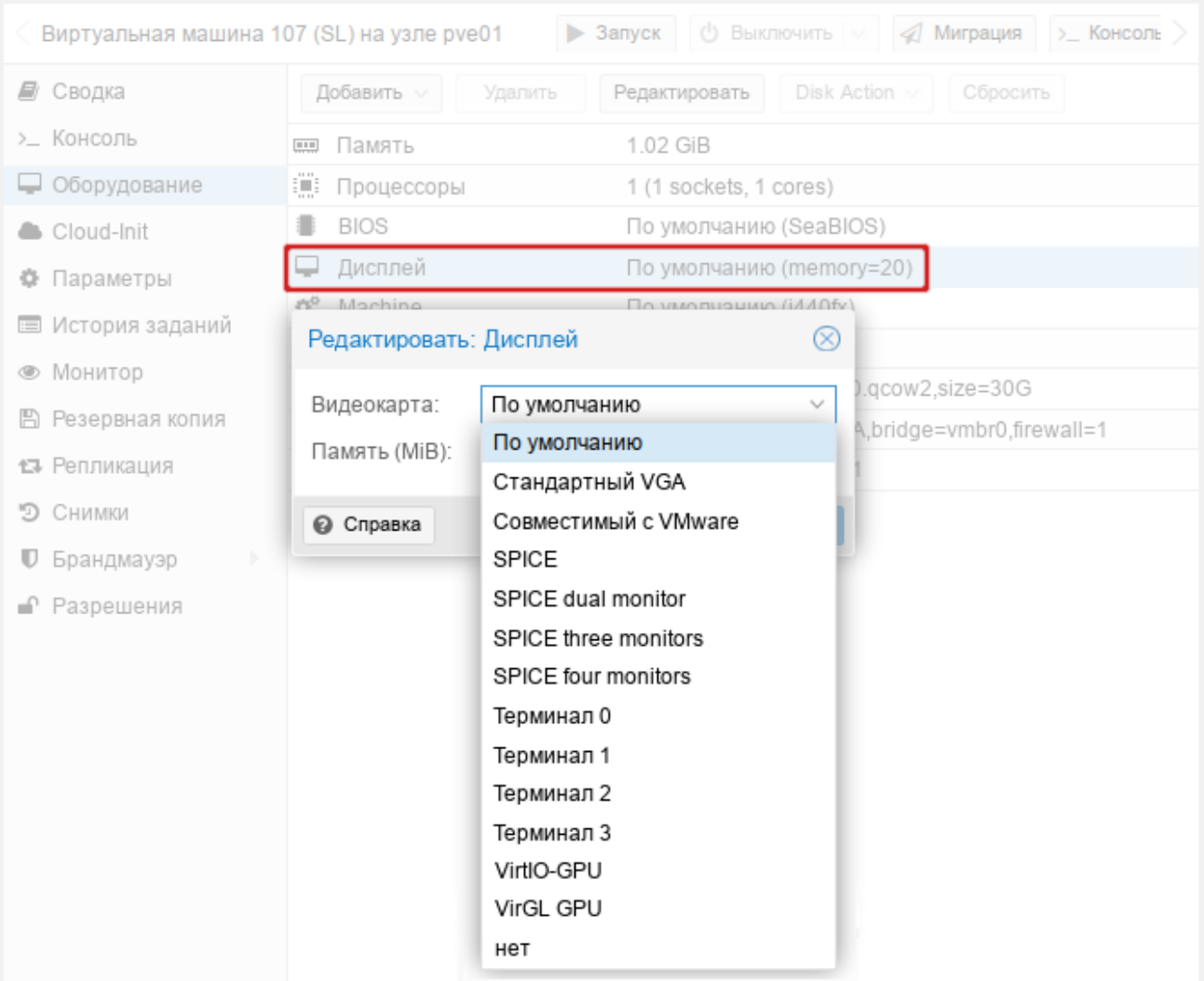


Рис. 123

Примечание. Для типов дисплеев «VirtIO» и «VirGL» по умолчанию включена поддержка SPICE.

Примечание. Для подключения к SPICE-серверу может использоваться любой SPICE-клиент (например, remote-viewer из пакета virt-viewer).

Можно изменить объем памяти, выделяемый виртуальному графическому процессору (поле «Память (MiB)»). Это может обеспечить более высокое разрешение внутри ВМ, особенно при использовании SPICE/QXL.

Поскольку память резервируется устройством дисплея, выбор режима нескольких мониторов для SPICE (например, qxl2 для двух мониторов) имеет некоторые последствия:

– ВМ с ОС Windows требуется устройство для каждого монитора. Поэтому PVE предоставляет ВМ дополнительное устройство для каждого монитора. Каждое устройство получает указанный объем памяти;

– ВМ с ОС Linux всегда могут включать больше виртуальных мониторов, но при выборе режима нескольких мониторов, объём памяти, предоставленный устройству, умножается на количество мониторов.

Выбор serialX («Терминал X») в качестве типа дисплея, отключает выход VGA и перенаправляет веб-консоль на выбранный последовательный порт. В этом случае настроенный параметр памяти дисплея игнорируется.

4.6.5.3 Дополнительные функции SPICE

Дополнительно в PVE можно включить две дополнительные функции SPICE:

– общий доступ к папкам – доступ к локальной папке из ВМ;
– потоковое видео – области быстрого обновления кодируются в видеопоток.

Включение дополнительных функций SPICE:

– в веб-интерфейсе (*Рис. 124*) (пункт «SPICE Enhancements» в разделе «Параметры» ВМ);
– в командной строке:

```
# qm set VMID -spice_enhancements foldersharing=1,videostreaming=all
```

Примечание. Чтобы использовать дополнительные функции SPICE, для параметра Дисплей ВМ должно быть установленно значение SPICE (qxl).

4.6.5.3.1 Общий доступ к папкам

Для возможности получения доступа к локальной папке, внутри ВМ должен быть установлен пакет `spice-webdavd`. В этом случае общая папка будет доступна через локальный сервер WebDAV по адресу http://localhost:9843.

PVE. Дополнительные функции SPICE

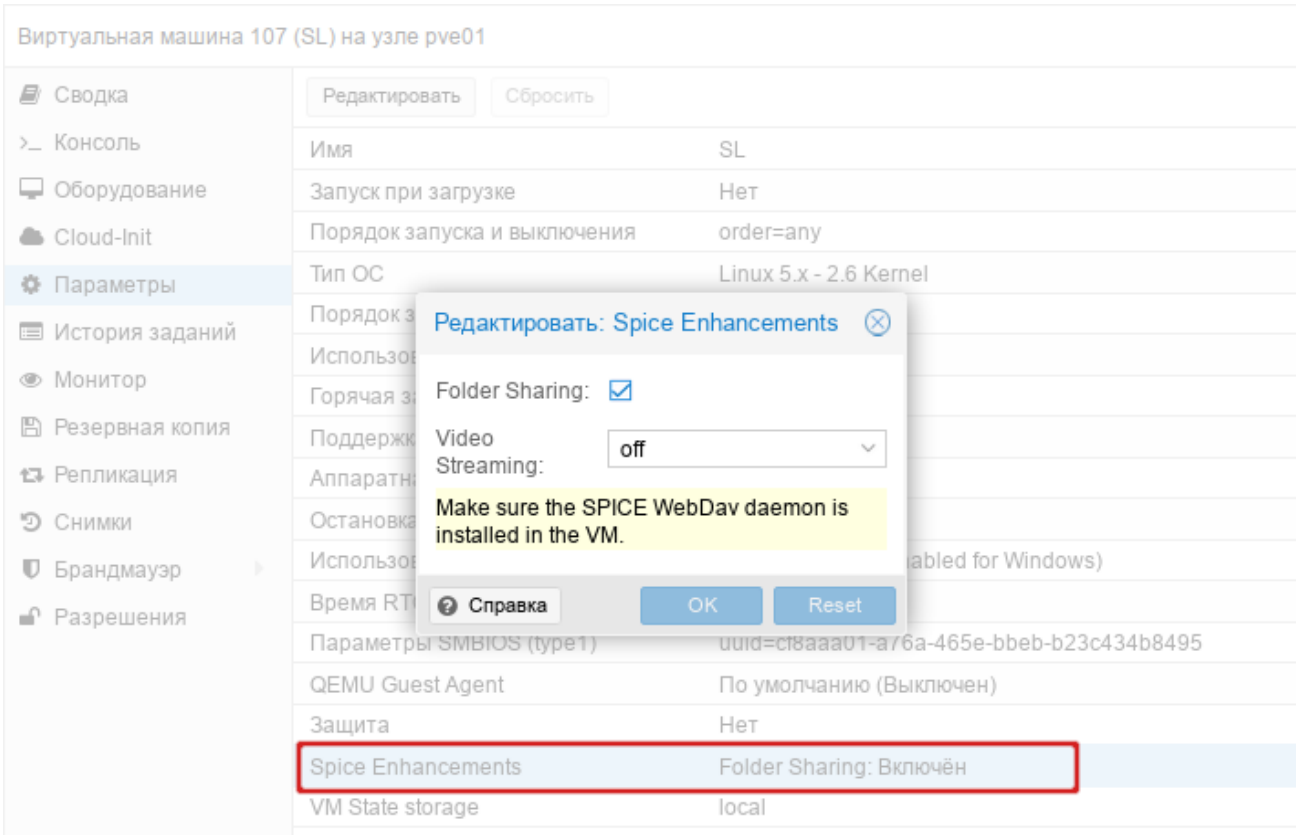


Рис. 124

Примечание. Чтобы открыть общий доступ к папке, следует в меню virt-viewer выбрать пункт «Настройки» («Preferences»), в открывшемся окне установить отметку «Общая папка» и выбрать папку для перенаправления (*Рис. 125*).

Совместный доступ к папке

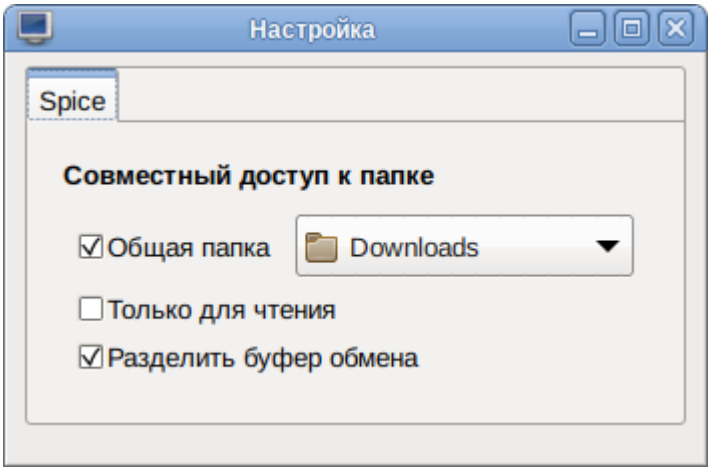


Рис. 125

Если в ВМ общая папка не отображается, следует проверить, что служба WebDAV (spice-webdavd) запущена. Также может потребоваться перезапустить сеанс SPICE.

Для возможности доступа к общей папке из файлового менеджера, а не из браузера, внутри ВМ должен быть установлен пакет davfs2.

Примечание. Для доступа к общей папке из файлового менеджера:

– «Dolphin» – выбрать пункт «Сеть»→«Сетевые службы»→«Сетевой каталог WebDav»→«Spice client folder»;
– «Thunar» – в адресной строке ввести адрес с указанием протокола dav или davs (dav://localhost:9843/).

4.6.5.3.2 Потоковое видео

Доступны две опции:

– «all» – все области быстрого обновления кодируются в видеопоток;
– «filter» – для принятия решения о том, следует ли использовать потоковое видео, используются дополнительные фильтры.

4.6.5.4 Проброс USB

Для проброса USB-устройства в ВМ необходимо:

1) перейти на вкладку «Оборудование» («Hardware») (Рис. 115);
2) нажать кнопку «Добавить» и выбрать в выпадающем списке пункт «Устройство USB» (*Рис. 126*);
3) откроется окно добавления устройства, в котором можно выбрать режим проброса:

– «Порт Spice» – сквозная передача SPICE USB (*Рис. 127*) (позволяет пробросить USB-устройство с клиента SPICE);
– «Использовать устройство USB по номеру» – проброс в ВМ конкретного USB-устройства (*Рис. 128*). USB-устройство можно выбрать в выпадающем списке Choose Device или указать вручную, указав *<ID-производителя>*:*<ID-устройства>* (можно получить из вывода команды `lsusb`).
– «Использовать порт USB» – проброс конкретного порта (*Рис. 129*) (в ВМ будет проброшено любое устройство, вставленное в этот порт). USB-порт можно выбрать в выпадающем списке «Выберите порт» или указать вручную, указав *<Номер_шины>:<Путь_к_порту>* (можно получить из вывода команды `lsusb -t`).

4) нажать кнопку «Добавить»;
5) остановить и запустить ВМ (перезагрузки недостаточно).

Кнопка «Добавить»→«Устройство USB»

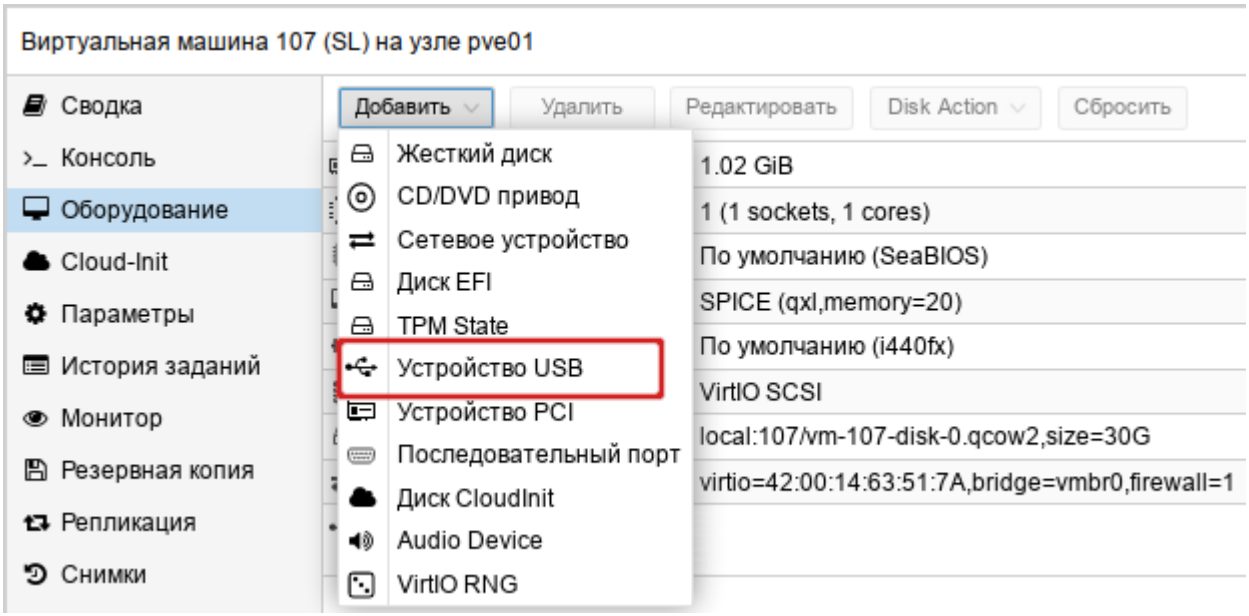


Рис. 126

Порт Spice

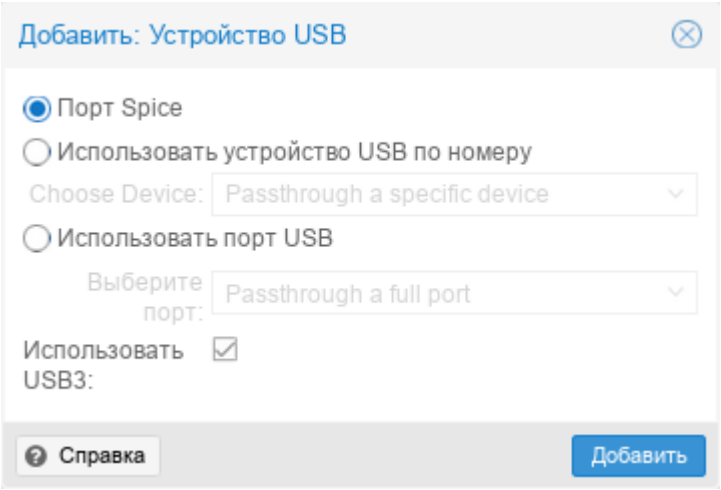


Рис. 127

Использовать устройство USB по номеру

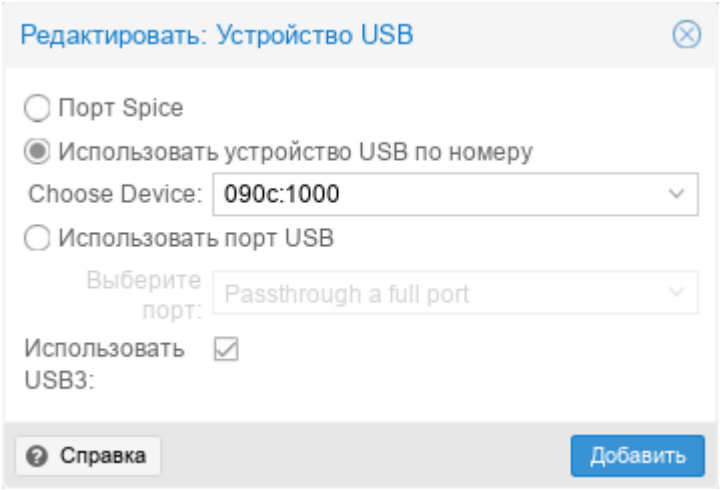


Рис. 128

Использовать порт USB

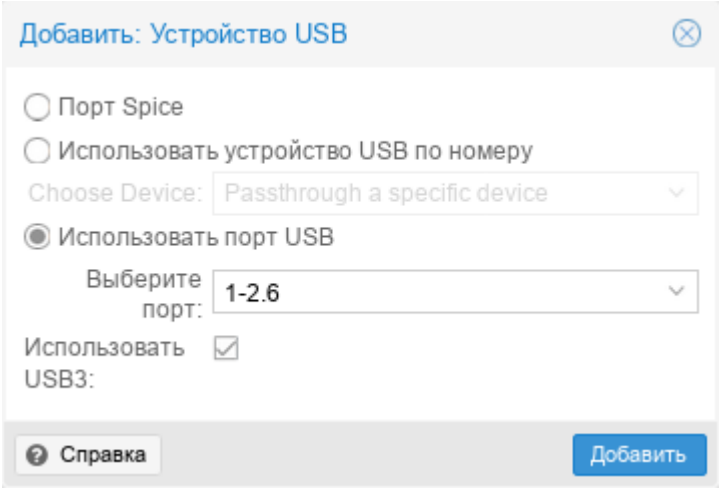


Рис. 129

Примечание. Список подключенных к ВМ и хосту USB-устройств можно получить, введя на вкладке «Монитор» соответственно команды `info usb` или `info usbhost` (*Рис. 130*).

Список подключенных к ВМ и хосту USB-устройств

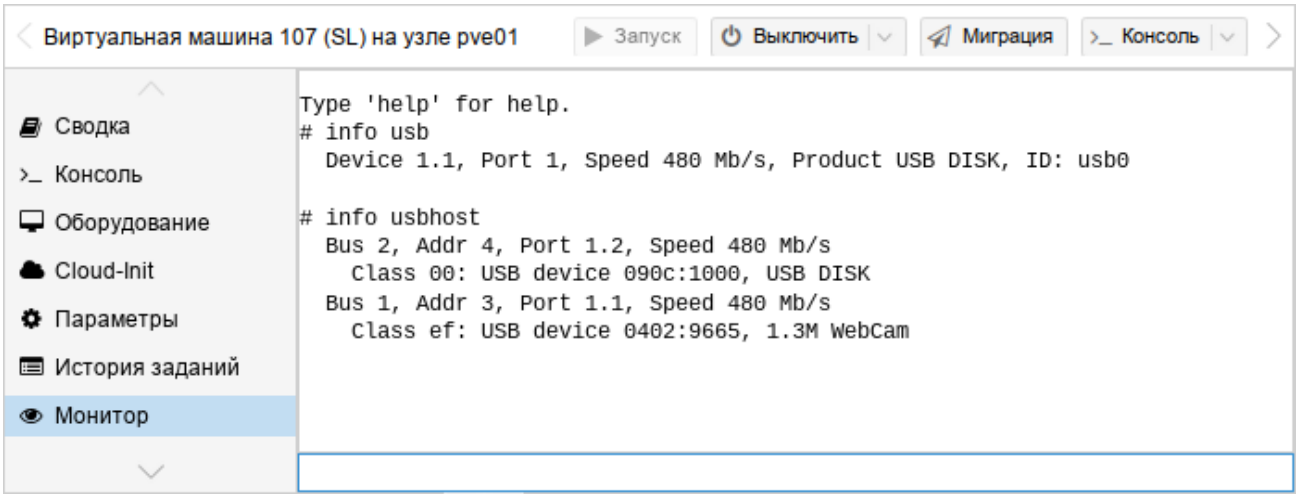


Рис. 130

Если USB-устройство присутствует в конфигурации ВМ (и для него указаны «Использовать устройство USB по номеру» или «Использовать порт USB») при запуске ВМ, но отсутствует на хосте, ВМ будет загружена без проблем. Как только устройство/порт станет доступным на хосте, оно будет проброшено в ВМ.

Примечание. Использование проброса типа «Использовать устройство USB по номеру» или «Использовать порт USB» не позволит переместить ВМ на другой хост, поскольку оборудование доступно только на хосте, на котором в данный момент находится ВМ.

4.6.5.5 BIOS и UEFI

По умолчанию, в качестве прошивки, используется SeaBIOS, который эмулирует BIOS x86. Можно также выбрать OVMF, который эмулирует UEFI.

При использовании OVMF, необходимо учитывать несколько моментов:

- для сохранения порядка загрузки, должен быть добавлен диск EFI (этот диск будет включен в резервные копии и моментальные снимки, и может быть только один);
- при использовании OVMF с виртуальным дисплеем (без проброса видеокарты в ВМ) необходимо установить разрешение клиента в меню OVMF (которое можно вызвать нажатием кнопки ESC во время загрузки) или выбрать SPICE в качестве типа дисплея.

Пример изменения прошивки ВМ на UEFI:

1) поменять тип прошивки на UEFI (Рис. 131);
2) добавить в конфигурацию ВМ EFI Disk (Рис. 132).

PVE. Настройка BIOS

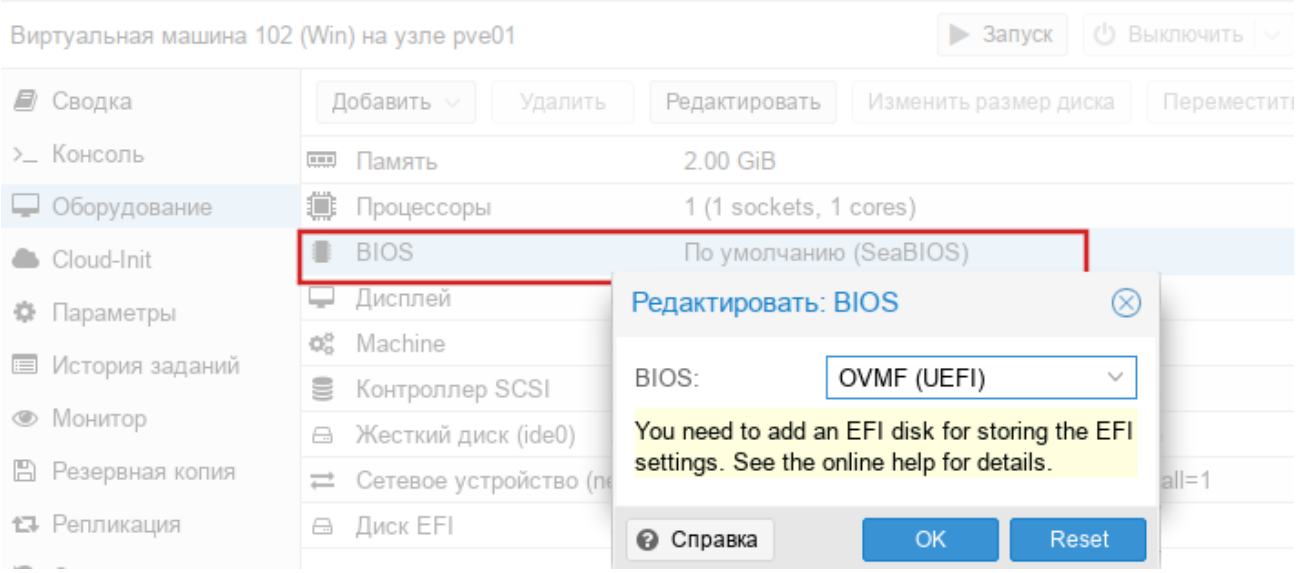


Рис. 131

PVE. Добавление диска EFI

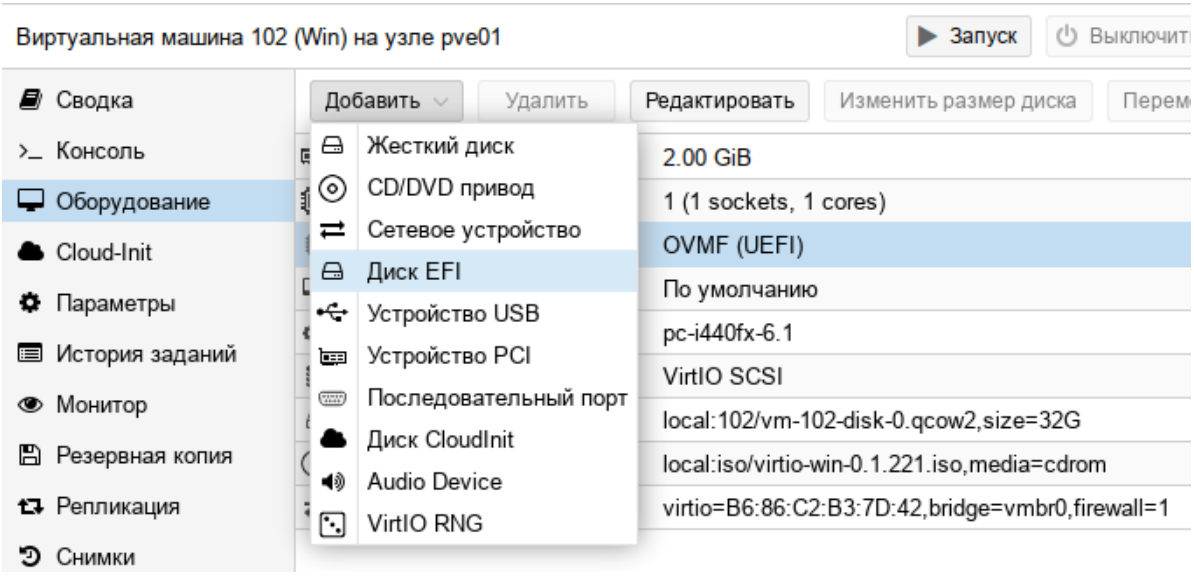


Рис. 132

Команда создания диска EFI:

```
# qm set <vm_id> -efdisk0 <storage>:1,format=<format>,efitype=4m,pre-enrolled-keys=1
```

где:

– <storage> – хранилище, в котором будет размещён диск;

– <format> – формат, поддерживаемый хранилищем;

– efitype – указывает, какую версию микропрограммы OVMF следует использовать. Для новых ВМ необходимо указывать 4м (это значение по умолчанию в графическом интерфейсе);

– pre-enroll-keys – указывает, должен ли efdisk поставляться с предварительно загруженными ключами безопасной загрузки для конкретного дистрибутива и Microsoft Standard Secure Boot. Включает безопасную загрузку по умолчанию.

4.6.5.6 *Доверенный платформенный модуль (TPM)*

TPM (англ. Trusted Platform Module) – спецификация, описывающая криптопроцессор, в котором хранятся криптографические ключи для защиты информации, а также обобщённое наименование реализаций указанной спецификации, например, в виде «чипа TPM» или «устройства безопасности TPM» (Dell).

Доверенный платформенный модуль можно добавить на этапе создания ВМ (вкладка «Система») или для, уже созданной, ВМ.

Добавление TPM в веб-интерфейсе («Добавить» → «TPM State») показано на *Рис. 133*.

Команда добавления TRM:

```
# qm set <vm_id> -tpmstate0 <storage>:1,version=<version>
```

где:

– <storage> – хранилище, в которое будет помещён модуль;

– <version> – версия (1.2 или 2.0).

PVE. Добавление TPM в веб-интерфейсе

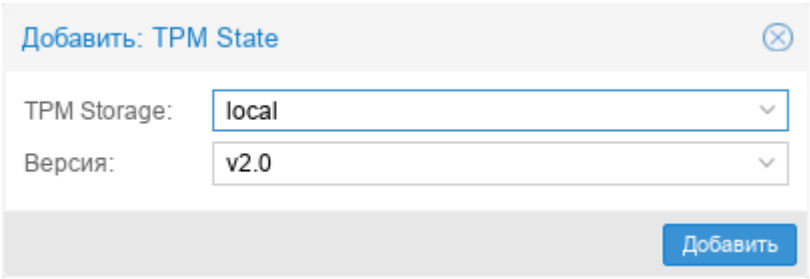


Рис. 133

4.6.5.7 *Проброс PCI(e)*

Проброс PCI(e) – это механизм, позволяющий ВМ управлять устройством PCI(e) хоста.

Примечание. Если устройство передано на ВМ, его нельзя будет использовать на хосте или в любой другой ВМ.

Поскольку пробросPCI(e) – это функция, требующая аппаратной поддержки, необходимо убедиться, что ваше оборудование (ЦП и материнская плата) поддерживает IOMMU (I/O Memory Management Unit).

Если оборудование поддерживает проброс, необходимо выполнить следующую настройку:

1) включить поддержку IOMMU в BIOS/UEFI;

2) для процессоров Intel – передать ядру параметр `intel_iommu=on` (для процессоров AMD он должен быть включен автоматически);

3) убедиться, что следующие модули загружены (этого можно добиться, добавив их в файл /etc/modules):

```
vfio
vfio_iommu_type1
vfio_pci
vfio_virqfd
```

4) перезагрузить систему, чтобы изменения вступили в силу, и убедиться, что проброс действительно включен:

```
# dmesg | grep -e DMAR -e IOMMU -e AMD-Vi
```

Наиболее часто используемый вариант проброса PCI(e) – это проброс всей карты PCI(e), например, GPU или сетевой карты. В этом случае хост не должен использовать карту. Этого можно добиться двумя методами:

- передать идентификаторы устройств в параметры модулей vfio-pci, добавив, например, в файл /etc/modprobe.d/vfio.conf строку:

```
options vfio-pci ids=1234:5678,4321:8765
```

где 1234:5678 и 4321:8765 – идентификаторы поставщика и устройства.

Посмотреть идентификаторы поставщика и устройства можно в выводе команды:

```
# lspci -nn
```

- занести на хосте драйвер в черный список, для этого добавить в файл /etc/modprobe.d/blacklist.conf:

```
blacklist DRIVERNAME
```

Для применения изменений необходимо перезагрузить систему.

Добавления устройства PCI ВМ:

- в веб-интерфейсе («Добавить» → «Устройство PCI» в разделе «Оборудование») (*Рис. 134*). В веб-интерфейсе можно назначить ВМ до 16 устройств PCI(e).
- в командной строке:

```
# qm set VMID -hostpci0 00:02.0
```

Если устройство имеет несколько функций (например, «00:02.0» и «00:02.1»), можно передать их с помощью сокращенного синтаксиса «00:02». Это эквивалентно установке флажка «All Functions» («Все функции») в веб-интерфейсе.

Идентификаторы поставщика и устройства PCI могут быть переопределены для сквозной записи конфигурации, и они необязательно должны соответствовать фактическим идентификаторам физического устройства. Доступные параметры: vendor-id, device-id, sub-vendor-id и sub-device-id. Можно установить любой или все из них, чтобы переопределить идентификаторы устройства по умолчанию:

```
# qm set VMID -hostpci0 02:00,device-id=0x10f6,sub-vendor-id=0x0000
```

PVE. Добавление устройства PCI

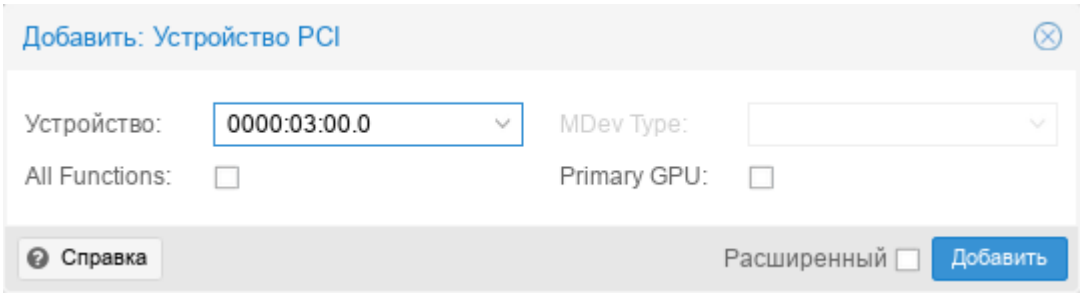


Рис. 134

4.6.6 Файлы конфигурации ВМ

Файлы конфигурации ВМ хранятся в файловой системе кластера PVE (`/etc/pve/qemu-server/<VMID>.conf`). Как и другие файлы, находящиеся в `/etc/pve/`, они автоматически реплицируются на все другие узлы кластера.

Примечание. VMID < 100 зарезервированы для внутренних целей. VMID должны быть уникальными для всего кластера.

Пример файла конфигурации:

```
boot: order=scsi0;scsi7;net0
cores: 1
memory: 2048
name: newVM
net0: virtio=3E:E9:24:FF:85:D9,bridge=vmbr0,firewall=1
numa: 0
ostype: l26
sata2: local-iso:iso/slinux-10.1-x86_64.iso,media=cdrom,size=4586146K
scsi0: local:100/vm-100-disk-0.qcow2,size=42G
scsihw: virtio-scsi-pci
smbios1: uuid=ee9db068-5427-4934-bf7a-5895c377b5af
sockets: 1
vmgenid: dfec8e3b-d391-40cb-8983-b4938461b79a
```

Файлы конфигурации ВМ используют простой формат: разделенные двоеточиями пары ключ/значение (пустые строки игнорируются, строки, начинающиеся с символа #, рассматриваются как комментарии и также игнорируются):

```
OPTION: value
```

Для применения изменений, которые напрямую вносились в файл конфигурации, необходимо перезапустить ВМ. По этой причине рекомендуется использовать команду `qm` для генерации и изменения этих файлов, либо выполнять такие действия в веб-интерфейсе.

При создании снимка ВМ, конфигурация ВМ во время снимка, сохраняется в этом же файле конфигурации в отдельном разделе. Например, после создания снимка «snapshot» файл конфигурации будет выглядеть следующим образом:

```
bootdisk: scsi0
…
parent: snapshot
…
vmgenid: dfec8e3b-d391-40cb-8983-b4938461b79a

[snapshot]
boot: order=scsi0;sata2;net0
cores: 1
memory: 2048
meta: creation-qemu=7.1.0,ctime=1671708251
name: NewVM
net0: virtio=3E:E9:24:FF:85:D9,bridge=vmbr0,firewall=1
numa: 0
ostype: l26
runningcpu: kvm64,enforce,+kvm_pv_eoi,+kvm_pv_unhalt,+lahf_lm,+sep
runningmachine: pc-i440fx-7.1+pve0
sata2: local-iso:iso/slinux-10.1-x86_64.iso,media=cdrom,size=4586146K
scsi0: local:100/vm-100-disk-0.qcow2,size=42G
scsihw: virtio-scsi-pci
smbios1: uuid=ee9db068-5427-4934-bf7a-5895c377b5af
snaptime: 1671724448
sockets: 1
vmgenid: dfec8e3b-d391-40cb-8983-b4938461b79a
vmstate: local:100/vm-100-state-first.raw
```

Свойство `parent` при этом используется для хранения родительских/дочерних отношений между снимками, а `snaptime` – это отметка времени создания снимка (эпоха Unix).

4.7 Создание и настройка контейнера LXC

4.7.1 Создание контейнера в графическом интерфейсе

Перед созданием контейнера можно загрузить шаблоны LXC в хранилище (см. раздел «Управление ISO-образами и шаблонами LXC»).

Для создания контейнера необходимо нажать кнопку «Создать CT», расположенную в правом верхнем углу веб-интерфейса PVE (Рис. 135). Будет запущен диалог «Создать: Контейнер LXC» (Рис. 136), который предоставляет графический интерфейс для настройки контейнера.

На первой вкладке «Общее» («General») необходимо указать (Рис. 136):

– «Узел» («Node») – узел назначения для данного контейнера;
– «CT ID» – идентификатор контейнера в численном выражении;
– «Имя хоста» («Hostname») – алфавитно-цифровая строка названия контейнера;
– «Непривилегированный контейнер» – определяет, как будут запускаться процессы контейнера (если процессам внутри контейнера не нужны полномочия администратора, то необходимо снять отметку с этого пункта);
– «Пул ресурсов» («Resource Pool») – логическая группа контейнеров. Чтобы иметь возможность выбора, пул должен быть предварительно создан;
– «Пароль» («Password») – пароль для данного контейнера;
«Открытый SSH ключ» («SSH public key») – ssh ключ.

Создание контейнера

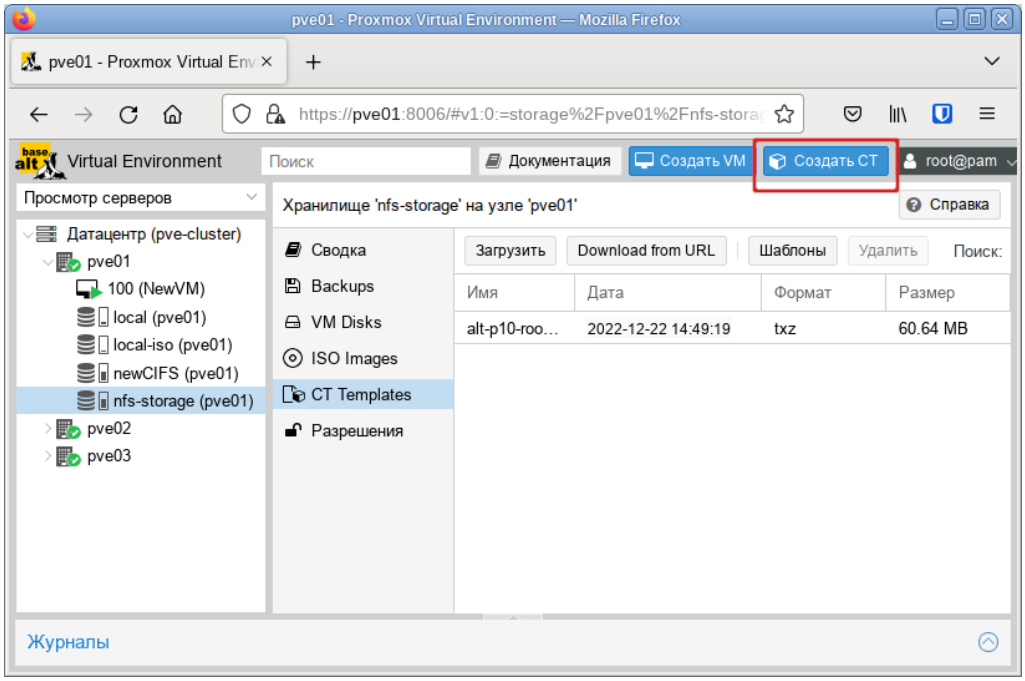


Рис. 135

Вкладка «Общее» диалога создания контейнера

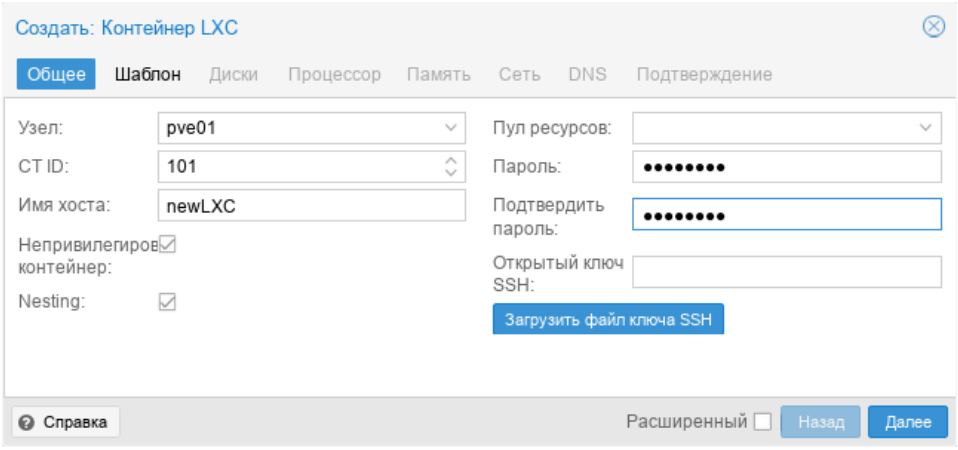


Рис. 136

На вкладке «Шаблон» («Template») следует выбрать (Рис. 137):

– «Хранилище» («Storage») – хранилище в котором хранятся шаблоны LXC;

– «Шаблон» («Template») – шаблон контейнера.

Вкладка «Шаблон» диалога создания контейнера

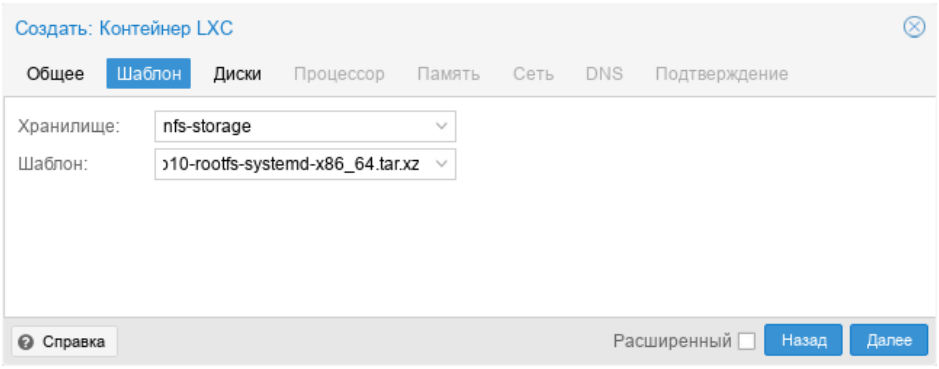


Рис. 137

На вкладке «Диски» («Disks») определяется хранилище, где будут храниться диски контейнера (Рис. 138). Здесь также можно определить размер виртуальных дисков (не следует выбирать размер диска менее 4 ГБ).

Вкладка «Диски» диалога создания контейнера

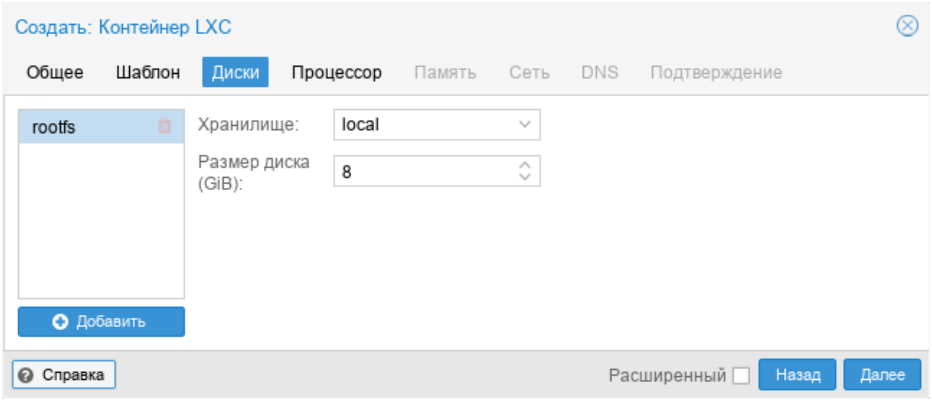


Рис. 138

На вкладке «Процессор» («CPU») определяется количество ядер процессора, которые будут выделены контейнеру (Рис. 139).

Вкладка «Процессор» диалога создания контейнера

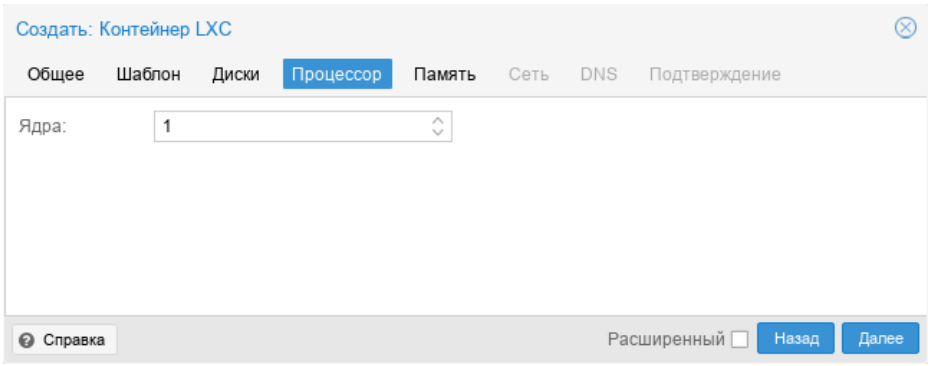


Рис. 139

На вкладке «Память» («Memory») настраиваются (Рис. 140):

– «Память» («Memory») (MiB) – выделяемая память в мегабайтах;
– «Подкачка» («Swap») (MiB) – выделяемое пространство подкачки в мегабайтах.

Вкладка «Память» диалога создания контейнера

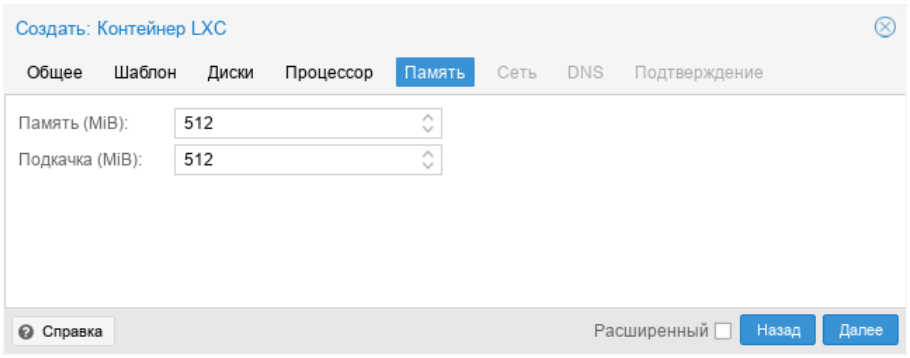


Рис. 140

Вкладка «Сеть» («Network») включает следующие настройки (Рис. *141*):

– «Имя» («Name») – определяет, как будет именоваться виртуальный сетевой интерфейс внутри контейнера (по умолчанию eth0);
– «Адрес MAC» («MAC address») – можно задать определенный MAC-адрес, необходимый для приложения в данном контейнере (по умолчанию, все MAC-адреса для виртуальных сетевых интерфейсов назначаются автоматически);
– «Сетевой мост» («Bridge») – выбор виртуального моста, к которому будет подключаться данный интерфейс (по умолчанию vmbr0);
– «Тег VLAN» («VLAN Tag») – применяется для установки идентификатора VLAN для данного виртуального интерфейса;
– «Ограничение трафика» («Rate limit») (MBit/s) – ограничение пропускной способности сетевой среды (в Мб/с). Для работы без ограничений следует оставить поле пустым;

– «Брандмауэр» («Firewall») – поддержка межсетевого экрана (если пункт отмечен, применяются правила хоста);
– «IPv4/IPv6» – можно настроить и IPv4, и IPv6 для виртуального сетевого интерфейса. IP-адреса можно устанавливать вручную или разрешить получать от DHCP-сервера для автоматического назначения IP. IP-адрес должен вводиться в нотации CIDR (например, 192.168.0.30/24).

Вкладка «Сеть» диалога создания контейнера

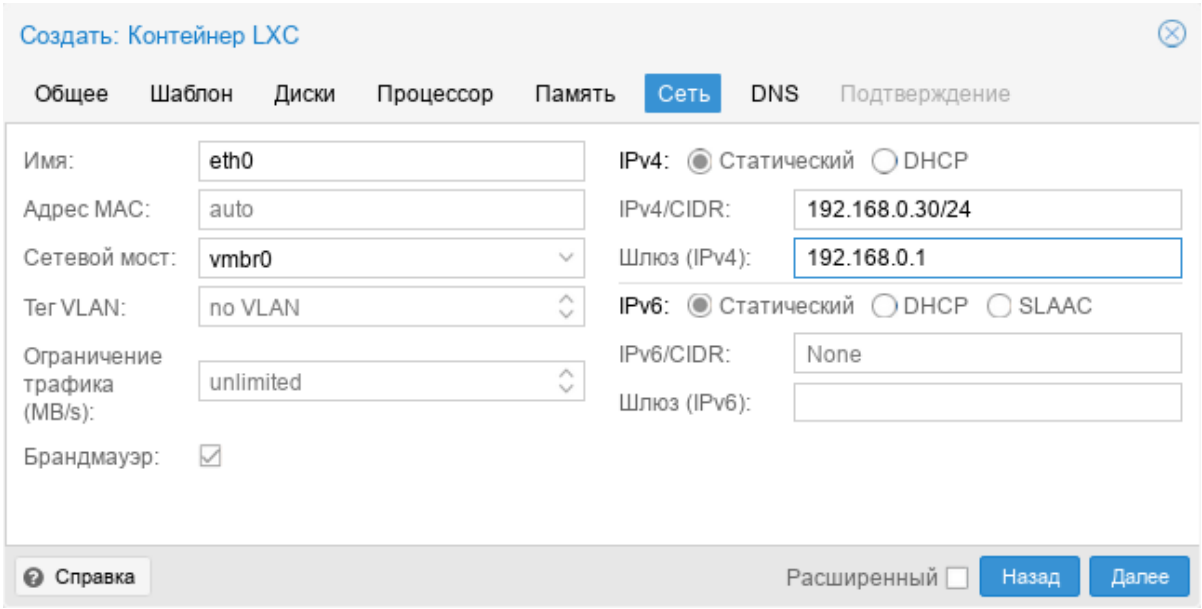


Рис. 141

Вкладка «DNS» содержит настройки (Рис. 142):

– «Домен DNS» («DNS domain») – имя домена (по умолчанию используются параметры хост системы);
– «Серверы DNS» («DNS server») – IP-адреса серверов DNS (по умолчанию используются параметры хост системы).

Вкладка «DNS» диалога создания контейнера

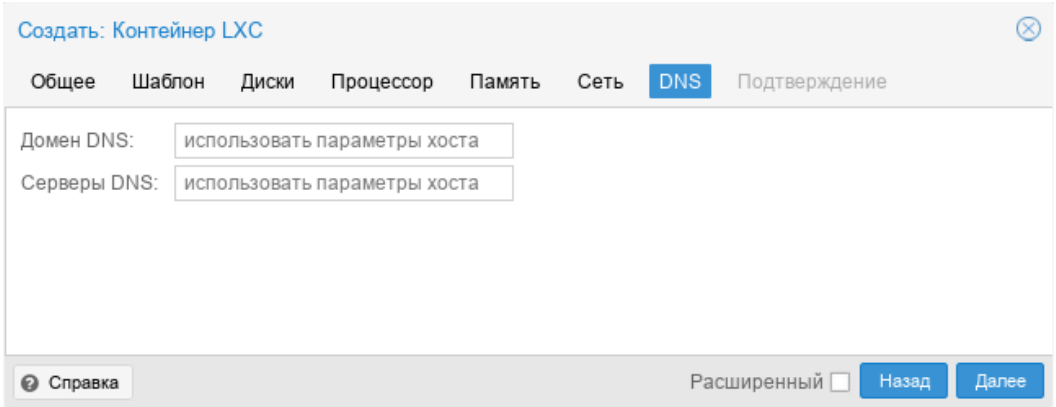


Рис. 142

Во вкладке «Подтверждение» («Confirm») отображаются все введенные или выбранные значения для данного контейнера (Рис. 143). Для создания контейнера необходимо нажать кнопку «Готово» («Finish»). Если необходимо внести изменения в параметры контейнера, можно перейти по вкладкам назад.

Вкладка «Подтверждение» диалога создания контейнера

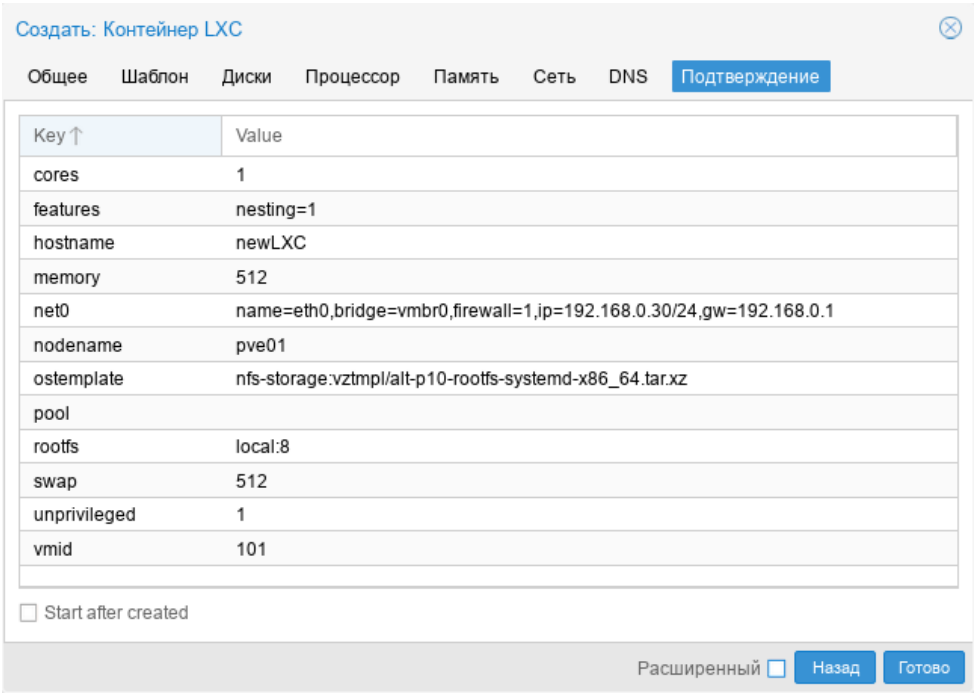


Рис. 143

Если отметить пункт «Start after created» контейнер будет запущен сразу после создания.

После нажатия кнопки «Готово» во вкладке «Подтверждение», диалог настройки закрывается и в браузере открывается новое окно, которое предлагает возможность наблюдать за построением PVE контейнера LXC из шаблона (Рис. 144).

Создание контейнера

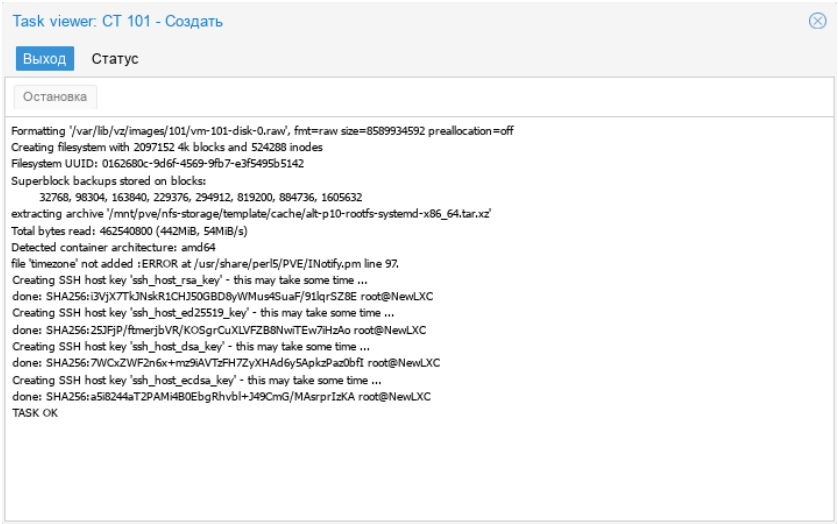


Рис. 144

4.7.2 Создание контейнера из шаблона в командной строке

Контейнер может быть создан из шаблона в командной строке хоста.

Следующий bash-сценарий иллюстрирует применение команды `pct` для создания контейнера:

```
#!/bin/bash
#### Set Variables ####
hostname="pve01"
vmid="104"
template_path="/var/lib/vz/template/cache"
storage="local"
description="alt-p10"
template="alt-p10-rootfs-systemd-x86_64.tar.xz"
ip="192.168.0.93/24"
nameserver="8.8.8.8"
ram="1024"
rootpw="password"
rootfs="4"
gateway="192.168.0.1"
bridge="vmbr0"
if="eth0"
#### Execute pct create using variable substitution ####
pct create $vmid \
  $template_path/$template \
  -description $description \
  -rootfs $rootfs \
  -hostname $hostname \
  -memory $ram \
  -nameserver $nameserver \
  -storage $storage \
  -password $rootpw \
  -net0 name=$if,ip=$ip,gw=$gateway,bridge=$bridge
```

4.7.3 Изменение настроек контейнера

Изменения в настройки контейнера можно вносить и после его создания. При этом изменения сразу же вступают в действие, без необходимости перезагрузки контейнера.

Есть три способа, которыми можно регулировать выделяемые контейнеру ресурсы:

- веб-интерфейс PVE;
- командная строка;
- изменение файла конфигурации.

4.7.3.1 Изменение настроек в веб-интерфейсе

В большинстве случаев изменение настроек контейнера и добавление виртуальных устройств может быть выполнено в веб-интерфейсе.

Для изменения настроек контейнера можно использовать вкладки (Рис. 145):

- «Ресурсы» (оперативная память, подкачка, количество ядер ЦПУ, размер диска);

– «Сеть»;
– «DNS»;
– «Параметры».

Изменений настроек контейнера в веб-интерфейсе PVE

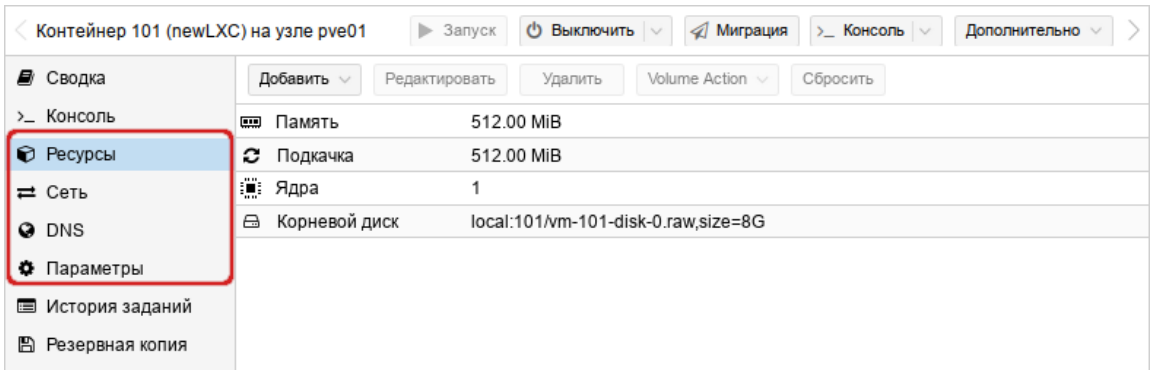


Рис. 145

Для редактирования ресурсов следует выполнить следующие действия:

1) в режиме просмотра по серверам выбрать контейнер;
2) перейти на вкладку «Ресурсы» («Resource»);
3) выбрать элемент для изменения: «Память» («Memory»), «Подкачка» («Swap») или «Ядра» («Cores»), и нажать кнопку «Редактировать» («Edit»);
4) в открывшемся диалоговом окне ввести нужные значения и нажать кнопку «OK».

Если необходимо изменить размер диска контейнера, например, увеличить до 18 ГБ вместо предварительно созданного 8 ГБ, нужно выбрать элемент «Корневой диск» («Root Disk»), нажать кнопку «Volume Action» → «Reassign Disk»«Изменить размер диска» («Resize»), в открывшемся диалоговом окне ввести значение увеличения размера диска (Рис. 146) и нажать кнопку «Изменить размер диска».

Изменение размера диска

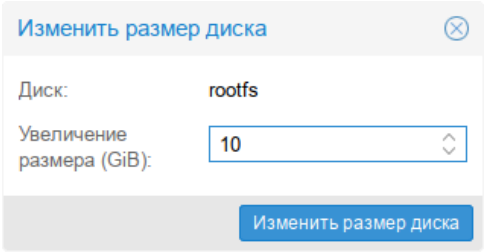


Рис. 146

Для перемещения образа диска в другое хранилище, нужно выбрать элемент «Корневой диск» («Root Disk»), нажать кнопку «Volume Action» → «Move Storage», в открывшемся окне (*Рис. 147*) в выпадающем меню «Целевое хранилище» выбрать хранилище-получатель, отметить, если это необходимо, пункт «Удалить источник» для удаления образа диска из исходного хранилища и нажать кнопку «Переместить том».

Диалоговое окно перемещения тома

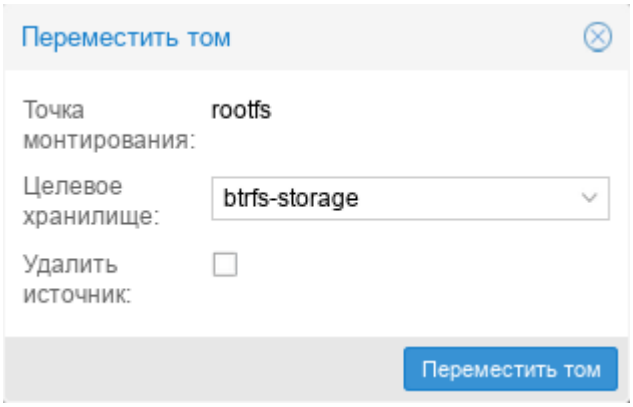


Рис. 147

Для изменения сетевых настроек контейнера необходимо:

1) в режиме просмотра по серверам выбрать контейнер;

2) перейти на вкладку «Сеть» («Network»). На экране отобразятся все настроенные для контейнера виртуальные сетевые интерфейсы (Рис. 148);

3) выбрать интерфейс и нажать кнопку «Редактировать» («Edit») (Рис. 149);

4) после внесения изменений нажать кнопку «OK».

Виртуальные сетевые интерфейсы контейнера

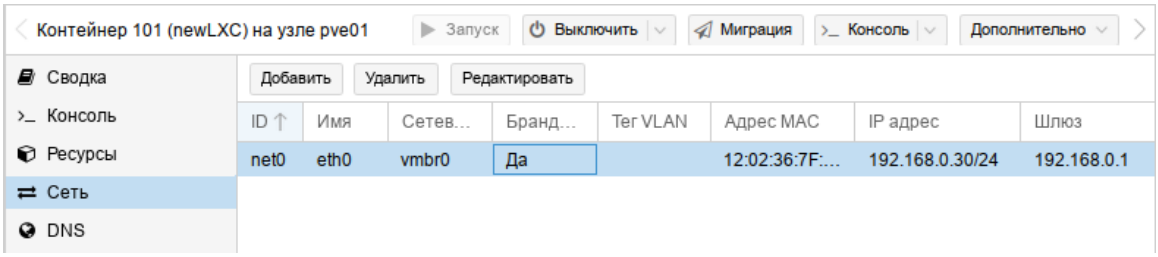


Рис. 148

Изменение сетевых настроек контейнера

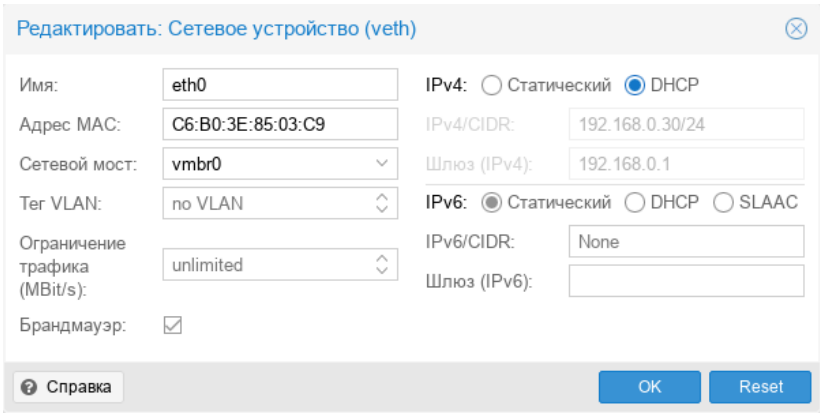


Рис. 149

На вкладке «Параметры» («Options») можно отредактировать разные настройки контейнера (*Рис. 150*), например, «Режим консоли» («Console mode»):

– «tty» – открывать соединение с одним из доступных tty-устройств (по умолчанию);

– «shell» – вызывать оболочку внутри контейнера (без входа в систему);

– «/dev/console» – подключаться к /dev/console.

Примечание. В случаях, когда изменение не может быть выполнено в горячем режиме, оно будет зарегистрировано как ожидающее изменение (выделяется цветом, см. *Рис. 151*). Такие изменения будут применены только после перезапуска контейнера.

Изменение настроек контейнера. Вкладка «Параметры»

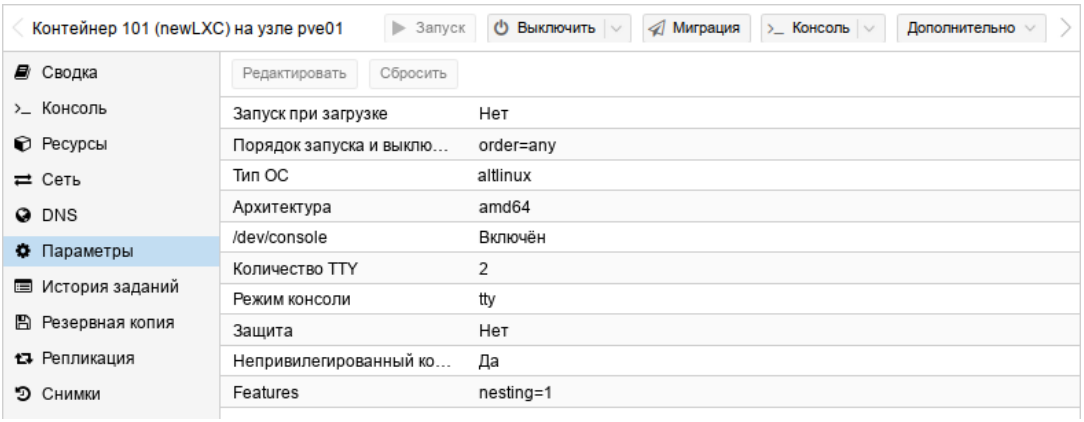


Рис. 150

Изменения, которые будут применены после перезапуска контейнера

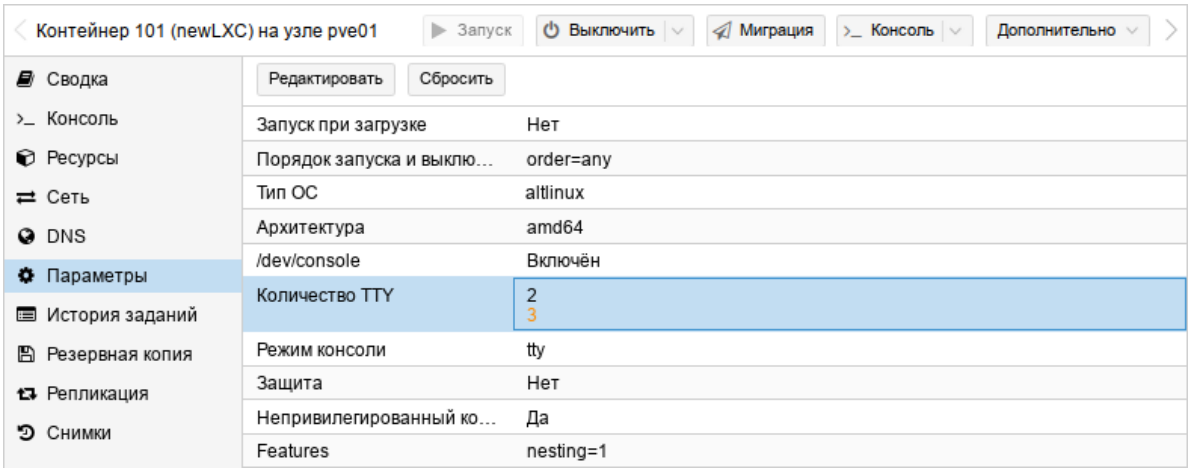


Рис. 151

4.7.3.2 Настройка ресурсов в командной строке

Если веб-интерфейс PVE недоступен, можно управлять контейнером в командной строке (либо через сеанс SSH, либо из консоли noVNC, или зарегистрировавшись на физическом хосте).

`pct` – утилита управления контейнерами LXC в PVE. Чтобы просмотреть доступные для контейнеров команды PVE, можно выполнить следующую команду:

```
# pct help
```

Формат использования команды для изменения ресурсов контейнера:

```
# pct set <ct_id> [options]
```

Например, изменить IP-адрес контейнера #101:

```
# pct set 101 –net0 name=eth0,bridge=vmbr0,ip=192.168.0.17/24,gw=192.168.0.1
```

Изменить количество выделенной контейнеру памяти:

```
# pct set <ct_id> -memory <int_value>
```

Команда изменения размера диска контейнера:

```
# pct set <ct_id> -rootfs <volume>,size=<int_value for GB>
```

Например, изменить размер диска контейнера #101 до 10 ГБ:

```
# pct set 101 -rootfs local:101/vm-101-disk-0.raw,size=10G
```

Показать конфигурацию контейнера:

```
pct config <ct_id>
```

Разблокировка заблокированного контейнера:

```
# pct unlock <ct_id>
```

Список контейнеров LXC данного узла:

```
# pct list
VMID       Status     Lock         Name
101        running                 newLXC
102        stopped                 pve01
103        stopped                 LXC2
```

Запуск и останов контейнера LXC из командной строки:

```
# pct start <ct_id>
# pct stop <ct_id>
```

4.7.3.3 Настройка ресурсов прямым изменением

В PVE файлы конфигурации контейнеров находятся в каталоге `/etc/pve/lxc`, а файлы конфигураций ВМ – в `/etc/pve/qemu-server/`.

У контейнеров LXC есть большое число параметров, которые не могут быть изменены в веб-интерфейсе или с помощью утилиты `pct`. Эти параметры могут быть настроены только путем изменений в файл конфигурации с последующим перезапуском контейнера.

Пример файла конфигурации контейнера `/etc/pve/lxc/102.conf`:

```
arch: amd64
cmode: shell
console: 0
cores: 1
features: nesting=1
hostname: newLXC
memory: 512
```

```
net0:
name=eth0,bridge=vmbr0,firewall=1,gw=192.168.0.1,hwaddr=C6:B0:3E:85:03
:C9,ip=192.168.0.30/24,type=veth
ostype: altlinux
rootfs: local:101/vm-101-disk-0.raw,size=8G
swap: 512
tty: 3
unprivileged: 1
```

4.7.4 Запуск и останов контейнеров

4.7.4.1 Изменение состояния контейнера в веб-интерфейсе

Для запуска контейнера следует выбрать его в левой панели; его иконка должна быть серого цвета, обозначая, что контейнер не запущен (Рис. 152).

Запустить контейнер можно, выбрав в контекстном меню контейнера пункт «Запуск» (Рис. 152), либо нажав кнопку «Запуск» («Start») (Рис. 153).

Запущенный контейнер будет обозначен зеленой стрелкой на значке контейнера.

Для запущенного контейнера доступны следующие действия (Рис. 153):

– «Выключить» («Shutdown») – остановка контейнера;

– «Остановка» («Stop») – остановка контейнера, путем прерывания его работы;

– «Перезапустить» («Reboot») – перезапуск контейнера.

Контекстное меню контейнера

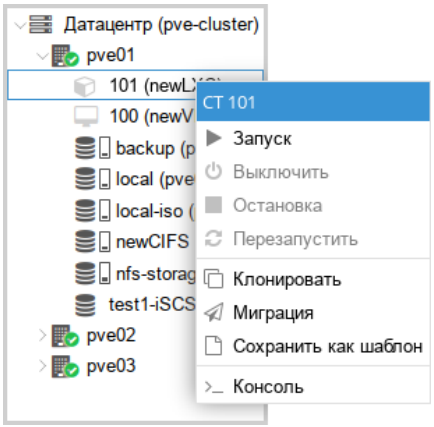


Рис. 152

Кнопки управления состоянием контейнера

Рис. 153

4.7.4.2 Изменение состояний контейнера в командной строке

Состоянием контейнера можно управлять из командной строки PVE (либо через сеанс SSH, либо из консоли noVNC, или зарегистрировавшись на физическом хосте).

Для запуска контейнера с VM ID 102 необходимо ввести команду:

```
# pct start 102
```

Этот же контейнер может быть остановлен при помощи команды:

```
# pct stop 102
```

4.7.5 Доступ к LXC контейнеру

Способы доступа к LXC контейнеру:

- консоль: noVNC, SPICE или xterm.js;
- SSH;
- интерфейс командной строки PVE.

Можно получить доступ к контейнеру из веб-интерфейса при помощи консоли noVNC. Это почти визуализированный удаленный доступ к экземпляру.

Для доступа к запущенному контейнеру в консоли следует выбрать в веб-интерфейсе нужный контейнер, а затем нажать кнопку «Консоль» («Console») и в выпадающем меню выбрать нужную консоль (Рис. 154).

Кнопка «Консоль»

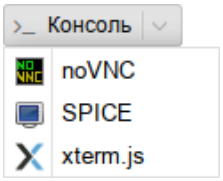


Рис. 154

Консоль также можно запустить, выбрав вкладку «Консоль» («Console») для контейнера (Рис. 155).

Консоль

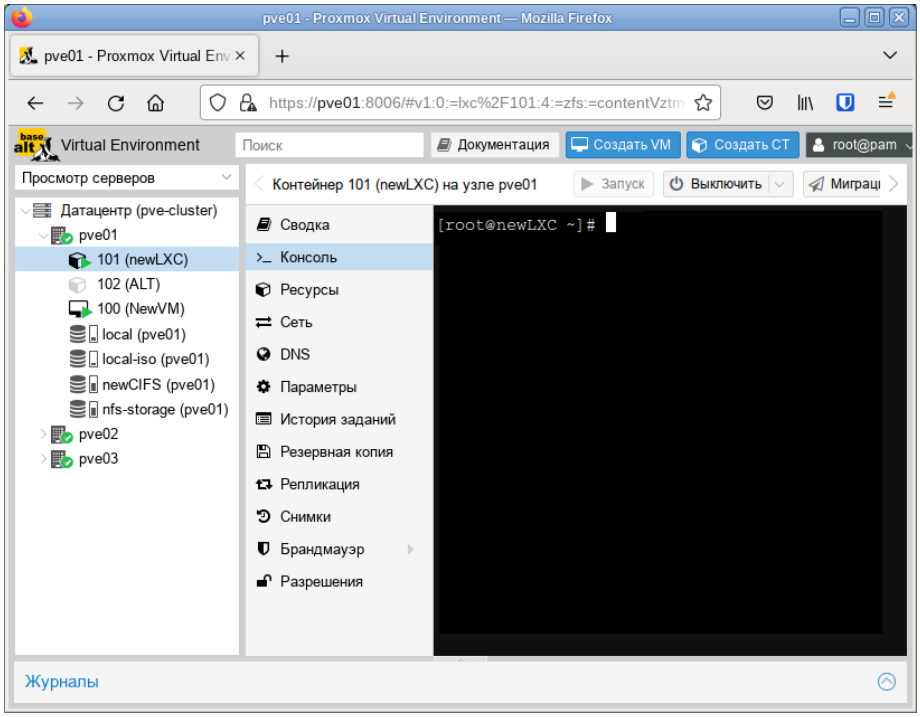


Рис. 155

Одной из функций LXC контейнера является возможность прямого доступа к оболочке контейнера через командную строку его узла хоста. Команда для доступа к оболочке контейнера LXC:

```
# pct enter <ct_id>
```

Данная команда предоставляет прямой доступ на ввод команд внутри указанного контейнера:

```
[root@pve01 ~]# pct enter 101
[root@newLXC ~]#
```

Таким образом был получен доступ к контейнеру LXC с именем newLXC на узле pve01. При этом для входа в контейнер не был запрошен пароль. Так как контейнер работает под пользователем root, можно выполнять внутри этого контейнера любые задачи. Завершив их, можно просто набрать `exit`.

П р и м е ч а н и е . При возникновении ошибки:

```
Insecure $ENV{ENV} while running with...
```

необходимо закомментировать строку: `"ENV=$HOME/.bashrc"` в файле `/root/.bashrc`.

Команды можно выполнять внутри контейнера без реального входа в такой контейнер:

```
# pct exec <ct_id> -- <command>
```

Например, создать каталог внутри контейнера и проверить, что этот каталог был создан:

```
# pct exec 101 mkdir /home/demouser
# pct exec 101 ls /home
demouser
```

Для выполнения внутри контейнера команды с параметрами необходимо изменить команду `pct`, добавив `--` после идентификатора контейнера:

```
# pct exec 101 -- df -H /
Файловая система Размер Использовано  Дост Использовано% Смонтировано в
/dev/loop0         8,4G         516M  7,4G            7% /
```

4.8 Миграция виртуальных машин и контейнеров

В случае, когда PVE управляет не одним физическим узлом, а кластером физических узлов, должна обеспечиваться возможность миграции ВМ с одного физического узла на другой. Миграция представляет собой заморозку состояния ВМ на одном узле, перенос данных и конфигурации на другой узел, и разморозку состояния ВМ на новом месте. Возможные сценарии, при которых может возникнуть необходимость миграции:

- отказ физического узла;
- необходимость перезагрузки узла после применения обновлений или обслуживания технических средств;
- перемещение ВМ с узла с низкой производительностью на высокопроизводительный узел.

Есть два механизма миграции:

- онлайн-миграция (Live Migration);
- офлайн-миграция.

Примечание. Миграция контейнеров без перезапуска в настоящее время не поддерживается. При выполнении миграции запущенного контейнера, контейнер будет выключен, перемещен, а затем снова запущен на целевом узле. Поскольку контейнеры легковесные, то это обычно приводит к простою в несколько сотен миллисекунд.

Для возможности онлайн-миграции ВМ должны выполняться следующие условия:

- у ВМ нет локальных ресурсов;
- хосты находятся в одном кластере PVE;
- между хостами имеется надежное сетевое соединение;
- на целевом хосте установлены такие же или более высокие версии пакетов PVE.

Миграция в реальном времени обеспечивает минимальное время простоя ВМ, но, в то же время занимает больше времени. При миграции в реальном времени (без выключения питания) процесс должен скопировать все содержимое оперативной памяти ВМ на новый узел. Чем больше объем выделенной ВМ памяти, тем дольше будет происходить ее перенос.

Если образ виртуального диска ВМ хранится в локальном хранилище узла PVE миграция в реальном времени не возможна. В этом случае ВМ должна быть перед миграцией выключена. В процессе миграции ВМ, хранящейся локально, PVE скопирует виртуальный диск на узел получателя с применением `rsync`.

Запустить процесс миграции можно как в графическом интерфейсе PVE, так в интерфейсе командной строки.

4.8.1 Миграция с применением графического интерфейса

Для миграции ВМ или контейнера необходимо выполнить следующие шаги:

1) выбрать ВМ или контейнер для миграции и нажать кнопку «Миграция» («Migrate») (Рис. 156);

2) в открывшемся диалоговом окне (Рис. 157) выбрать узел назначения, на который будет осуществляться миграция, и нажать кнопку «Миграция» («Migrate»).

Выбор ВМ или контейнера для миграции

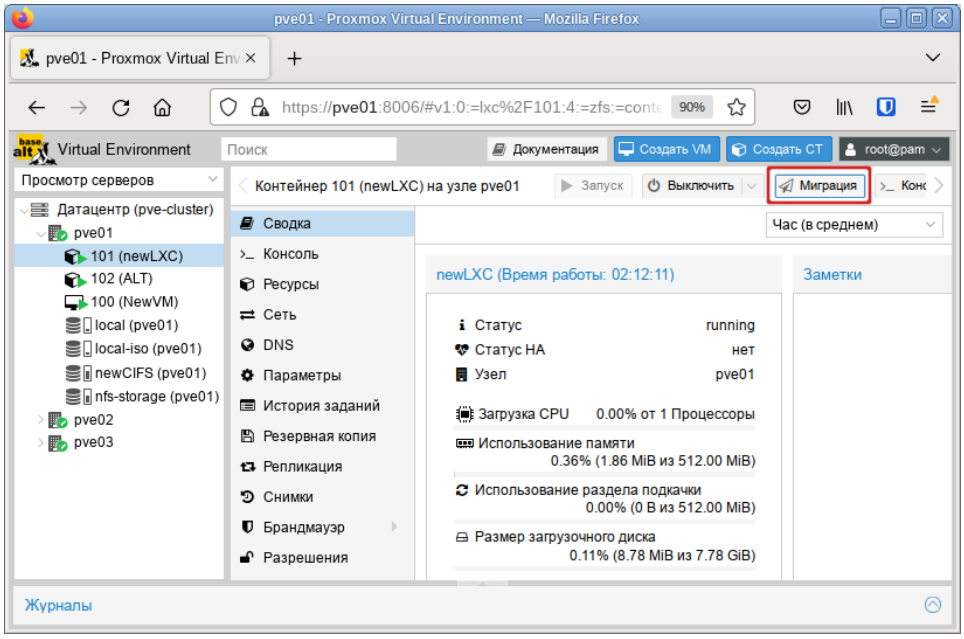


Рис. 156

Примечание. Режим миграции будет выбран автоматически (Рис. 157, Рис. 158, Рис. 159) в зависимости от состояния ВМ/контейнера (запущен/остановлен).

Миграция контейнера с перезапуском

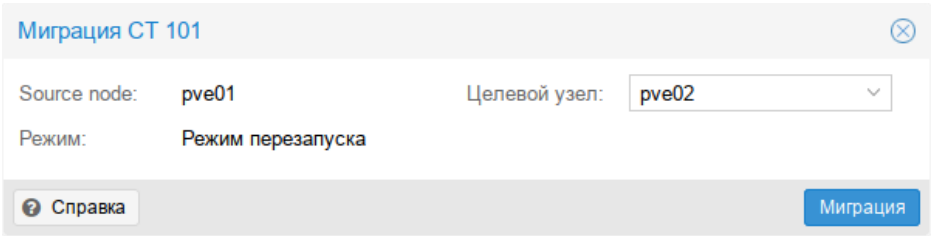


Рис. 157

Миграция ВМ Онлайн

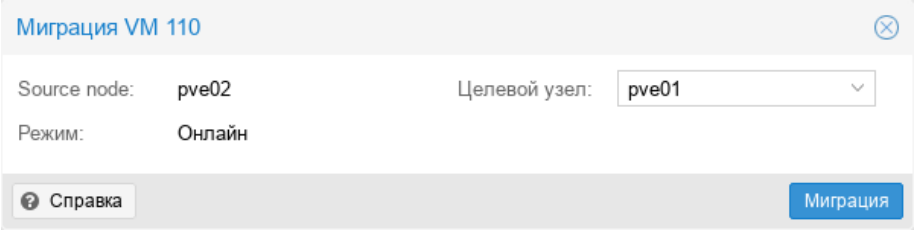


Рис. 158

Миграция ВМ Офлайн

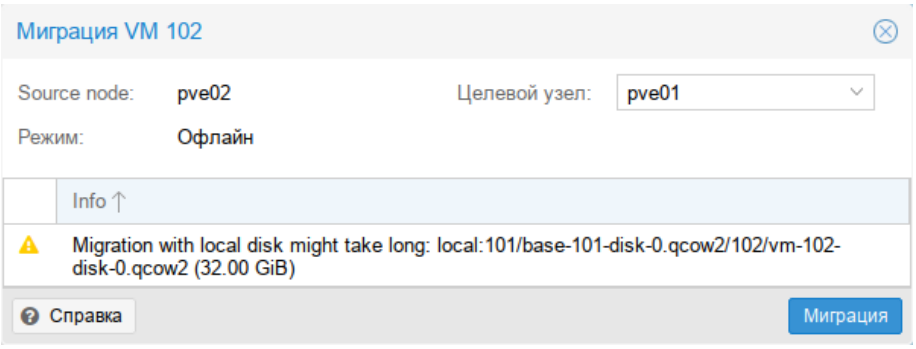


Рис. 159

4.8.2 Миграция с применением командной строки

Чтобы осуществить миграцию ВМ необходимо выполнить следующую команду:

```
# qm migrate <vmid> <target> [OPTIONS]
```

Для осуществления миграции ВМ в реальном времени следует использовать метр `--online`.

Чтобы осуществить миграцию контейнера необходимо выполнить следующую команду:

```
# pct migrate <ctid> <target> [OPTIONS]
```

Поскольку миграция контейнеров в реальном времени не возможна, можно выполнить миграцию работающего контейнера с перезапуском, добавив параметр `--restart`. Например:

```
# pct migrate 101 pve02 –restart
```

4.8.3 Миграция ВМ из внешнего гипервизора

Экспорт ВМ из внешнего гипервизора обычно заключается в переносе одного или нескольких образов дисков с файлом конфигурации, описывающим настройки ВМ (ОЗУ, количество ядер). Образы дисков могут быть в формате vmdk (VMware или VirtualBox), или qcow2 (KVM). Наиболее популярным форматом конфигурации для экспорта ВМ является стандарт OVF.

Примечание. Для ВМ Windows необходимо также установить паравиртуализированные драйверы Windows.

4.8.3.1 Миграция KVM ВМ в PVE

В данном разделе рассмотрен процесс миграции ВМ из OpenNebula в PVE.

Выключить ВМ на хосте источнике. Найти путь до образа жесткого диска, который используется в ВМ (в данной команде 14 – id образа диска ВМ):

```
$ oneimage show 14
IMAGE 14 INFORMATION
ID             : 14
NAME           : ALT Linux p9
USER           : oneadmin
GROUP          : oneadmin
LOCK           : None
DATASTORE      : default
TYPE           : OS
REGISTER TIME  : 04/30 11:00:42
PERSISTENT     : Yes
SOURCE         : /var/lib/one//datastores/1/f811a893808a9d8f5bf1c029b3c7e905
FSTYPE         : save_as
SIZE           : 12G
STATE          : used
RUNNING_VMS    : 1

PERMISSIONS
OWNER          : um-
GROUP          : ---
OTHER          : ---

IMAGE TEMPLATE
DEV_PREFIX="vd"
DRIVER="qcow2"
SAVED_DISK_ID="0"
SAVED_IMAGE_ID="7"
SAVED_VM_ID="46"
SAVE_AS_HOT="YES"
```

где `/var/lib/one//datastores/1/f811a893808a9d8f5bf1c029b3c7e905` – адрес образа жёсткого диска ВМ.

Скопировать данный образ на хост назначения с PVE.

Примечание. В OpenNebula любой диск ВМ можно экспортировать в новый образ (если ВМ находится в состояниях RUNNING, POWEROFF или SUSPENDED):

```
$ onevm disk-saveas <vmid> <diskid> <img_name>  [--type type --snapshot snapshot]
```

где `--type <type>` – тип нового образа (по умолчанию raw); `--snapshot <snapshot_id>` – снимок диска, который будет использован в качестве источника нового образа (по умолчанию текущее состояние диска).

Экспорт диска ВМ:

```
$ onevm disk-saveas 125 0 test.qcow2
Image ID: 44
```

Инфомация об образе диска ВМ:

```
$ oneimage show 44
MAGE 44 INFORMATION
ID             : 44
NAME           : test.qcow2
USER           : oneadmin
GROUP          : oneadmin
LOCK           : None
DATASTORE      : default
TYPE           : OS
REGISTER TIME  : 07/12 21:34:42
PERSISTENT     : No
SOURCE         : /var/lib/one//datastores/1/9d6336a88d6ab62ea1dce65d81e55881
FSTYPE         : save_as
SIZE           : 12G
STATE          : rdy
RUNNING_VMS    : 0

PERMISSIONS
OWNER          : um-
GROUP          : ---
OTHER          : ---

IMAGE TEMPLATE
DEV_PREFIX="vd"
DRIVER="qcow2"
SAVED_DISK_ID="0"
SAVED_IMAGE_ID="14"
SAVED_VM_ID="125"
SAVE_AS_HOT="YES"

VIRTUAL MACHINES
```

Информация о диске:

```
$ qemu-img info /var/lib/one//datastores/1/9d6336a88d6ab62ea1dce65d81e55881
image: /var/lib/one//datastores/1/9d6336a88d6ab62ea1dce65d81e55881
file format: qcow2
virtual size: 12 GiB (12884901888 bytes)
disk size: 3.52 GiB
```

```
cluster_size: 65536
Format specific information:
    compat: 1.1
    compression type: zlib
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
    extended l2: false
```

На хосте назначения подключить образ диска к ВМ (рассмотрено подключение на основе Directory Storage), выполнив следующие действия:

1) создать новую ВМ в веб-интерфейсе PVE или командой:

```
# qm create 120 --bootdisk scsi0 --net0 virtio,bridge=vmbr0 --scsihw virtio-scsi-pci
```

2) чтобы использовать в PVE образ диска в формате qcow2 (полученный из другой системы KVM, либо преобразованный из другого формата), его необходимо импортировать. Команда импорта:

```
# qm importdisk <vmid> <source> <storage> [OPTIONS]
```

Команда импорта диска f811a893808a9d8f5bf1c029b3c7e905 в хранилище local, для ВМ с ID 120 (подразумевается, что образ импортируемого диска находится в каталоге, из которого происходит выполнение команды):

```
# qm importdisk 120 f811a893808a9d8f5bf1c029b3c7e905 local --format qcow2
importing disk 'f811a893808a9d8f5bf1c029b3c7e905' to VM 120 ...
…
Successfully imported disk as 'unused0:local:120/vm-120-disk-0.qcow2'
```

3) привязать диск к ВМ:

- в веб-интерфейсе PVE: перейти на вкладку «Оборудование», созданной ВМ. В списке устройств будет показан неиспользуемый жесткий диск, выбрать его, выбрать режим «SCSI» и нажать кнопку «Добавить» (Рис. 160).
- в командной строке:

```
# qm set 120 --scsi0 local:120/vm-120-disk-0.qcow2
update VM 120: -scsi0 local:120/vm-120-disk-0.qcow2
```

4) донастроить параметры процессора, памяти, сетевых интерфейсов, порядок загрузки;

5) включить ВМ.

Добавление диска к ВМ

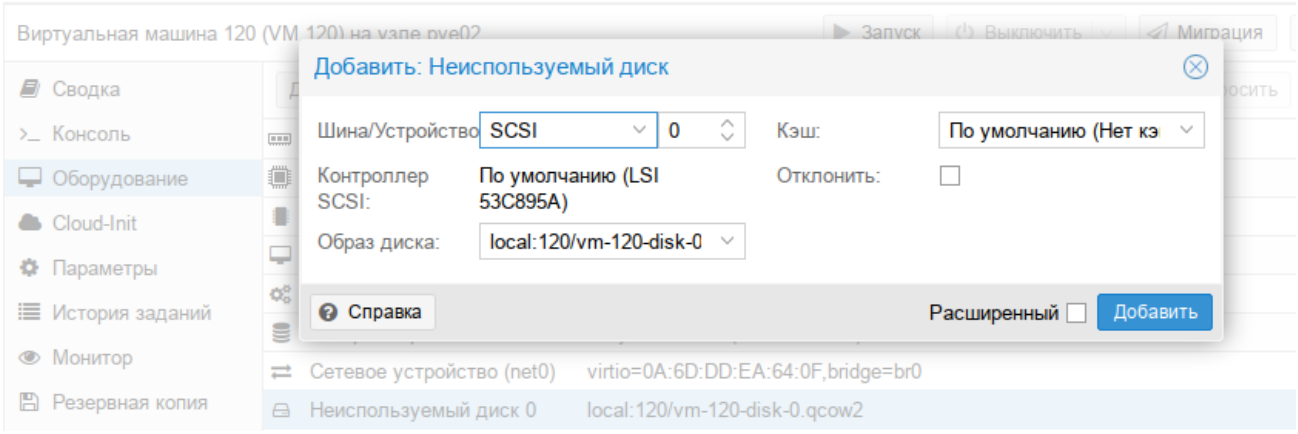


Рис. 160

4.8.3.2 *Миграция ВМ из VMware в PVE*

Экспорт ВМ из внешнего гипервизора обычно заключается в переносе одного или нескольких образов дисков с файлом конфигурации, описывающим настройки ВМ (ОЗУ, количество ядер). Образы дисков могут быть в формате vmdk (VMware или VirtualBox), или qcow2 (KVM).

В данном разделе рассмотрена миграция ВМ из VMware в PVE, на примере ВМ с ОС Windows 7.

Подготовить ОС Windows. ОС Windows должна загружаться с дисков в режиме IDE.

Подготовить образ диска. Необходимо преобразовать образ диска в тип `single growable virtual disk`. Сделать это можно с помощью утилиты `vmware-vdiskmanager` (поставляется в комплекте с VMWare Workstation). Для преобразования образа перейти в папку с образами дисков и выполнить команду:

```
"C:\Program Files\VMware\VMware Server\vmware-vdiskmanager"
-r win7.vmdk -t 0 win7-pve.vmdk
```

где win7.vmdk – файл с образом диска.

Подключить образ диска к ВМ одним из трёх указанных способов:

1) подключение образа диска к ВМ на основе Directory Storage:

- в веб-интерфейсе PVE создать ВМ KVM;
- скопировать преобразованный образ win7-pve.vmdk в каталог с образами ВМ /var/lib/vz/images/VMID, где VMID – VMID, созданной виртуальной машины (можно воспользоваться WinSCP);
- преобразовать файл win7-pve.vmdk в qemu формат:

```
# qemu-img convert -f vmdk win7-pve.vmdk -O qcow2 win7-pve.qcow2
```

- добавить в конфигурационный файл ВМ (`/etc/pve/nodes/pve02/qemu-server/VMID.conf`) строку:

```
unused0: local:100/win7-pve.qcow2
```

где 100 – VMID, а local – хранилище в PVE.

– перейти в веб-интерфейсе PVE на вкладку «Оборудование», созданной ВМ. В списке устройств будет показан неиспользуемый жесткий диск, выбрать его, выбрать режим IDE и нажать кнопку «Добавить» (*Рис. 161*).

Добавление диска к ВМ

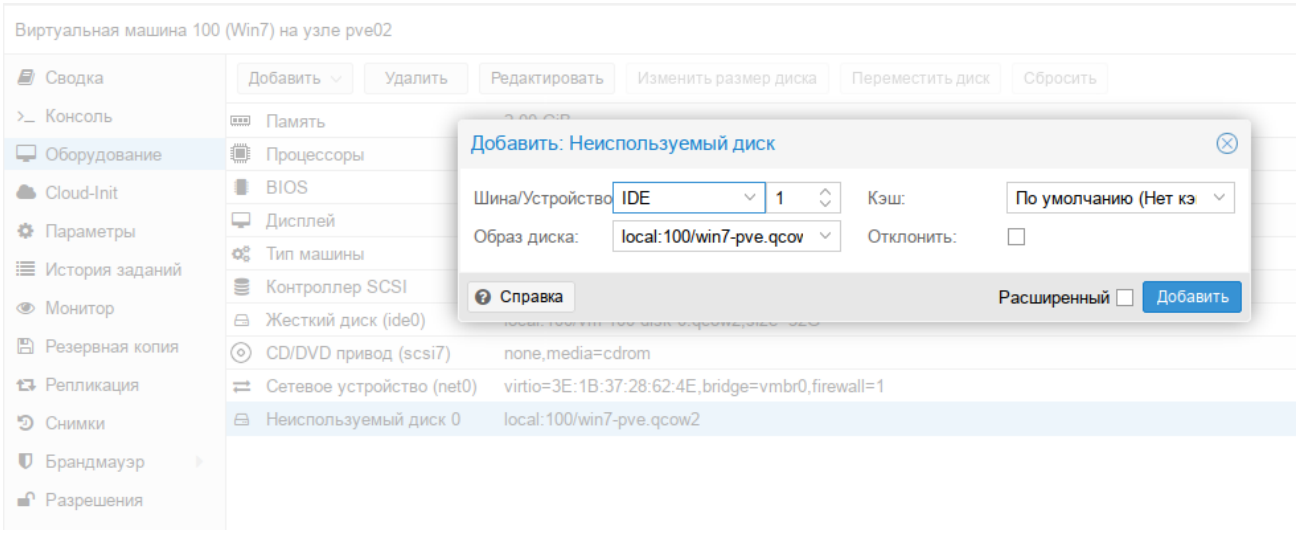


Рис. 161

2) подключение образа диска к ВМ на основе LVM Storage:

– в веб-интерфейсе PVE создать ВМ с диском большего размера, чем диск в образе vmdk. Посмотреть размер диска в образе можно командой:

```
# qemu-img info win7-pve.vmdk
image: win7-pve.vmdk
file format: vmdk
virtual size: 127G (136365211648 bytes)
disk size: 20.7 GiB
cluster_size: 65536
Format specific information:
   cid: 3274246794
   parent cid: 4294967295
   create type: streamOptimized
   extents:
       [0]:
       compressed: true
       virtual size: 136365211648
       filename: win7-pve.vmdk
```

```
cluster size: 65536
format:
```

В данном случае необходимо создать диск в режиме IDE размером не меньше 127GB.

- скопировать преобразованный образ win7-pve.vmdk в каталог с образами ВМ `/var/lib/vz/images/VMID`, где VMID – VMID, созданной ВМ (можно воспользоваться WinSCP);
- перейти в консоль ноды кластера и посмотреть, как называется LVM диск созданной ВМ (диск должен быть в статусе ACTIVE):

```
# lvscan
ACTIVE            '/dev/sharedsv/vm-101-disk-1' [130,00 GiB] inherit
```

- сконвертировать образ vdmk в raw формат непосредственно на LVM-устройство:

```
# qemu-img convert -f vmdk win7-pve.vmdk -O raw /dev/sharedsv/vm-101-disk-1
```

3) подключение образа диска к ВМ на основе CEPH Storage:

- в веб-интерфейсе PVE создать ВМ с диском большего размера, чем диск в образе vmdk. Посмотреть размер диска в образе можно командой:

```
# qemu-img info win7-pve.vmdk
```

- скопировать преобразованный образ win7-pve.vmdk в каталог с образами ВМ `/var/lib/vz/images/VMID`, где VMID – VMID, созданной виртуальной машины;
- перейти в консоль ноды кластера. Отобразить образ из пула CEPH в локальное блочное устройство:

```
# rbd map rbd01/vm-100-disk-1
/dev/rbd0
```

Примечание. Имя нужного пула можно посмотреть на вкладке «Датацентр» → «Хранилище» → «rbd-storage».

- сконвертировать образ vdmk в raw формат непосредственно на отображенное устройство:

```
# qemu-img convert -f vmdk win7-pve.vmdk -O raw /dev/rbd0
```

Адаптация новой ВМ:

1) проверить режим работы жесткого диска: для Windows – IDE, для Linux – SCSI.

2) установить режим VIRTIO для жесткого диска (режим VIRTIO также доступен для Windows, но сразу загрузиться в этом режиме система не может):

- загрузиться в режиме IDE и выключить машину. Добавить еще один диск в режиме VIRTIO и включить машину. Windows установит нужные драйвера;
- выключить машину;

- изменить режим основного диска с IDE на VIRTIO;
- загрузить систему, которая должна применить VIRTIO драйвер и выдать сообщение, что драйвер от RedHat.

3) включить ВМ. Первое включение займет какое-то время (будут загружены необходимые драйвера).

4.8.3.3 Пример импорта Windows OVF

Скопировать файлы ovf и vmdk на хост PVE. Создать новую ВМ, используя имя ВМ, информацию о ЦП и памяти из файла конфигурации OVF, и импортировать диски в хранилище local-lvm:

```
# qm importovf 999 WinDev2212Eval.ovf local-lvm
```

Примечание. Сеть необходимо настроить вручную.

4.9 Клонирование виртуальных машин

Простой способ развернуть множество ВМ одного типа – создать клон существующей ВМ.

Существует два вида клонов:

- Полный клон – результатом такой копии является независимая ВМ. Новая ВМ не имеет общих ресурсов с оригинальной ВМ. При таком клонировании можно выбрать целевое хранилище, поэтому его можно использовать для переноса ВМ в совершенно другое хранилище. Также при создании клона можно изменить формат образа диска, если драйвер хранилища поддерживает несколько форматов.
- Связанный клон – такой клон является перезаписываемой копией, исходное содержимое которой совпадает с исходными данными. Создание связанного клона происходит практически мгновенно и изначально не требует дополнительного места. Клоны называются связанными, потому что новый образ диска ссылается на оригинал. Немодифицированные блоки данных считываются из исходного образа, а изменения записываются (и затем считываются) из нового местоположения (исходный образ при этом должен работать в режиме только для чтения). С помощью PVE можно преобразовать любую ВМ в шаблон (см. ниже). Такие шаблоны впоследствии могут быть использованы для эффективного создания связанных клонов. При создании связанных клонов невозможно изменить целевое хранилище.

Примечание. При создании полного клона необходимо прочитать и скопировать все данные образа ВМ. Это обычно намного медленнее, чем создание связанного клона.

Весь функционал клонирования доступен в веб-интерфейсе PVE.

Для клонирования ВМ необходимо выполнить следующие шаги:

1) создать ВМ с необходимыми настройками (все создаваемые из такой ВМ клоны будут иметь идентичные настройки) или воспользоваться уже существующей ВМ;

2) в контекстном меню ВМ выбрать пункт «Клонировать» («Clone») (Рис. 162);

3) откроется диалоговое окно (Рис. 163), со следующими полями:

– «Целевой узел» («Target node») – узел получатель клонируемой ВМ (для создания новой ВМ на другом узле необходимо чтобы ВМ находилась в общем хранилище и это хранилище должно быть доступно на целевом узле);
– «VM ID» – идентификатор ВМ;
– «Имя» («Name») – название ВМ;
– «Пул ресурсов»(«Resource Pool») – пул, к которому будет относиться ВМ;
– «Режим» («Mode») – метод клонирования (если клонирование происходит из шаблона ВМ). Доступны значения: «Полное клонирование» («Full Clone») и «Связанная копия» («Linked Clone»);
– «Снимок» («Snapshot») – снимок из которого будет создаваться клон (если снимки существуют);
– «Целевое хранилище» («Target Storage») – хранилище для клонируемых виртуальных дисков;
– «Формат» («Format») – формат образа виртуального диска.

4) для запуска процесса клонирования необходимо нажать кнопку «Клонировать».

Контекстное меню ВМ

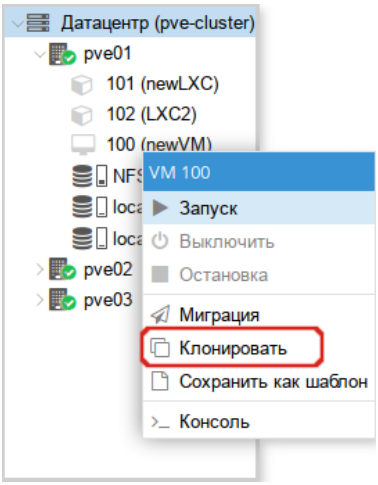


Рис. 162

Настройки клонирования

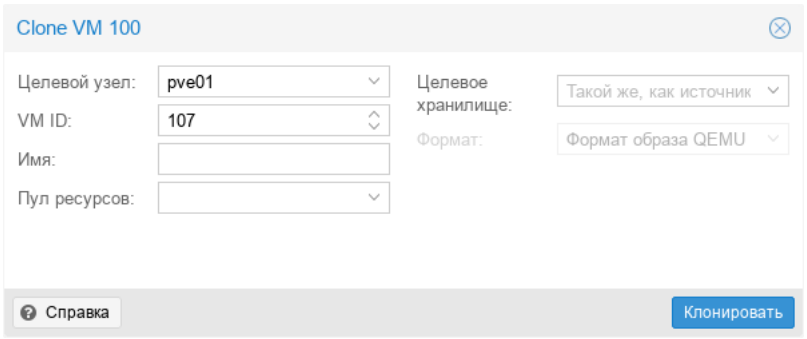


Рис. 163

Некоторые типы хранилищ позволяют копировать определенный снимок ВМ (Рис. 164), который по умолчанию соответствует текущим данным ВМ. Клон ВМ никогда не содержит дополнительных снимков оригинальной ВМ.

Выбор снимка для клонирования

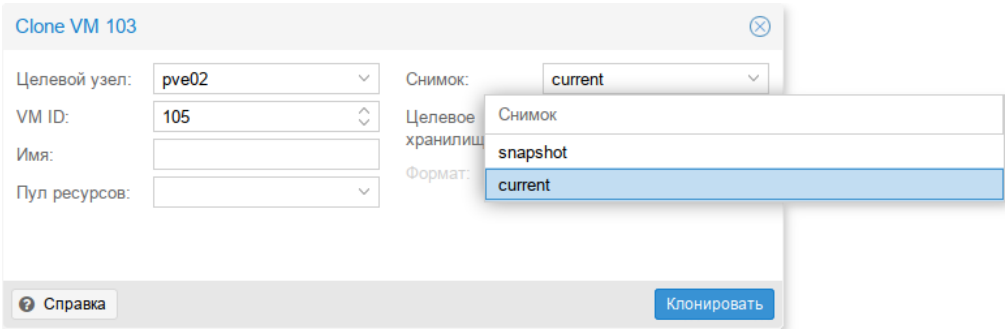


Рис. 164

ВМ можно преобразовать в шаблон. Такие шаблоны доступны только для чтения, и их можно использовать для создания связанных клонов.

Для преобразования ВМ в шаблон необходимо в контекстном меню ВМ выбрать пункт «Сохранить как шаблон» («Convert to template») и в ответ на запрос на подтверждения, нажать кнопку «Да».

Примечание. Запустить шаблоны невозможно, так как это приведет к изменению образов дисков. Если необходимо изменить шаблон, следует создать связанный клон и изменить его.

4.10 Резервное копирование (backup)

PVE предоставляет полностью интегрированное решение, использующее возможности всех хранилищ и всех типов гостевых систем.

Резервные копии PVE представляют собой полные резервные копии – они содержат конфигурацию ВМ/CT и все данные. Резервное копирование может быть запущено через графический интерфейс или с помощью утилиты командной строки `vzdump`.

4.10.1 Алгоритмы резервного копирования

Инструментарий для создания резервных копий PVE поддерживает следующие механизмы сжатия:

- сжатие LZO – алгоритм сжатия данных без потерь (реализуется в PVE утилитой `lzop`). Особенностью этого алгоритма является скоростная распаковка. Следовательно, любая резервная копия, созданная с помощью этого алгоритма, может при необходимости быть развернута за минимальное время.
- сжатие GZIP – при использовании этого алгоритма резервная копия будет «на лету» сжиматься утилитой GNU Zip, использующей мощный алгоритм Deflate. Упор делается на максимальное сжатие данных, что позволяет сократить место на диске, занимаемое резервными копиями. Главным отличием от LZO является то, что процедуры компрессии/декомпрессии занимают достаточно большое количество времени.
- сжатие Zstandard (zstd) – алгоритм сжатия данных без потерь. В настоящее время Zstandard является самым быстрым из этих трех алгоритмов. Многопоточность – еще одно преимущество zstd перед lzo и gzip.

4.10.2 Режимы резервного копирования

Режимы резервного копирования для ВМ:

- режим остановки (Stop) – обеспечивает самую высокую надежность резервного копирования, но требует полного выключения ВМ. В этом режиме ВМ отправляется команда на штатное выключение, после остановки выполняется резервное копирование и затем отдается команда на включение ВМ. Количество ошибок при таком подходе минимально и чаще всего сводится к нулю;
- режим ожидания (Suspend) – ВМ временно «замораживает» свое состояние, до окончания процесса резервного копирования. Содержимое оперативной памяти не стирается, что позволяет продолжить работу ровно с той точки, на которой работа была приостановлена. Сервер простаивает во время копирования информации, но при этом нет необходимости выключения/включения ВМ, что достаточно критично для некоторых сервисов;
- режим снимка (Snapshot) – обеспечивает мимнимальное время простоя ВМ (использование этого механизма не прерывает работу ВМ), но имеет два очень серьезных недостатка – могут возникать проблемы из-за блокировок файлов операционной системой и самая низкая скорость создания. Резервные копии, созданные этим методом, надо всегда проверять в тестовой среде.

Режимы резервного копирования для контейнеров:

- режим остановки (Stop) – остановка контейнера на время резервного копирования. Это может привести к длительному простою;
- режим ожидания (Suspend) – этот режим использует `rsync` для копирования данных контейнера во временную папку (опция `--tmpdir`). Затем контейнер приостанавливается и `rsync` копирует измененные файлы. После этого контейнер возобновляет свою работу. Это приводит к минимальному времени простоя, но требует дополнительное пространство для хранения копии контейнера. Когда контейнер находится в локальной файловой системе и хранилищем резервной копии является сервер NFS, необходимо установить `--tmpdir` также и на локальную файловую систему, так как это приведет к повышению производительности. Использование локального `tmpdir` также необходимо, если требуется сделать резервную копию локального контейнера с использованием списков контроля доступа (ACL) в режиме ожидания, если хранилище резервных копий – сервер NFS.
- режим снимка (Snapshot) – этот режим использует возможности мгновенных снимков основного хранилища. Сначала, контейнер будет приостановлен для обеспечения согласованности данных, будет сделан временный снимок томов контейнера, а содержимое снимка будет заархивировано в tar-файле, далее временный снимок удаляется. Для возможности использования этого режима необходимо, чтобы тома резервных копий находились в хранилищах, поддерживающих моментальные снимки.

4.10.3 Резервное хранилище

Перед тем, как настроить резервное копирование, необходимо определить хранилище резервных копий. Хранилище резервных копий должно быть хранилищем уровня файлов, так как резервные копии хранятся в виде обычных файлов. В большинстве случаев можно использовать сервер NFS для хранения резервных копий. Если хранилище будет использоваться только для резервных копий, следует выставить соответствующие настройки (Рис. 165).

Настройка хранилища NFS

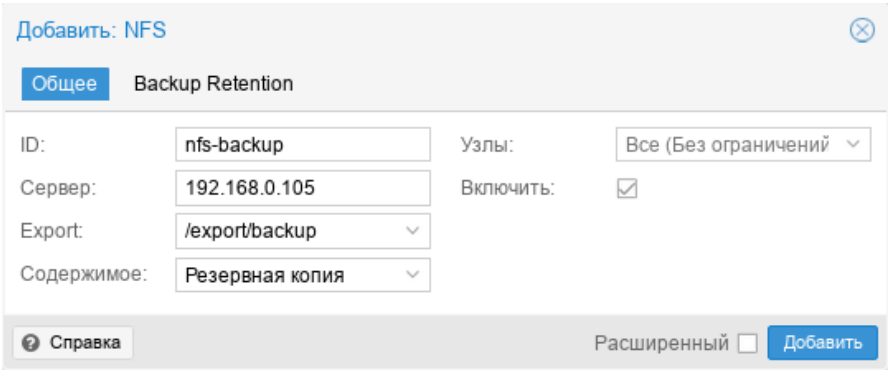


Рис. 165

На вкладке «Backup Retention» можно указать параметры хранения резервных копий (*Рис. 166*).

Параметры хранения резервных копий в хранилище NFS

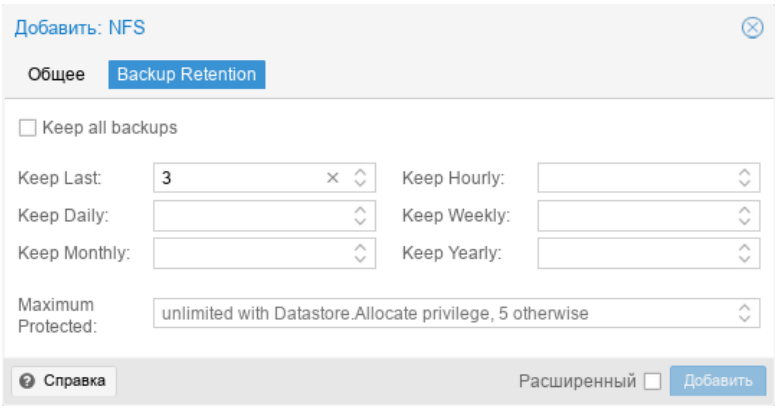


Рис. 166

Доступны следующие варианты хранения резервных копий (в скобках указаны параметры опции prune-backups команды `vzdump`):

- «Keep all backups» (`keep-all=<1|0>`) – хранить все резервные копии (если отмечен этот пункт, другие параметры не могут быть установлены);
- «Keep Last» (`keep-last=<N>`) – хранить <N> последних резервных копий;
- «Keep Hourly» (`keep-hourly=<N>`) – хранить резервные копии за последние <N> часов (если за один час создается более одной резервной копии, сохраняется только последняя);
- «Keep Daily» (`keep-daily=<N>`) – хранить резервные копии за последние <N> дней (если за один день создается более одной резервной копии, сохраняется только самая последняя);
- «Keep Weekly» (`keep-weekly=<N>`) – хранить резервные копии за последние <N> недель (если за одну неделю создается более одной резервной копии, сохраняется только последняя);
- «Keep Monthly» (`keep-monthly=<N>`) – хранить резервные копии за последние <N> месяцев (если за один месяц создается более одной резервной копии, сохраняется только самая последняя);
- «Keep Yearly» (`keep-yearly <N>`) – хранить резервные копии за последние <N> лет (если за один год создается более одной резервной копии, сохраняется только самая последняя).

Варианты хранения обрабатываются в указанном выше порядке. Каждый вариант распространяется только на резервное копирование в определенный период времени.

Пример указания параметров хранения резервных копий при создании задания:

```
# vzdump 777 --prune-backups keep-last=3,keep-daily=13,keep-yearly=9
```

Несмотря на то что можно передавать параметры хранения резервных копий непосредственно в при создании задания, рекомендуется настроить эти параметры на уровне хранилища.

4.10.4 Резервное копирование по расписанию

Задания для резервного копирования можно запланировать так, чтобы они выполнялись автоматически в определенные дни и часы для конкретных узлов и гостевых систем. Конфигурирование заданий для создания резервных копий выполняется на уровне центра обработки данных в веб-интерфейсе, при этом будет создана запись cron в `/etc/cron.d/vzdump`.

4.10.5 Формат расписания

Для настройки расписания используются события календаря системного времени (см. `man 7 systemd.time`).

Используется следующий формат:

```
[WEEKDAY] [[YEARS-]MONTHS-DAYS] [HOURS:MINUTES[:SECONDS]]
```

WEEKDAY – дни недели, указанные в трёх буквенном варианте на английском: mon,tue,wed,thu,fri,sat и sun. Можно использовать несколько дней в виде списка, разделённого запятыми. Можно задать диапазон дней, указав день начала и окончания, разделённые двумя точками («..»), например mon..fri. Форматы можно смешивать. Если опущено, подразумевается «*».

Формат времени – время указывается в виде списка интервалов часов и минут. Часы и минуты разделяются знаком «:». И часы, и минуты могут быть списком и диапазонами значений в том же формате, что и дни недели. Можно не указывать часы, если они не нужны. В этом случае подразумевается «*». Допустимый диапазон значений: 0–23 для часов и 0–59 для минут.

Специальные занчения приведены в табл. 8. В таблице 9 приведены примеры периодов времени.

Т а б л и ц а 8 – **Специальные значения**

Расписание	Значение	Синтаксис
minutely	Каждую минуту	*-*-* *:*:00
hourly	Каждый час	*-*-* *:00:00
daily	Раз в день	*-*-* 00:00:00
weekly	Раз в неделю	mon *-*-* 00:00:00
monthly	Раз в месяц	*-*-01 00:00:00
yearly или annually	Раз в год	*-01-01 00:00:00
quarterly	Раз в квартал	*-01,04,07,10-01 00:00:00

Расписание	Значение	Синтаксис
semiannually или semi-annually	Раз в полгода	*-01,07-01 00:00:00

Т а б л и ц а 9 – **Примеры**

Расписание	Эквивалент	Значение
mon,tue,wed,thu,fri	mon..fri	Каждый будний день в 00:00
sat,sun	sat..sun	В субботу и воскресенье в 00:00
mon,wed,fri	-	В понедельник, среду и пятницу в 00:00
12:05	12:05	Каждый день в 12:05
*/5	0/5	Каждые пять минут
mon..wed 30/10	mon,tue,wed 30/10	В понедельник, среду и пятницу в 30, 40 и 50 минут каждого часа
mon..fri 8..17,22:0/15	-	Каждые 15 минут с 8 часов до 18 и с 22 до 23 в будний день
fri 12..13:5/20	fri 12,13:5/20	В пятницу в 12:05, 12:25, 12:45, 13:05, 13:25 и 13:45
12,14,16,18,20,22:5	12/2:5	Каждые два часа каждый день с 12:05 до 22:05
*	*/1	Ежеминутно (минимальный интервал)
*-05	-	Пятого числа каждого месяца
Sat *-1..7 15:00	-	Первую субботу каждого месяца в 15:00
2023-10-22	-	22 октября 2023 года в 00:00

4.10.6 Настройка резервного копирования в графическом интерфейсе

Для того чтобы создать расписание резервного копирования, необходимо перейти во вкладку «Резервная копия» («Backup») датацентра (Рис. 167) и нажать кнопку «Добавить».

Вкладка «Резервная копия»

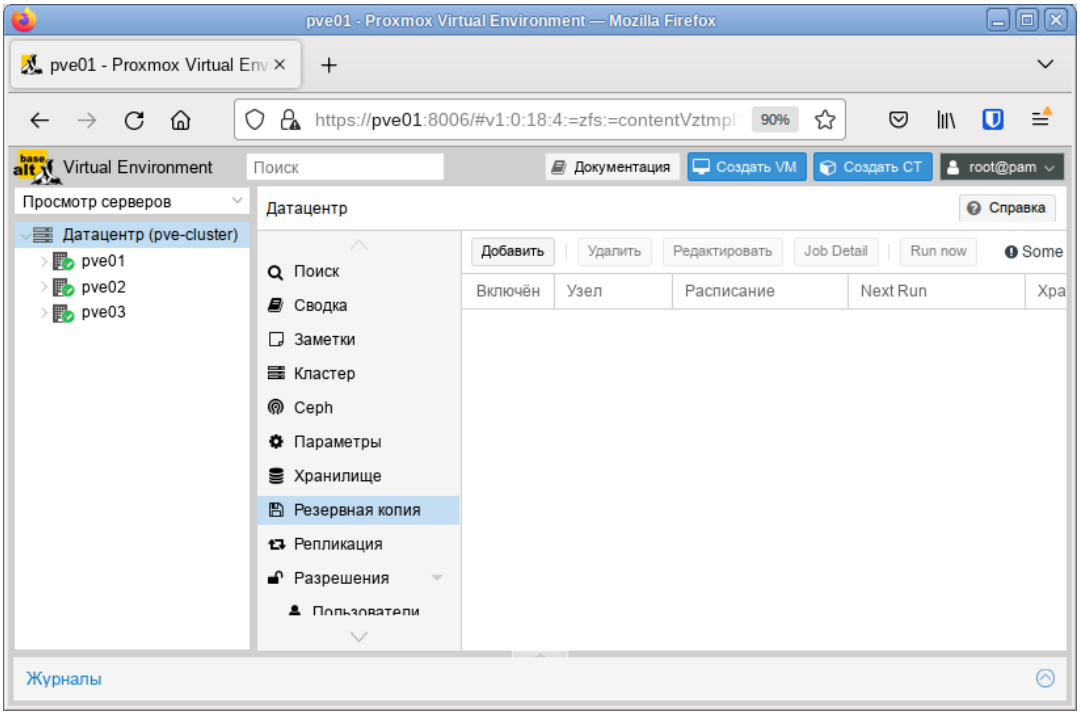


Рис. 167

При создании задания на резервирование, необходимо указать (Рис. 168):

– «Узел» («Node») – можно создавать график из одного места по разным узлам (серверам);
– «Хранилище» («Storage») – точка смонтированного накопителя, куда будет проходить копирование;
– «Расписание» («Schedule») – здесь можно указать расписание резервного копирования. Можно выбрать период из списка (*Рис. 169*) или указать вручную;
– «Режим выбора» («Selection mode») – возможные значения: «Учитывая выбранные VM» («Include selected VMS»), «Все» («All»), «Исключить выбранные VM» («Exclude selected VMS»), «Pool based»;
– «Отправить письмо» («Send email to») – адрес, на который будут приходить отчёты о выполнении резервного копирования;
– «Email» – принимает два значения: «Всегда» («Notify always») – сообщение будет приходить при любом результате резервного копирования, «Только при ошибках» («On failure only») – сообщение будет приходить только в случае неудачной попытки резервного копирования;
– «Сжатие» («Compression») – метод сжатия, принимает четыре значения: «ZSTD (быстро и хорошо)» (по умолчанию), «LZO (быстро)», «GZIP (хорошо)» и «нет»;
– «Режим» («Mode») – режим ВМ, в котором будет выполняться резервное копирование. Может принимать три значения (Рис. 170): «Снимок», «Приостановить», «Остановка».

Создание задания для резервного копирования. Вкладка «Общее»

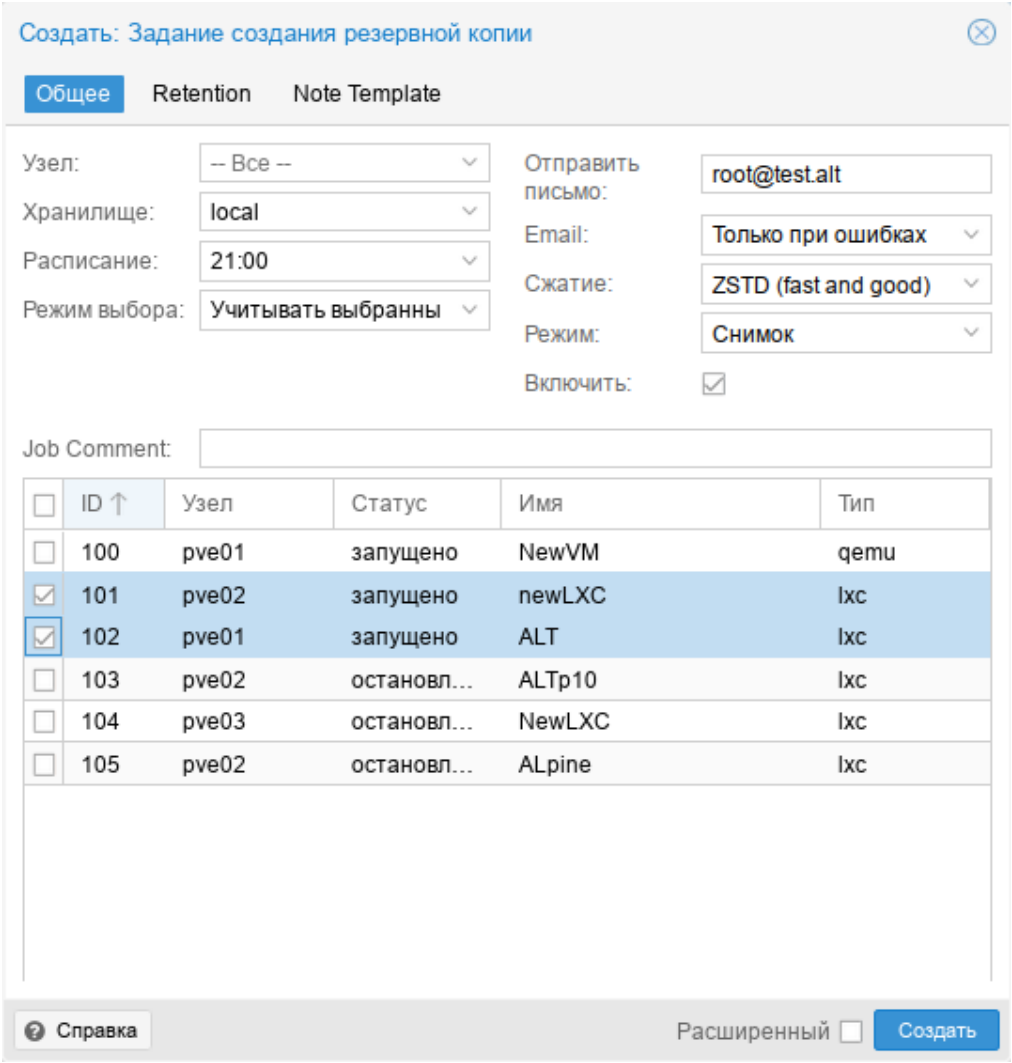


Рис. 168

Выбор периода

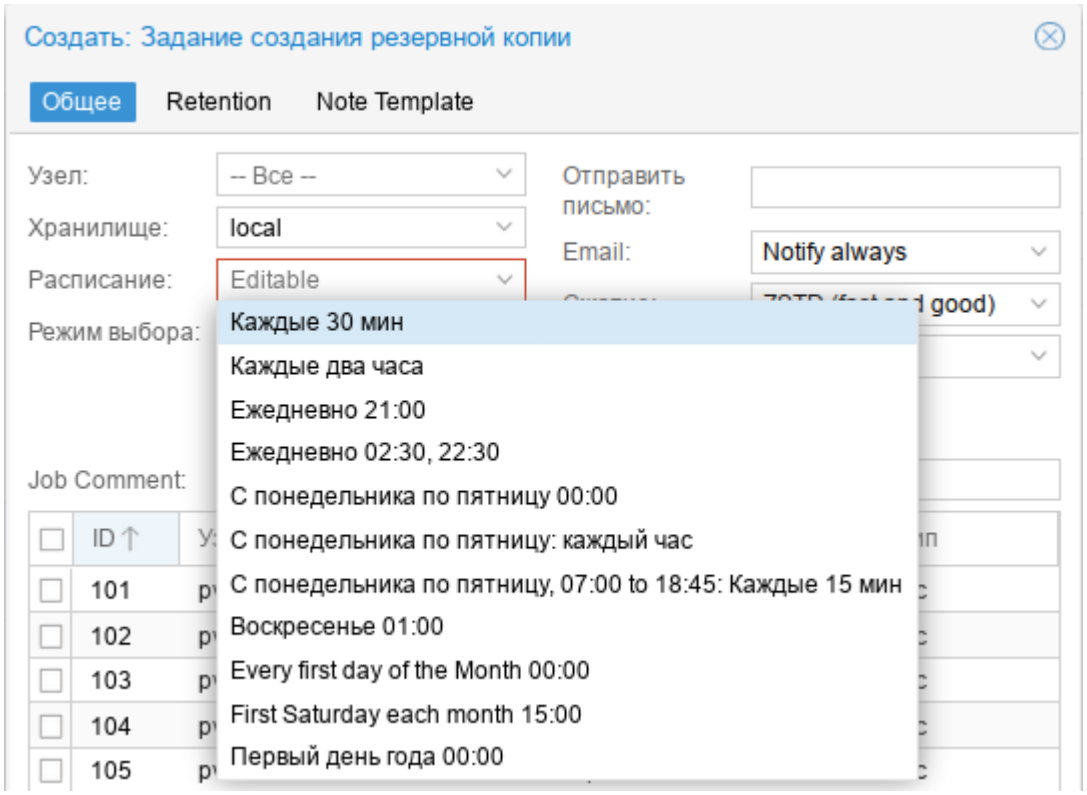


Рис. 169

Выбор режима создания резервной копии

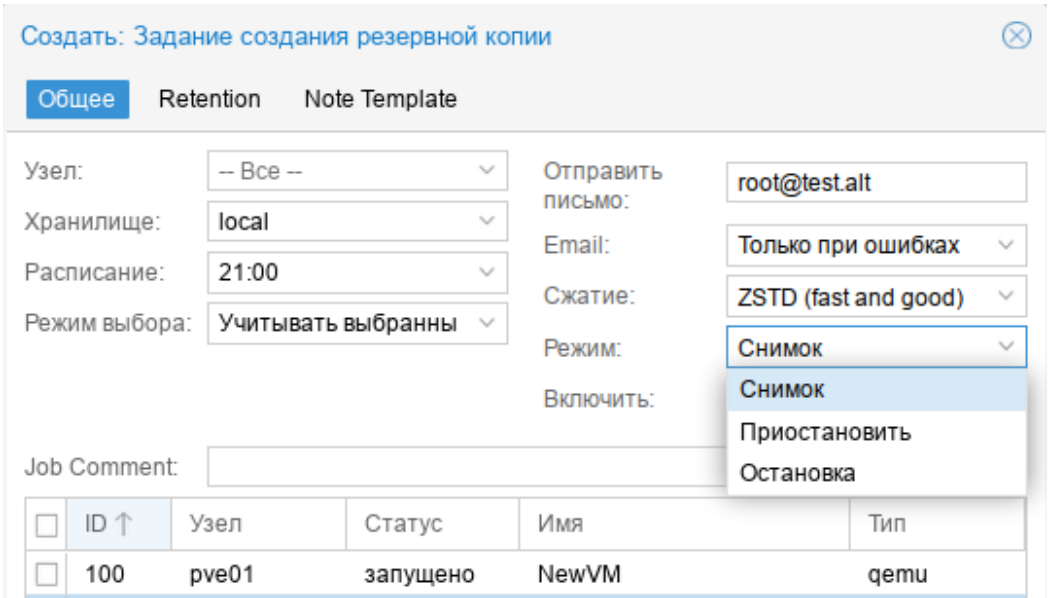


Рис. 170

На вкладке «Retention» можно настроить параметры хранения резервных копий (*Рис. 171*).

Создание задания для резервного копирования. Вкладка «Retention»

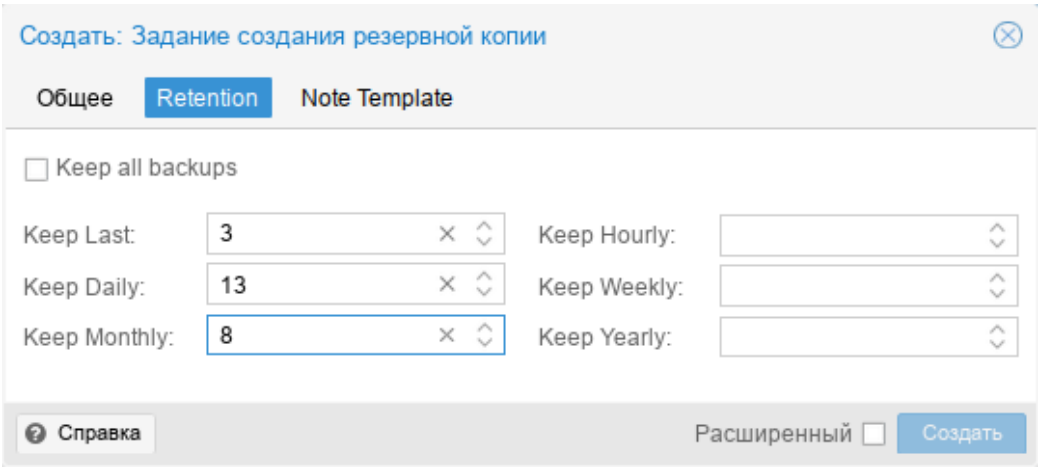


Рис. 171

На вкладке «Шаблоны заметок» («Note Templates») можно настроить примечание, которое будет добавляться к резервным копиям. Строка примечания может содержать переменные, заключенные в две фигурные скобки. Поддерживаются следующие переменные:

– {{cluster}} – имя кластера;
– {{guestname}} – имя ВМ/контейнера;
– {{node}} – имя узла, для которого создается резервная копия;
– {{vmid}} – VMID ВМ/контейнера.

Создание задания для резервного копирования. Вкладка «Note Template»

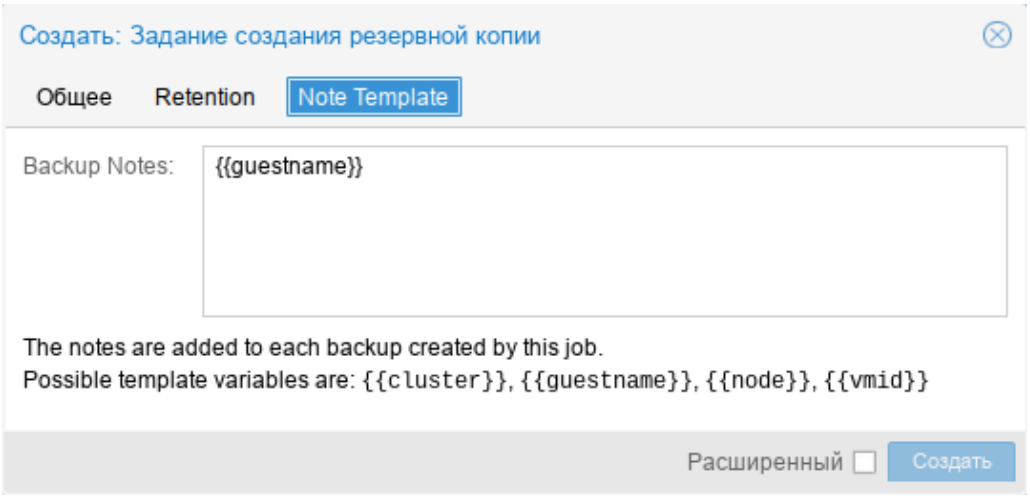


Рис. 172

После указания необходимых параметров и нажатия кнопки «Создать», задание для резервного копирования появляется в списке (Рис. 173). Запись о задании создаётся в файле `/etc/pve/jobs.cfg`. Оно будет запускаться в назначенное время.

Задание резервного копирования

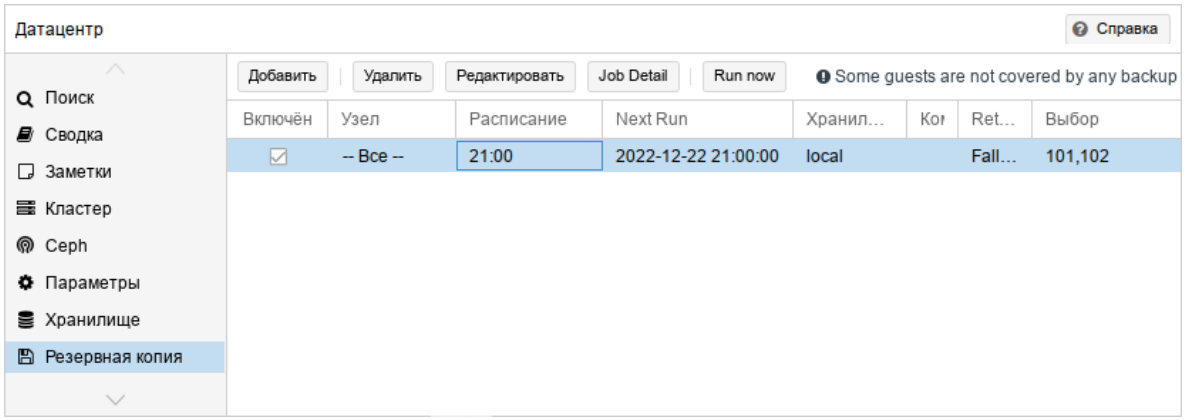


Рис. 173

Также существует возможность запустить задание по требованию – кнопка «Run now».

Для того чтобы разово создать резервную копию конкретной ВМ, достаточно открыть ВМ, выбрать в ней раздел «Резервная копия» («Backup») и нажать кнопку «Создать резервную копию сейчас» («Backup Now») (Рис. 174).

Вкладка «Резервная копия» ВМ

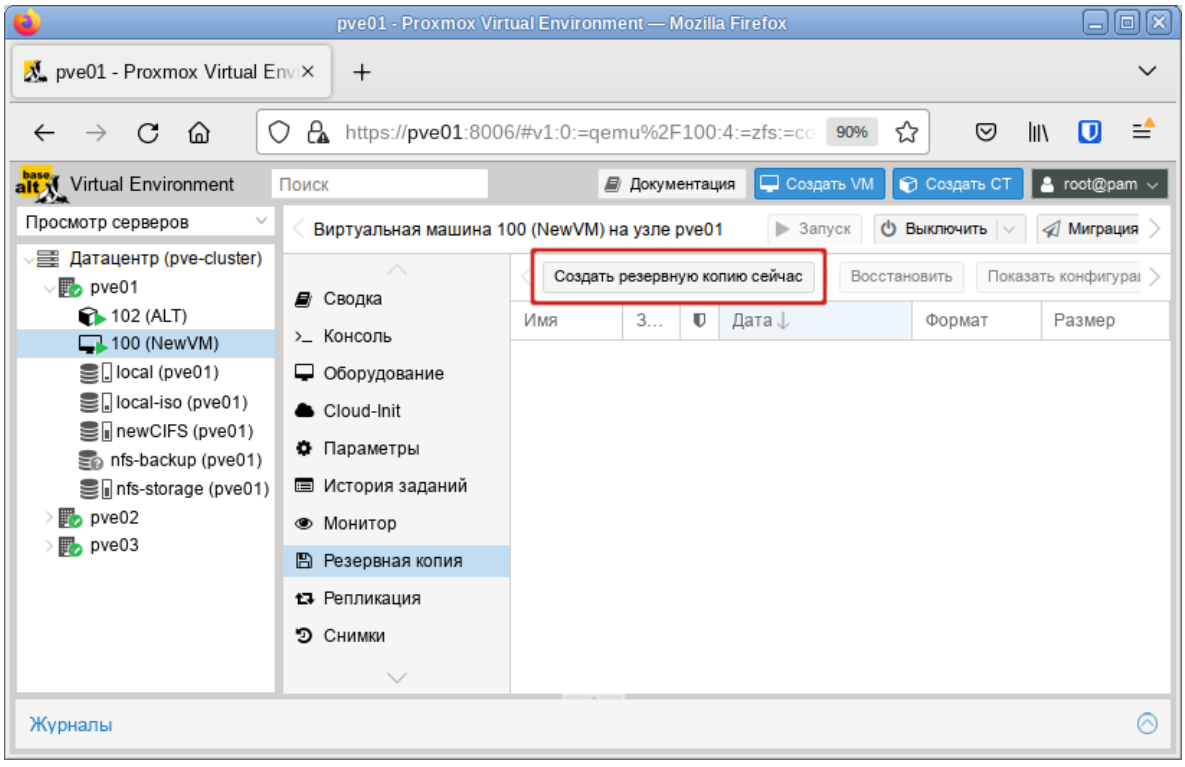


Рис. 174

Далее следует указать параметры резервного копирования (Рис. 175).

После создания резервной копии рекомендуется сразу убедиться, что из нее можно восстановить ВМ. Для этого необходимо открыть хранилище с резервной копией (Рис. 176) и начать процесс восстановления (Рис. 177). При восстановлении можно указать новое имя и переопределить некоторые параметры ВМ.

Выбор режима создания резервной копии

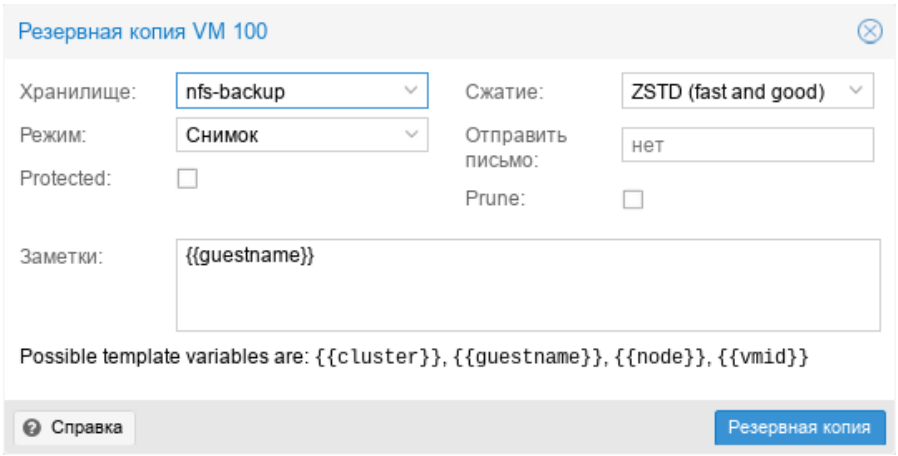


Рис. 175

Резервная копия в хранилище nfs-backup

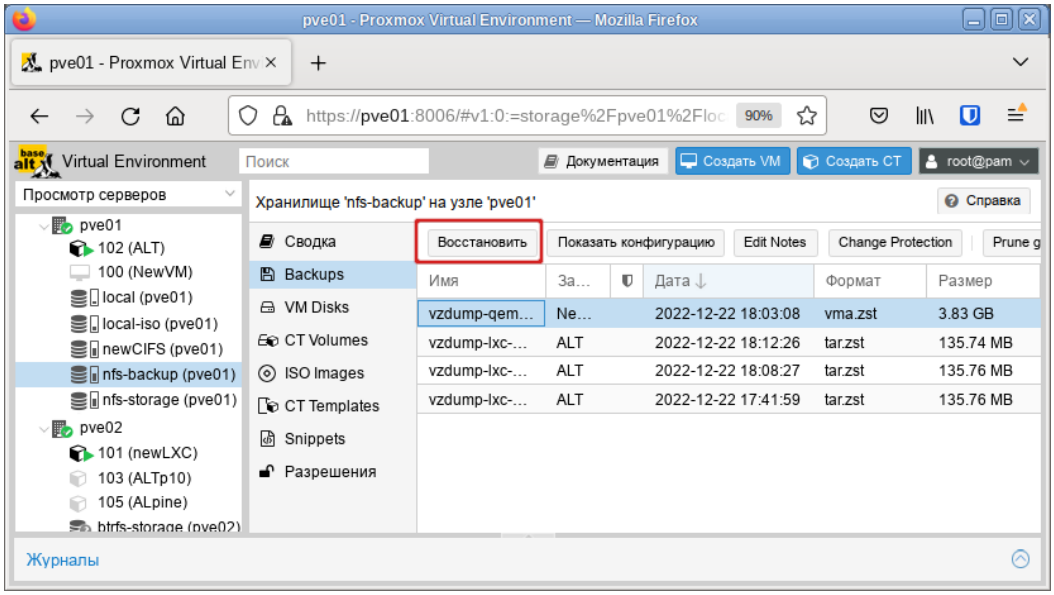


Рис. 176

Восстановить ВМ из резервной копии

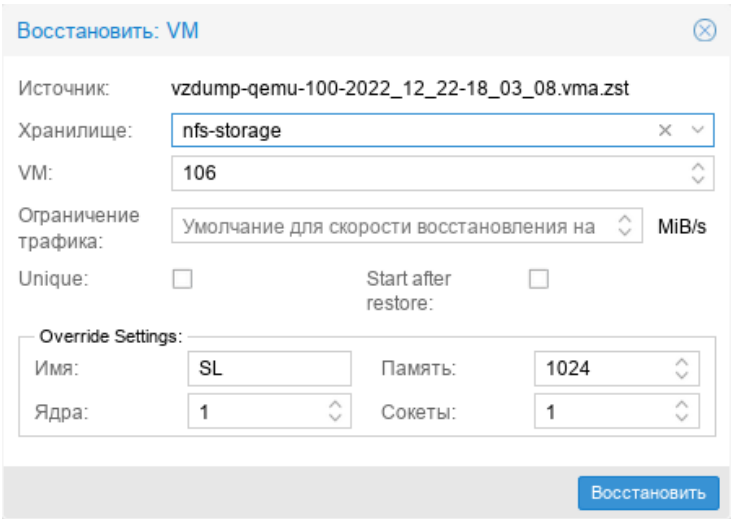


Рис. 177

Если восстанавливать из резервной копии в интерфейсе ВМ (Рис. 178), то будет предложена только замена существующей ВМ.

Восстановление из резервной копии в интерфейсе ВМ

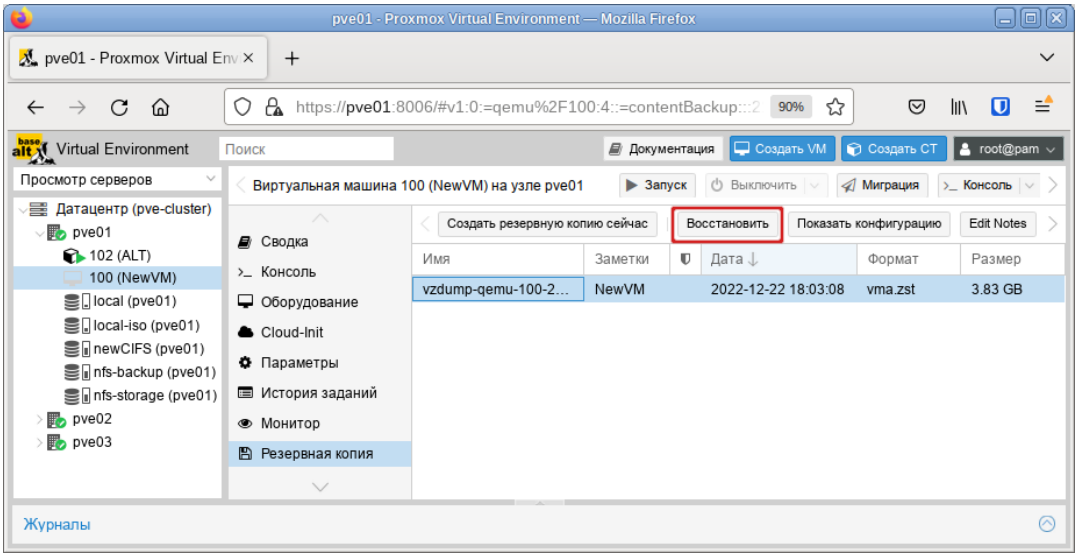


Рис. 178

Резервную копию можно пометить как защищённую, чтобы предотвратить ее удаление (*Рис. 179*).

Пометить резервную копию как защищенную

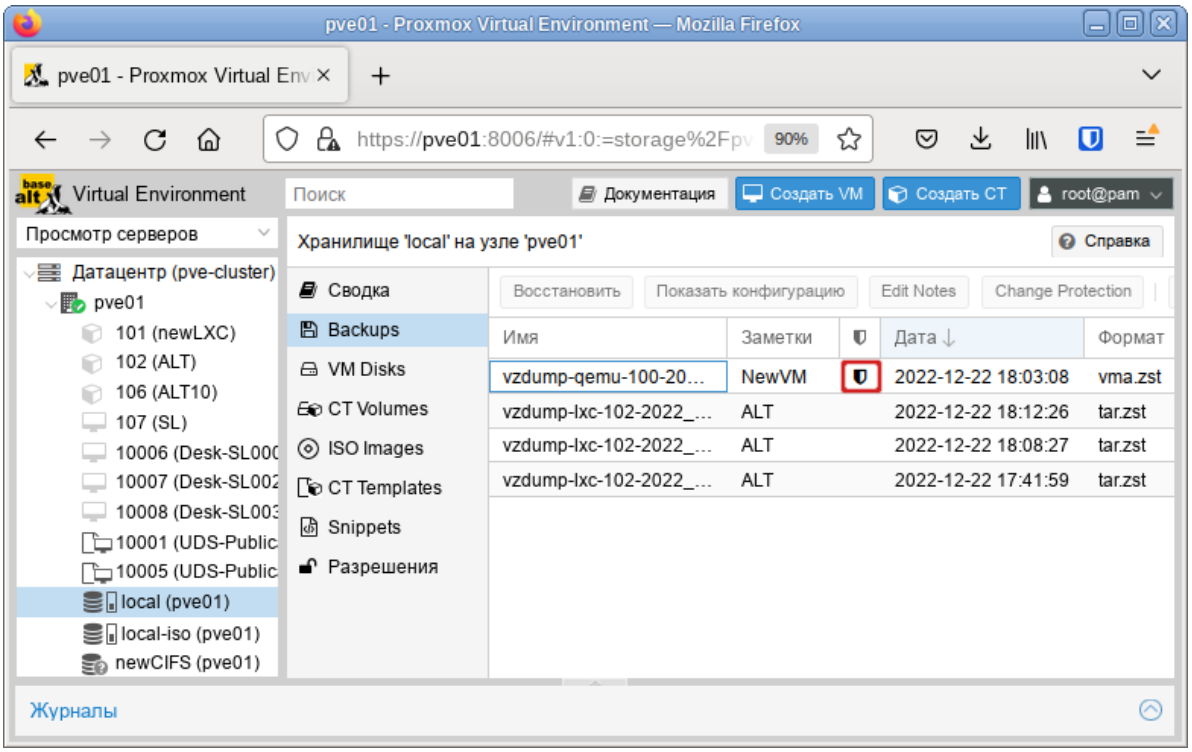


Рис. 179

Примечание. Попытка удалить защищенную резервную копию через пользовательский интерфейс, интерфейс командной строки или API PVE не удастся. Но так как это обеспечивается PVE, а не файловой системой, ручное удаление самого файла резервной копии по-прежнему возможно для любого, у кого есть доступ на запись к хранилищу резервных копий.

4.10.7 Резервное копирование из командной строки

4.10.7.1 Файлы резервных копий

Все создаваемые резервные копии будут сохраняться в поддиректории «dump». Имя файла резервной копии будет иметь вид:

- vzdump-qemu-номер_машины-дата-время.vma.zst в случае выбора метода сжатия ZST;
- vzdump-qemu-номер_машины-дата-время.vma.gz в случае выбора метода сжатия GZIP;
- vzdump-qemu-номер_машины-дата-время.vma.lzo для использования метода LZO.

4.10.7.2 Восстановление

Восстановить данные из резервных копий можно в веб-интерфейсе PVE или с помощью следующих утилит:

- `pct restore` – утилита восстановления контейнера;
- `qmrestore` – утилита восстановления ВМ.

4.10.7.3 Ограничение пропускной способности

Для восстановления одной или нескольких больших резервных копий может потребоваться много ресурсов, особенно пропускной способности хранилища как для чтения из резервного хранилища, так и для записи в целевое хранилище. Это может негативно повлиять на работу других ВМ, так как доступ к хранилищу может быть перегружен. Чтобы этого избежать, можно установить ограничение полосы пропускания для задания резервного копирования. В PVE есть два вида ограничений для восстановления и архивирования:

- per-restore limit – максимальный объем полосы пропускания для чтения из архива резервной копии;
- per-storage write limit – максимальный объем полосы пропускания, используемый для записи в конкретное хранилище.

Ограничение чтения косвенно влияет на ограничение записи. Меньшее ограничение на задание перезапишет большее ограничение на хранилище. Увеличение лимита на задание приведёт к перезаписи лимита на хранилище, только если для данного хранилища есть разрешения «Data.Allocate»

Примечание. Чтобы отключить все ограничения для конкретного задания можно использовать значение `0` для параметра `bwlimit`. Это может быть полезно, если требуется как можно быстрее восстановить ВМ.

Установить ограничение пропускной способности по умолчанию для хранилища, можно с помощью команды:

```
# pvesm set STORAGEID --bwlimit restore=KIBs
```

4.10.7.4 Файл конфигурация vzdump.conf

Глобальные настройки создания резервных копий хранятся в файле конфигурации `/etc/vzdump.conf`. Каждая строка файла имеет следующий формат (пустые строки в файле игнорируются, строки, начинающиеся с символа #, рассматриваются как комментарии и также игнорируются):

```
OPTION: value
```

Поддерживаемые опции представлены в табл. 10.

Пример `vzdump.conf`:

```
tmpdir: /mnt/fast_local_disk
storage: my_backup_storage
mode: snapshot
bwlimit: 10000
```

4.10.7.5 Файлы, не включаемые в резервную копию

Примечание. Эта опция доступна только при создании резервных копий контейнеров.

Команда vzdump по умолчанию пропускает следующие файлы (отключается с помощью опции `--stdexcludes 0`):

```
/tmp/?*
/var/tmp/?*
/var/run/?*pid
```

Кроме того, можно вручную указать какие файлы исключать (дополнительно), например:

```
# vzdump 777 --exclude-path /tmp/ --exclude-path '/var/foo*'
```

Файлы конфигурации ВМ и контейнеров также хранятся внутри архива резервных копий (в `/etc/vzdump/`) и будут корректно восстановлены.

4.10.7.6 Примеры

Создать простую резервную копию гостевой системы 103 – без снимков, только архив гостевой части и конфигурационного файла в каталог резервного копирования по умолчанию (обычно `/var/lib/vz/dump/`):

```
# vzdump 103
```

Использовать `rsync` и режим приостановки для создания снимка (минимальное время простоя):

```
# vzdump 103 --mode suspend
```

Сделать резервную копию всей гостевой системы и отправить отчет пользователям root и admin:

```
# vzdump --all --mode suspend --mailto root --mailto admin
```

Т а б л и ц а 10 – Опции резервного копирования

Опция	Описание
bwlimit: integer (0 - N) (default=0)	Ограничение пропускной способности ввода/вывода (Кб/с)
compress: (0 | 1 | gzip | lzo | zstd) (default=0)	Сжатие файла резервной копии
dumpdir: string	Записать результирующие файлы в указанный каталог
exclude-path: string	Исключить определенные файлы/каталоги
ionice: integer (0 - 8) (default=7)	Установить CFQ приоритет ionice
lockwait: integer (0 - N) (default=180)	Максимальное время ожидания для глобальной блокировки (в минутах)
mailnotification: (always | failure) (default=always)	Указание, когда следует отправить отчет по электронной почте
mailto: string	Разделенный запятыми список адресов электронной почты, на которые будут приходить уведомления
maxfiles: integer (1 - N) (default=1)	Максимальное количество файлов резервных копий ВМ
mode: (snapshot | stop | suspend) (default=snapshot)	Режим резервного копирования
pigz: integer (default=0)	Использует pigz вместо gzip при N>0. N=1 использует половину ядер (uses half of cores), при N>1 N – количество потоков
prune-backups: [keep-all=<1|0>] [,keep-daily=<N>] [,keep-hourly=<N>] [,keep-last=<N>] [,keep-monthly=<N>] [,keep-weekly=<N>] [,keep-yearly=<N>]	Использовать эти параметры хранения вместо параметров из конфигурации хранилища (см.выше)
remove: boolean (default=1)	Удалить старые резервные копии, если их больше, чем установлено опцией maxfiles
script: string	Использовать указанный скрипт
stdexcludes: boolean (default=1)	Исключить временные файлы и файлы журналов
stopwait: integer (0 - N) (default=10)	Максимальное время ожидания пока гостевая система не остановится (минуты)
storage: string	Хранить полученный файл в этом хранилище
tmpdir: string	Хранить временные файлы в указанном каталоге
zstd: integer (default = 1)	Количество потоков zstd. N = 0 использовать половину доступных ядер, N> 0 использовать N как количество потоков

Использовать режим мгновенного снимка (снапшота) (нет времени простоя) и каталог для хранения резервных копий /mnt/backup:

```
# vzdump 103 --dumpdir /mnt/backup --mode snapshot
```

Резервное копирование более чем одной ВМ (выборочно):

```
# vzdump 101 102 103 --mailto root
```

Резервное копирование всех ВМ, исключая 101 и 102:

```
# vzdump --mode suspend --exclude 101,102
```

Восстановить контейнер в новый контейнер 600:

```
# pct restore 600 /mnt/backup/vzdump-lxc-777.tar
```

Восстановить QemuServer VM в VM 601:

```
# qmrestore /mnt/backup/vzdump-qemu-888.vma 601
```

Клонировать существующий контейнер 101 в новый контейнер 300 с 4GB корневой файловой системы:

```
# vzdump 101 --stdout | pct restore --rootfs 4 300 -
```

4.11 Снимки (snapshot)

Снимки ВМ – это файловые снимки состояния, данных диска и конфигурации ВМ в определенный момент времени. Можно создать несколько снимков ВМ даже во время ее работы. Затем можно возвратить ее в любое из предыдущих состояний, применив моментальный снимок к ВМ.

Чтобы создать снимок состояния системы необходимо в меню ВМ выбрать пункт «Снимки» («Snapshots») и нажать кнопку «Сделать снимок» (Рис. 180). В открывшемся окне (Рис. 181) следует ввести название снимка и нажать кнопку «Сделать снимок».

Окно управления снимками ВМ

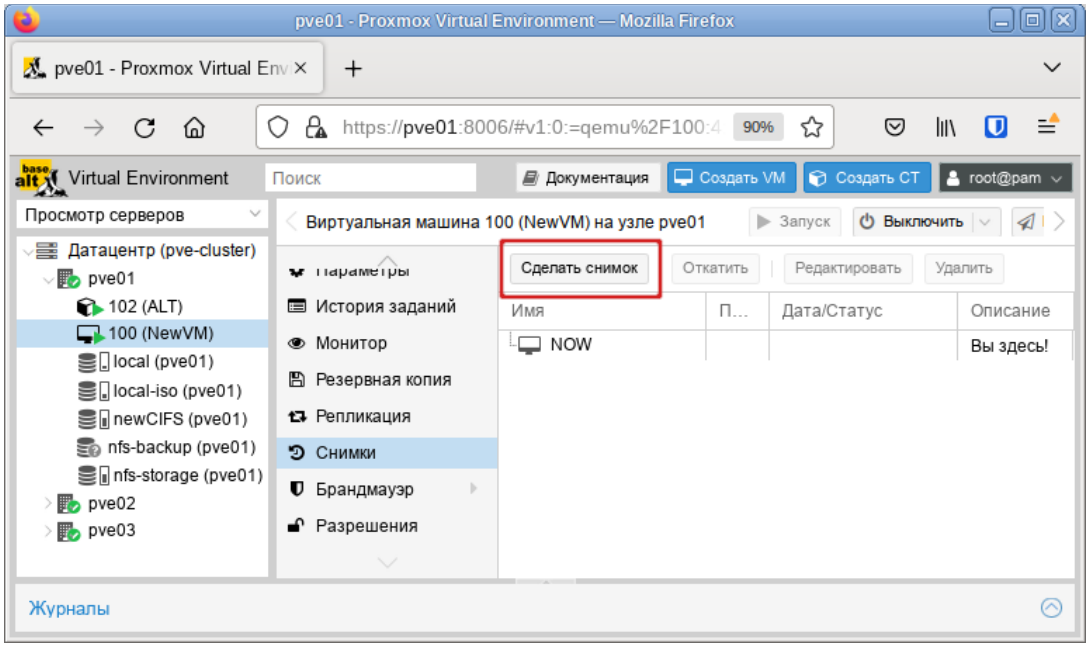


Рис. 180

Создание снимка ВМ

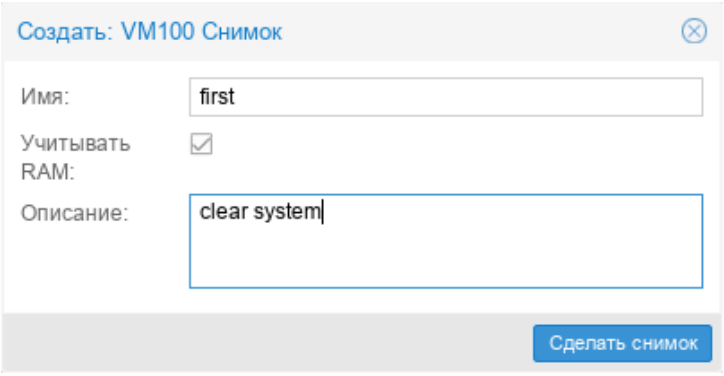


Рис. 181

Для того чтобы восстановить ВМ из снимка, необходимо в меню ВМ выбрать пункт «Снимки» («Snapshots»), выбрать снимок (Рис. 182) и нажать кнопку «Откатить».

Восстановление ОС из снимка

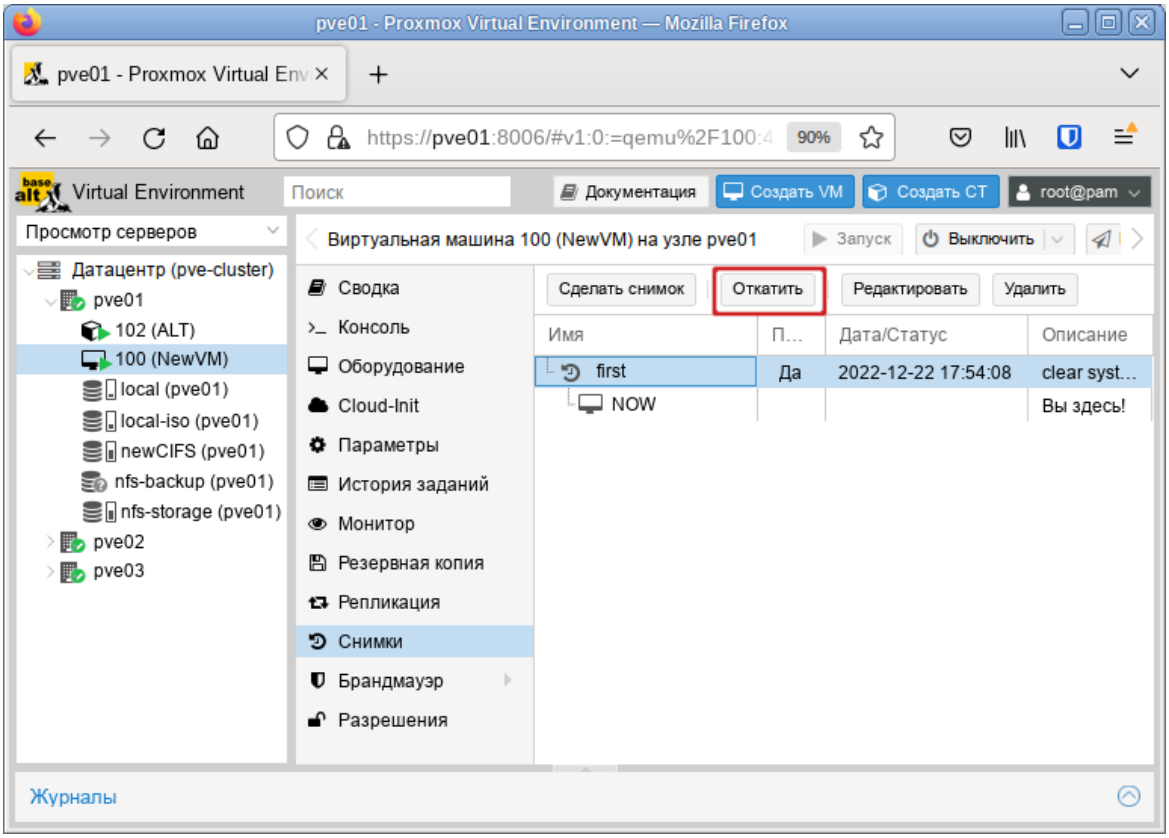


Рис. 182

При создании снимков, `qm` сохраняет конфигурацию ВМ во время снимка в отдельном разделе в файле конфигурации ВМ. Например, после создания снимка с именем `first` файл конфигурации будет выглядеть следующим образом:

```
boot: order=scsi0;sata2;net0
cores: 1
memory: 1024
```

```
meta: creation-qemu=7.1.0,ctime=1671708251
name: NewVM
net0: virtio=3E:E9:24:FF:85:D9,bridge=vmbr0,firewall=1
numa: 0
ostype: l26
parent: first
sata2: local-iso:iso/slinux-10.1-x86_64.iso,media=cdrom,size=4586146K
scsi0: local:100/vm-100-disk-0.qcow2,size=42G
scsihw: virtio-scsi-pci
smbios1: uuid=ee9db068-5427-4934-bf7a-5895c377b5af
sockets: 1
vmgenid: dfec8e3b-d391-40cb-8983-b4938461b79a

[first]
#clear system
boot: order=scsi0;sata2;net0
cores: 1
memory: 1024
meta: creation-qemu=7.1.0,ctime=1671708251
name: NewVM
net0: virtio=3E:E9:24:FF:85:D9,bridge=vmbr0,firewall=1
numa: 0
ostype: l26
runningcpu: kvm64,enforce,+kvm_pv_eoi,+kvm_pv_unhalt,+lahf_lm,+sep
runningmachine: pc-i440fx-7.1+pve0
sata2: local-iso:iso/slinux-10.1-x86_64.iso,media=cdrom,size=4586146K
scsi0: local:100/vm-100-disk-0.qcow2,size=42G
scsihw: virtio-scsi-pci
smbios1: uuid=ee9db068-5427-4934-bf7a-5895c377b5af
snaptime: 1671724448
sockets: 1
vmgenid: dfec8e3b-d391-40cb-8983-b4938461b79a
vmstate: local:100/vm-100-state-first.raw
```

Свойство `parent` используется для хранения родительских/дочерних отношений между снимками, `snaptime` – это отметка времени создания снимка (эпоха Unix).

4.12 Встроенный мониторинг PVE

Все данные о потреблении ресурсов и производительности можно найти на вкладках «Сводка» («Summary») узлов PVE и ВМ. Можно просматривать данные на основе почасового, ежедневного, еженедельного или за год периодов.

На Рис. 183 показана «Сводка» («Summary») узла pve01 со списком для выбора периода данных.

Просмотреть список всех узлов, ВМ и контейнеров в кластере можно, выбрав «Датацентр» → «Поиск» («Datacentre» → «Search») **(**Рис. 184**)**. Этот список может быть отсортирован по полям: «Тип» («Type»), «Описание» («Description»), «Использование диска %» («Disk usage»), «Использование памяти» («Memory usage»), «Загрузка CPU» («CPU usage») и «Время работы» («Uptime»). В этом списке отображается потребление ресурсов только в реальном масштабе времени.

Выбор периода данных, для отображения отчета

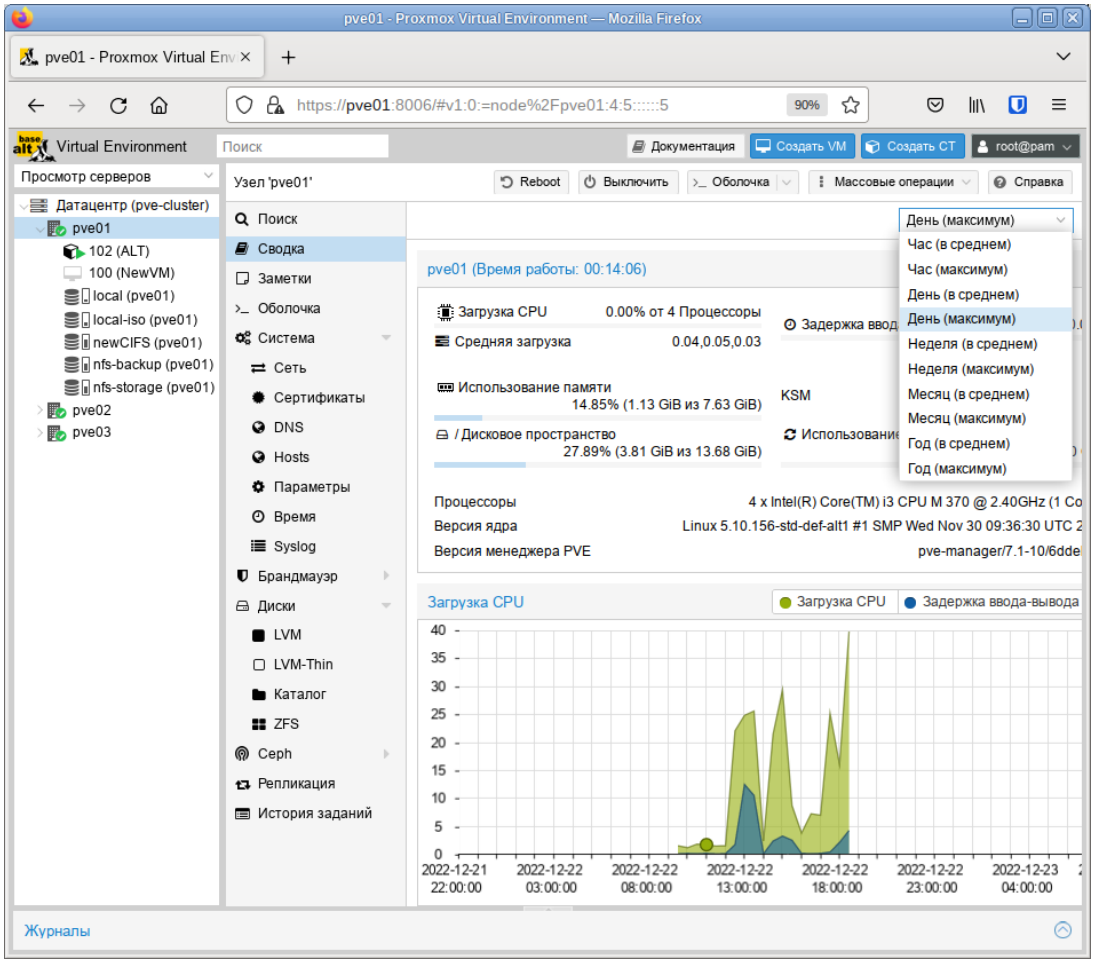


Рис. 183

Потребление ресурсов

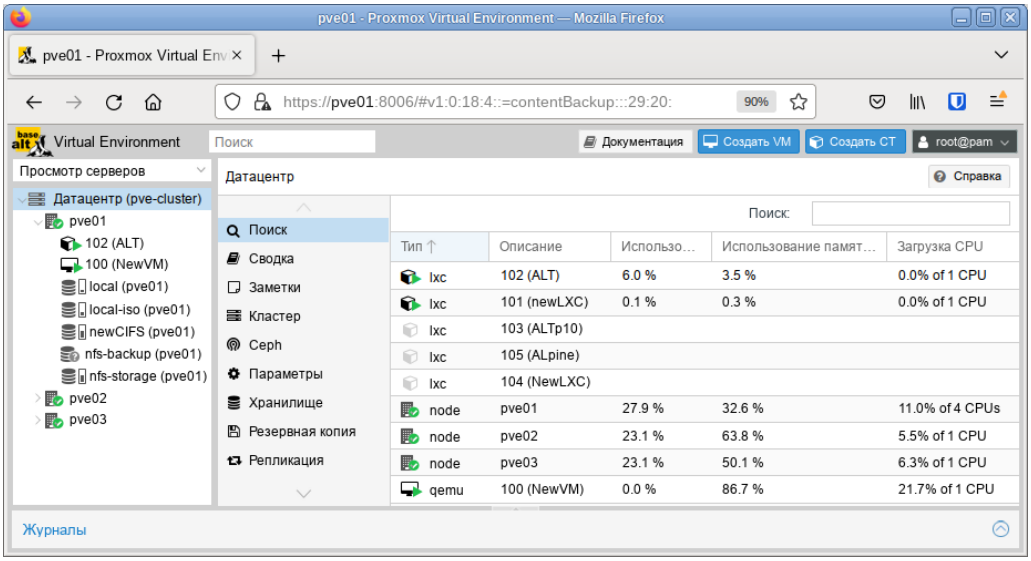


Рис. 184

Для мониторинга состояния локальных дисков используется пакет `smartmontools`. Он содержит набор инструментов для мониторинга и управления S.M.A.R.T. системой для локальных жестких дисков.

Получить статус диска можно, выполнив следующую команду:

```
# smartctl -a /dev/sdX
```

где /dev/sdX – это путь к одному из локальных дисков.

Включить поддержку SMART для диска, если она отключена:

```
# smartctl -s on /dev/sdX
```

Просмотреть S.M.A.R.T. статус диска в веб-интерфейсе можно, выбрав в разделе «Диски» нужный диск и нажав кнопку «Показать данные S.M.A.R.T.» (*Рис. 185*).

Кнопка «Показать данные S.M.A.R.T.»

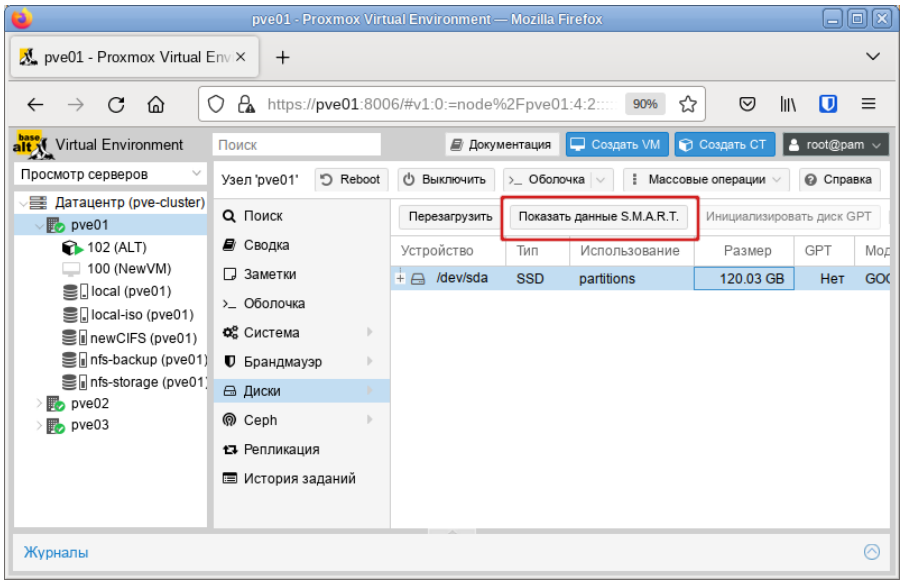


Рис. 185

По умолчанию, `smartmontools daemon smartd` активен и включен, и сканирует диски в /dev/sdX и /dev/hdX каждые 30 минут на наличие ошибок и предупреждений, а также отправляет сообщение электронной почты пользователю root в случае обнаружения проблемы (для пользователя root в PVE должен быть введен действительный адрес электронной почты).

Электронное сообщение будет содержать имя узла, где возникла проблема, а также параметры самого устройства, такие как серийный номер и идентификатор дискового устройства. Если та же самая ошибка продолжит возникать, узел будет отсылать электронное сообщение каждые 24 часа. Основываясь на содержащейся в электронном сообщении информации можно определить отказавшее устройство и заменить его в случае такой необходимости.

4.13 Высокая доступность PVE

Высокая доступность PVE (High Availability, HA) позволяет кластеру перемещать или мигрировать ВМ с отказавшего узла на жизнеспособный узел без вмешательства пользователя.

Для функционирования HA в PVE необходимо чтобы все ВМ использовали общее хранилище. HA PVE обрабатывает только узлы PVE и ВМ в пределах кластера PVE. Такую функциональность HA не следует путать с избыточностью общих хранилищ, которую PVE может применять в своем развертывании HA. Общие хранилища сторонних производителей могут предоставлять свою собственную функциональность HA.

В вычислительном узле PVE могут существовать свои уровни избыточности, например, применение RAID, дополнительные источники питания, объединение/агрегация сетей. HA в PVE не подменяет собой ни один из этих уровней, а просто способствует использованию функций избыточности ВМ для сохранения их в рабочем состоянии при отказе какого-либо узла.

4.13.1 Как работает высокая доступность PVE

PVE предоставляет программный стек ha-manager, который может автоматически обнаруживать ошибки и выполнять автоматический переход на другой ресурс. Основной блок управления, управляемый ha-manager называется ресурсом. Ресурс (сервис) однозначно идентифицируется идентификатором сервиса (SID), который состоит из типа ресурса и идентификатора, специфичного для данного типа, например, vm: 100 (ресурс типа ВМ с идентификатором 100).

В случае, когда по какой-либо причине узел становится недоступным, HA PVE ожидает 60 секунд прежде чем выполнится ограждение (fencing) отказавшего узла. Ограждение предотвращает службы кластера от возврата в рабочее состояние в этом месте. Затем HA перемещает эти ВМ и контейнеры на следующий доступный узел в группе участников HA. Даже если узел с ВМ включен, но потерял связь с сетевой средой, HA PVE попытается переместить все ВМ с этого узла на другой узел.

При возврате отказавшего узла в рабочее состояние, HA не переместит ВМ на первоначальный узел. Это необходимо выполнять вручную. При этом ВМ может быть перемещена вручную только если HA запрещен для данной ВМ. Поэтому сначала следует выключить HA, а затем переместить на первоначальный узел и включить HA на данной ВМ вновь.

4.13.2 Требования для настройки высокой доступности

Среда PVE для настройки HA должна отвечать следующим требованиям:

- кластер, содержащий, как минимум, три узла (для получения надежного кворума);
- общее хранилище для ВМ и контейнеров;
- аппаратное резервирование;
- использование надежных «серверных» компонентов;
- аппаратный сторожевой таймер (если он недоступен, используется программный таймер ядра Linux);
- дополнительные устройства ограждения (fencing).

Примечание. В случае построения виртуальной инфраструктуры на серверах HP необходимо запретить загрузку модуля ядра hpwdt. Для этого необходимо создать файл `/etc/modprobe.d/nohpwdt.conf` со следующим содержимым (для применения изменений следует перезагрузить систему):

```
# Do not load the 'hpwdt' module on boot.
blacklist hpwdt
```

4.13.3 Настройка высокой доступности PVE

Все настройки HA PVE могут быть выполнены в веб-интерфейсе в разделе «Датацентр» → «HA» («Datacenter» → «HA») (Рис. 186).

Меню HA. Статус настройки HA

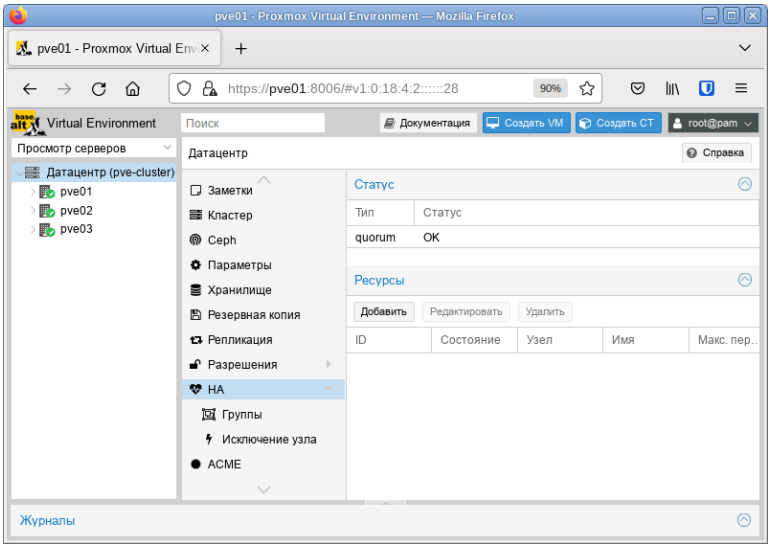


Рис. 186

4.13.3.1 Создание группы высокой доступности

Наиболее характерным примером использования групп HA являются некие программные решения или инфраструктура ВМ, которые должны работать совместно (например, контроллер домена, файловый сервер и т.д.). Назначенные в определенную группу ВМ могут перемещаться только между узлами участниками этой группы. Например, есть шесть узлов, три из которых обладают всей полнотой ресурсов, достаточной для исполнения виртуального сервера базы данных, а другие три узла выполняют виртуальные рабочие столы или решения VDI. Можно создать две группы, для которых виртуальные серверы баз данных могут перемещаться только в пределах тех узлов, которые будут назначены для данной группы. Это гарантирует, что ВМ переместится на тот узел, который будет способен исполнять такие ВМ.

Для включения HA необходимо создать как минимум одну группу.

Для создания группы следует нажать кнопку «Создать» («Create») в подменю «Группы» («Groups»).

Элементы, доступные в блоке диалога «Группа HA» (Рис. 187):

- «ID» – название HA группы;
- «Узел» («Node») – назначение узлов в создаваемую группу (нужно выбрать, по крайней мере, один узел);
- «Restricted» – разрешение перемещения ВМ со стороны HA PVE только в рамках узлов участников данной группы HA. Если перемещать ВМ некуда, то эти ВМ будут автоматически остановлены;
- «Nofailback» – используется для предотвращения автоматического восстановления состояния ВМ/контейнера при восстановлении узла в кластере (не рекомендуется включать эту опцию).

Диалог создания группы

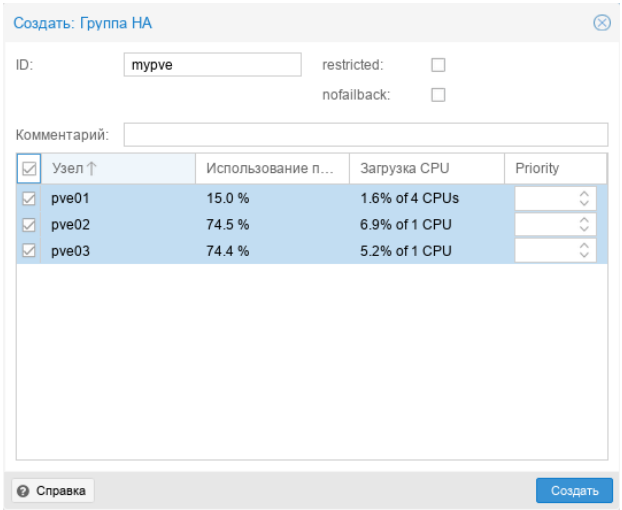


Рис. 187

На Рис. 188 представлено подменю «Группы» с созданной группой.

Подменю «Группы» с созданной группой

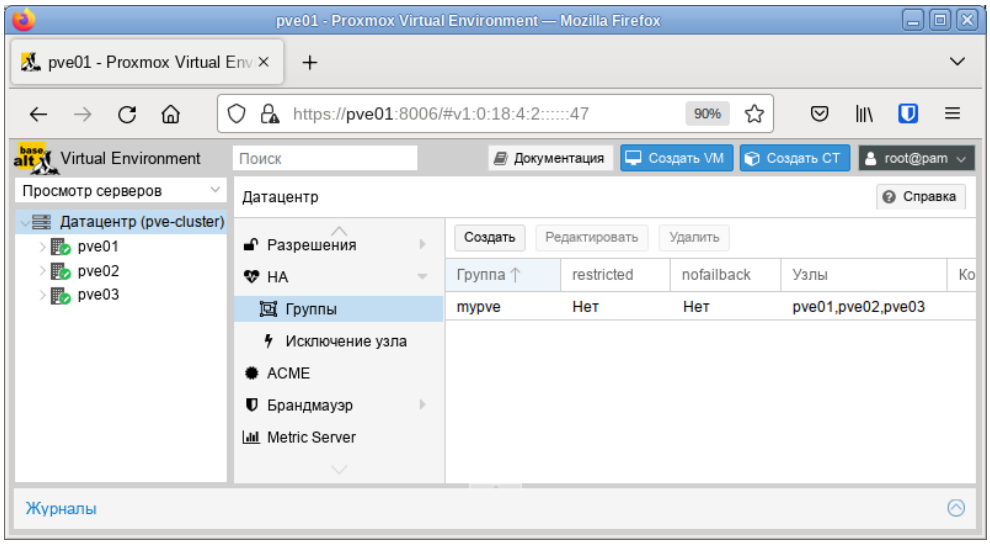


Рис. 188

4.13.3.2 Добавление ресурсов

Для включения HA для ВМ или контейнера следует нажать на кнопку «Добавить» в разделе «Ресурсы» меню «HA». В открывшемся диалоговом окне нужно выбрать ВМ/контейнер и группу HA (Рис. 189).

Добавление Ресурса в группу

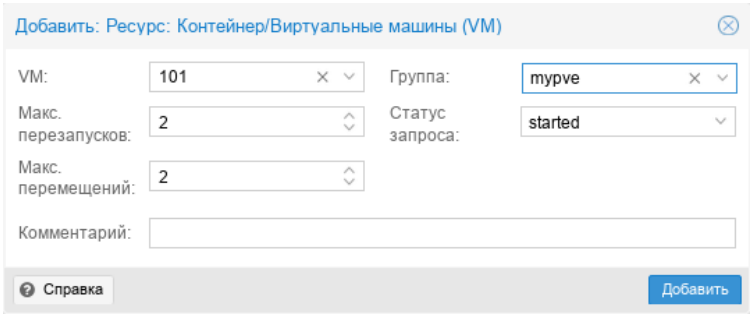


Рис. 189

В окне можно настроить следующие параметры:

– «Макс. перезапусков» («Max. Restart») – количество попыток запуска ВМ/контейнера на новом узле после перемещения;
– «Макс. перемещений» («Max. Relocate») – количество попыток перемещения ВМ/контейнера на новый узел;
– «Статус запроса» («Request State») – доступны варианты: started – кластер менеджер будет пытаться поддерживать состояние машины в запущенном состоянии; stopped – при отказе узла перемещать ресурс, но не пытаться запустить; ignored – ресурс, который не надо перемещать при отказе узла; disabled – в этот статус переходят ВМ, которые находятся в состоянии «error».

На Рис. 190 показана группа HA PVE и добавленные в нее ВМ и контейнеры, которыми будет управлять HA.

Раздел «Статус» («Status») отображает текущее состояние функциональности HA:

- кворум кластера установлен;
- главный узел pve03 группы HA активен и последний временной штамп жизнеспособности (heartbeat timestamp) проверен;
- все узлы, участвующие в группе HA активны и последний временной штамп жизнеспособности (heartbeat timestamp) проверен.

Список ресурсов

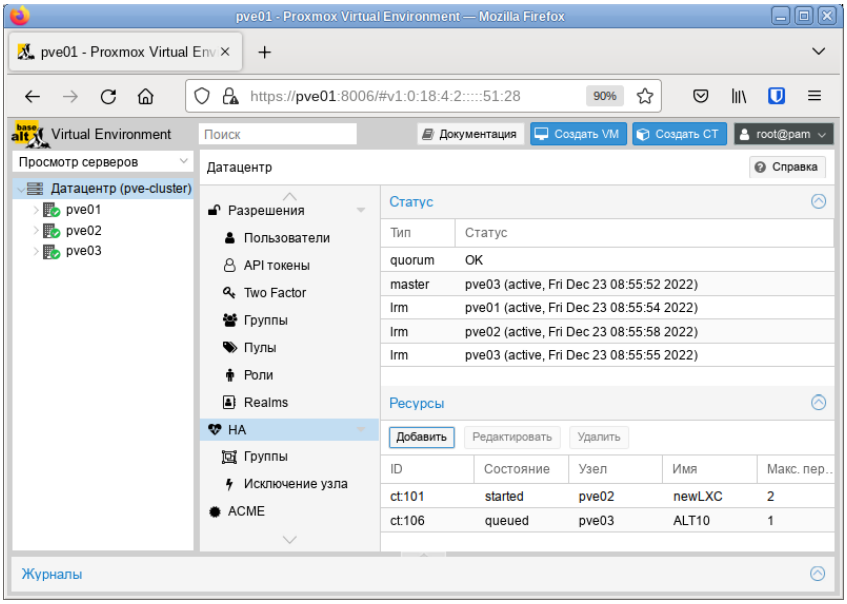


Рис. 190

Просмотреть состояние функциональности HA можно и в консоли:

```
# ha-manager status
quorum OK
master pve03 (active, Fri Dec 23 08:56:22 2022)
lrm pve01 (active, Fri Dec 23 08:56:14 2022)
lrm pve02 (active, Fri Dec 23 08:56:18 2022)
lrm pve03 (active, Fri Dec 23 08:56:15 2022)
service ct:101 (pve02, started)
service ct:106 (pve03, started)
```

4.13.4 Тестирование настройки высокой доступности PVE

Для того чтобы убедиться, что HA действительно работает можно отключить сетевое соединение для pve03 и понаблюдать за окном «Статус» (Рис. 190) на предмет изменений HA.

После того как соединение с узлом pve03 будет потеряно, он будет помечен как не доступный. По истечению 60 секунд, HA PVE предоставит следующий доступный в группе HA узел в качестве главного (Рис. 191).

После того как HA PVE предоставит новый ведущий узел для группы HA, будет запущено ограждение для ресурсов ВМ/контейнера для подготовки к перемещению их на другой узел. В процессе ограждения, все связанные с данной ВМ службы ограждаются, что означает, что даже если отказавший узел вернется в строй на этом этапе, ВМ не будут иметь возможность восстановить свою нормальную работу. Затем ВМ/контейнер полностью останавливается. Так как узел сам по себе отключен, ВМ/контейнер не может выполнить миграцию в реальном режиме времени, поскольку состояние оперативной памяти исполняемой ВМ не может быть получено с отключенного узла.

Изменение главного узла на pve02

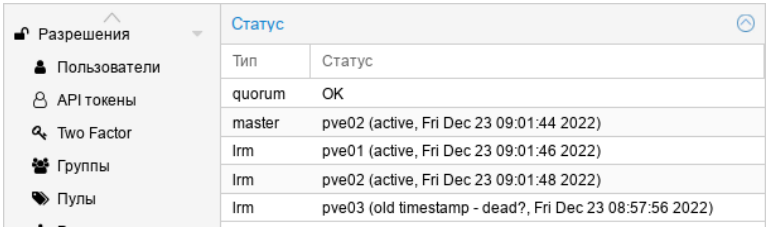


Рис. 191

После остановки, ВМ/контейнер перемещается на следующий свободный узел в группе HA и автоматически запускается. В данном примере контейнер 106 перемещен на узел pve01 и запущен (Рис. 192).

Контейнер 106 запущен на узле pve01

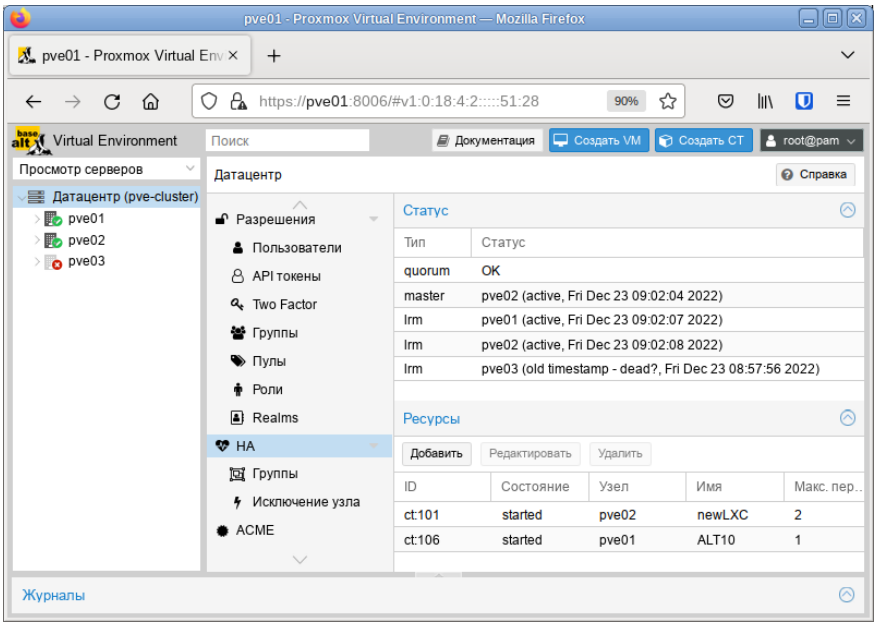


Рис. 192

В случае возникновения любой ошибки, HA PVE выполнит несколько попыток восстановления в соответствии с политиками `restart` и `relocate`. Если все попытки окажутся неудачными, HA PVE поместит ресурсы в ошибочное состояние и не будет выполнять для них никаких задач.

4.14 Пользователи и их права

PVE поддерживает несколько источников аутентификации, например Linux PAM, интегрированный сервер аутентификации PVE (Рис. 193), LDAP, Active Directory и OpenID Connect.

Выбор типа аутентификации в веб-интерфейсе

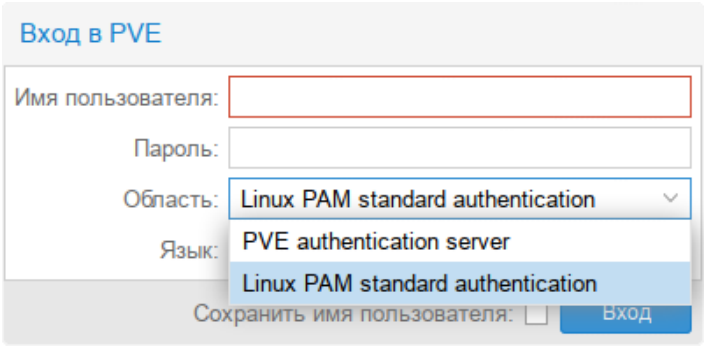


Рис. 193

Используя основанное на ролях управление пользователями и разрешениями для всех объектов (ВМ, хранилищ, узлов и т. д.), можно определить многоуровневый доступ.

PVE хранит данные пользователей в файле `/etc/pve/user.cfg`:

```
# cat /etc/pve/user.cfg
user:root@pam:1:0::::::
user:test@pve:1:0::::::
user:testuser@pve:1:0::::Just a test::
user:user@pam:1:0::::::

group:admin:user@pam::
group:testgroup:test@pve::
```

Пользователя часто внутренне идентифицируют по имени пользователя и области аутентификации в форме <user>@<realm>.

После установки PVE существует один пользователь root@pam, который соответствует суперпользователю ОС. Этого пользователя нельзя удалить, все системные письма будут отправляться на адрес электронной почты, назначенный этому пользователю. Суперпользователь имеет

неограниченные права, поэтому рекомендуется добавить других пользователей с меньшими правами.

Каждый пользователь может быть членом нескольких групп.

4.14.1 Области аутентификации

Чтобы пользователь мог выполнить какое-либо действие (например, просмотр, изменение или удаление ВМ), ему необходимо иметь соответствующие разрешения.

Доступны следующие области (методы) аутентификации:

- Стандартная аутентификация Linux PAM – общесистемная аутентификация пользователей;
- Сервер аутентификации PVE – пользователи полностью управляются PVE и могут менять свои пароли через графический интерфейс. Этот метод аутентификации удобен для небольших (или даже средних) установок, где пользователям не нужен доступ ни к чему, кроме PVE;
- Сервер LDAP – позволяет использовать внешний LDAP-сервер для аутентификации пользователей (например, OpenLDAP);
- Сервер Active Directory **(**Active Directory Server**)** – позволяет аутентифицировать пользователей через AD. Поддерживает LDAP в качестве протокола аутентификации;
- OpenID Connect Server – уровень идентификации поверх протокола OATH 2.0. Позволяет аутентифицировать пользователей на основе аутентификации, выполняемой внешним сервером авторизации.

Настройки области аутентификации хранятся в файле `/etc/pve/domains.cfg`.

4.14.1.1 Стандартная аутентификация Linux PAM

При использовании аутентификации Linux PAM, системный пользователь должен существовать (должен быть создан, например, с помощью команды `adduser`) на всех узлах, на которых пользователю разрешено войти в систему. Если пользователи PAM существуют в хост-системе PVE, соответствующие записи могут быть добавлены в PVE, чтобы эти пользователи могли входить в систему, используя свое системное имя и пароль.

Область Linux PAM создается по умолчанию и не может быть удалена. Администратор может добавить требование двухфакторной аутентификации для пользователей данной области («Require TFA») и установить её в качестве области по умолчанию для входа в систему («По умолчанию») (*Рис. 194*).

Для добавления нового пользователя, необходимо в окне «Датацентр» → «Разрешения» → «Пользователи» нажать кнопку «Добавить». На Рис. 195 показано создание нового пользователя с использованием PAM аутентификации (системный пользователь user должен существовать, в качестве пароля будет использоваться пароль для входа в систему).

Конфигурация PAM аутентификации

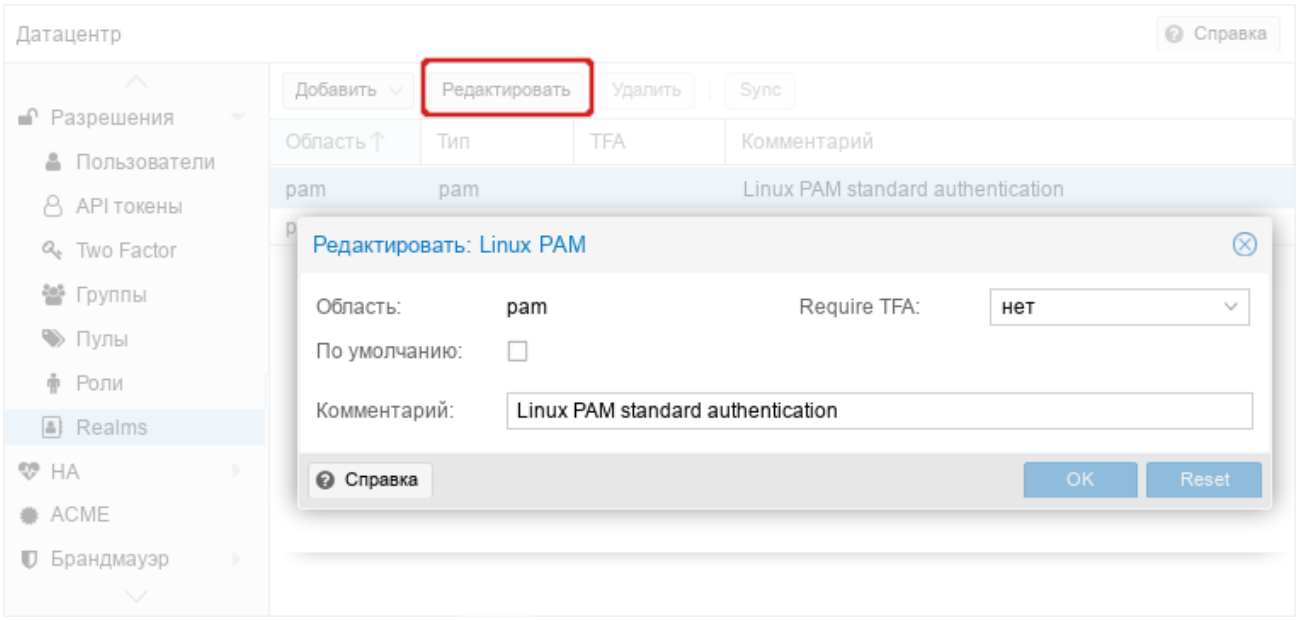


Рис. 194

Создание нового пользователя с использованием PAM аутентификации

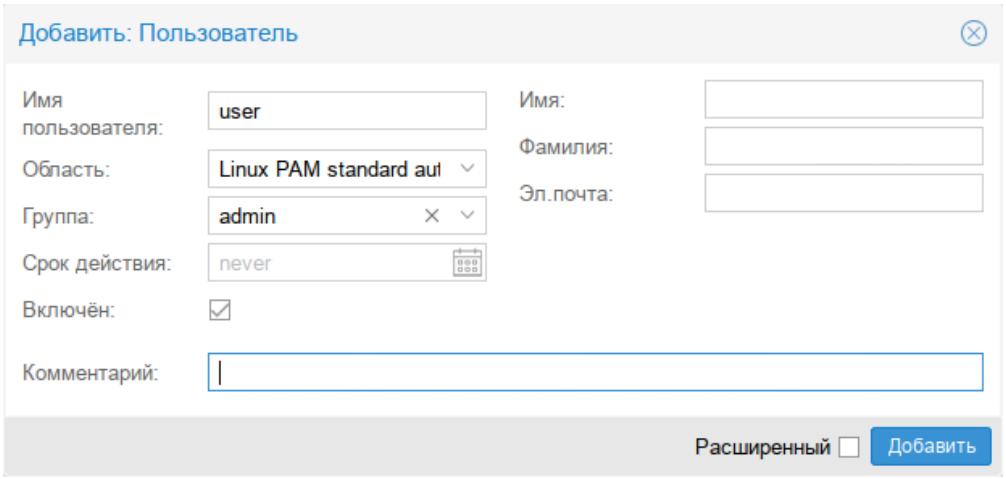


Рис. 195

4.14.1.2 Стандартная аутентификация Linux PAM

Область аутентификации PVE представляет собой хранилище паролей в стиле Unix (`/etc/pve/priv/shadow.cfg`). Пароль шифруется с использованием метода хеширования SHA-256.

Область создается по умолчанию, и, как и в случае с Linux PAM, для неё можно добавить требование двухфакторной аутентификации («Require TFA») и установить её в качестве области по умолчанию для входа в систему («По умолчанию») (*Рис. 196*).

Для добавления нового пользователя, необходимо в окне «Датацентр» → «Разрешения» → «Пользователи» нажать кнопку «Добавить». На Рис. 197 показано создание нового пользователя с использованием PVE аутентификации.

Конфигурация PVE аутентификации

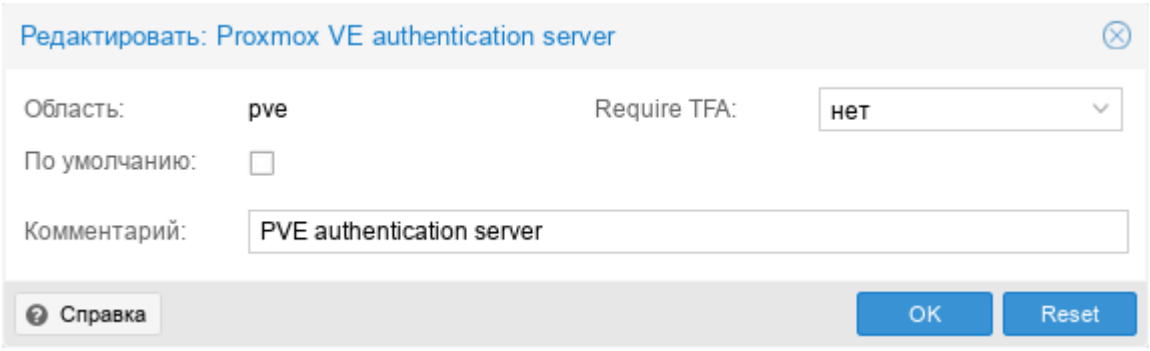


Рис. 196

Создание нового пользователя с использованием PVE аутентификации

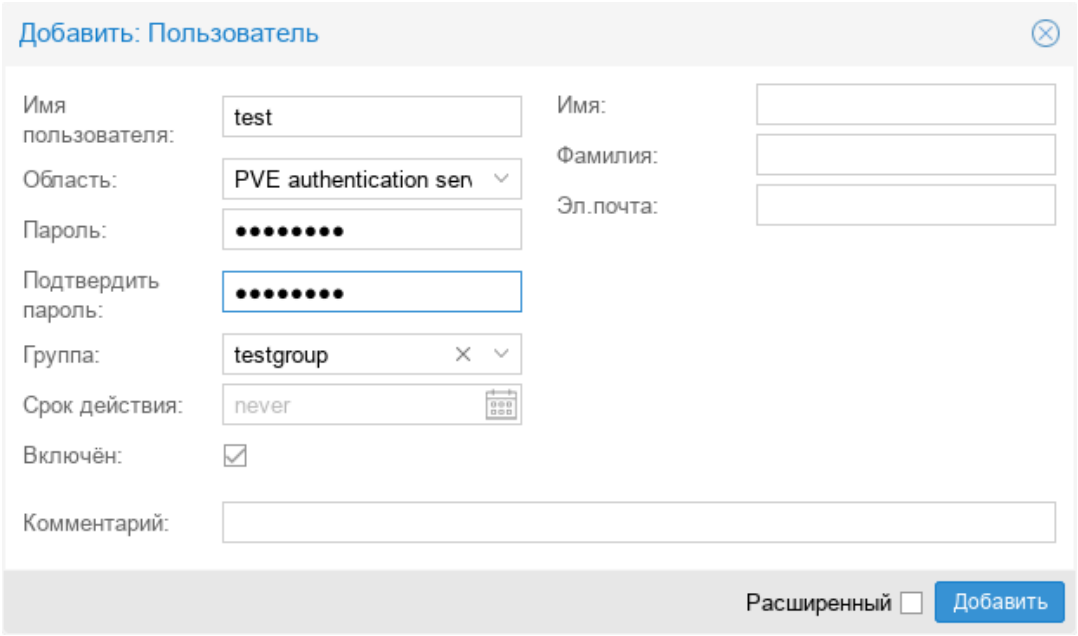


Рис. 197

Примеры использования командной строки для управления пользователями PVE:

– создать пользователя:

```
# pveum useradd testuser@pve -comment "Just a test"
```

– задать или изменить пароль:

```
# pveum passwd testuser@pve
```

– отключить пользователя:

```
# pveum usermod testuser@pve -enable 0
```

– создать новую группу:

```
# pveum groupadd testgroup
```

– создать новую роль:

```
# pveum roleadd PVE_Power-only -privs "VM.PowerMgmt VM.Console"
```

4.14.1.3 LDAP аутентификация

В данном разделе приведён пример настройки LDAP аутентификации для аутентификации на сервере FreeIPA. В примере используются следующие исходные данные:

– ipa.example.test, 192.168.0.113 – сервер FreeIPA;
– admin@example.test – учётная запись с правами чтения LDAP;
– pve – группа, пользователи которой имеют право аутентифицироваться в PVE.

Для настройки аутентификации FreeIPA необходимо выполнить следующие шаги:

1) создать область аутентификации LDAP. Для этого в разделе «Датацентр» → «Разрешения» → «Realms» нажать кнопку «Добавить» → «Сервер LDAP» (*Рис. 198*);

2) на вкладке «Общее» (*Рис. 199*) указать следующие данные:

– «Область» – идентификатор области;
– «Имя основного домена» (base_dn) – каталог, в котором выполняется поиск пользователей (dc=example,dc=test);
– «Имя пользовательского атрибута» (user_attr) – атрибут LDAP, содержащий имя пользователя, с которым пользователи будут входить в систему (uid);
– «По умолчанию» – установить область в качестве области по умолчанию для входа в систему;
– «Сервер» – IP-адрес или имя FreeIPA-сервера (ipa.example.test или 192.168.0.113);
– «Резервный сервер» (опционально) – адрес резервного сервера на случай, если основной сервер недоступен;
– «Порт» – порт, который прослушивает сервер LDAP (обычно 389 без ssl, 636 с ssl);
– «SSL» – использовать ssl;
– «Require TFA» – требовать двухфакторную аутентификацию.

3) на вкладке «Sync Options» (*Рис. 200*) заполнить параметры синхронизации (в скобках указаны значения, используемые в данном примере):

«Bind User» – имя пользователя (*uid=admin,cn=users,cn=accounts,dc=example,dc=test*);

– «Bind Password» – пароль пользователя;
– «E-Mail attribute (опционально);
– «Groupname attr. » – атрибут имени группы (*cn*);
– «User classes» – класс пользователей LDAP (*person*);
– «Group classes» – класс групп LDAP (*posixGroup*);

«User Filter» – фильтр пользователей (*memberOf=cn=pve,cn=groups,cn=accounts,dc=example,dc=test*);

– «Group Filter» – фильтр групп (*(|(cn=*pve*)(dc=ipa)(dc=example)(dc=test))*);

4) нажать кнопку «Добавить»;

5) выбрать добавленную область и нажать кнопку «Синхронизировать» («Sync») (Рис. 211);

6) указать, если необходимо, параметры синхронизации и нажать кнопку «Синхронизировать» («Sync») (Рис. 212).

В результате синхронизации пользователи и группы PVE будут синхронизированы с сервером FreeIPA LDAP. Сведения о пользователях и группах можно проверить на вкладках «Пользователи» и «Группы».

7) Настроить разрешения для группы/пользователя на вкладке «Разрешения».

Создать область аутентификации LDAP

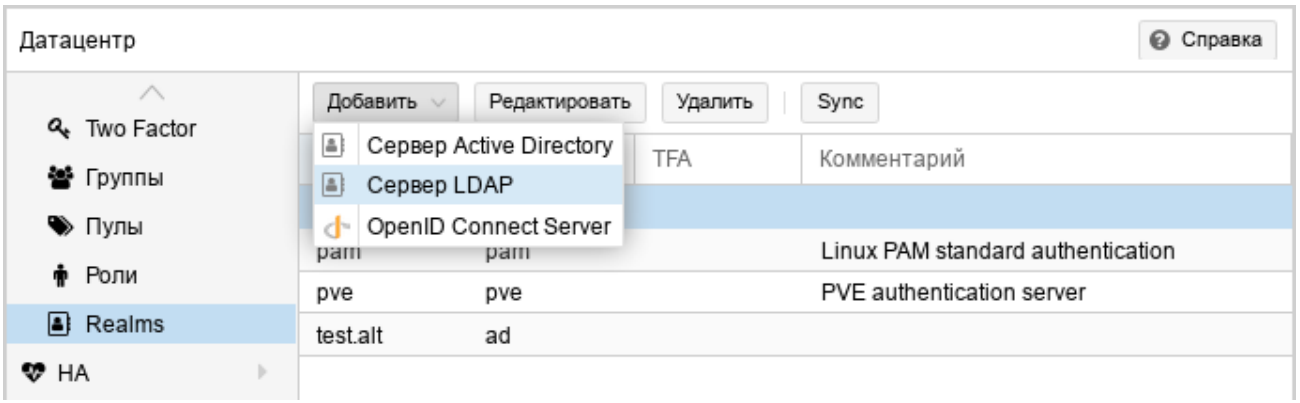


Рис. 198

Настройка LDAP аутентификации (вкладка «Общее»)

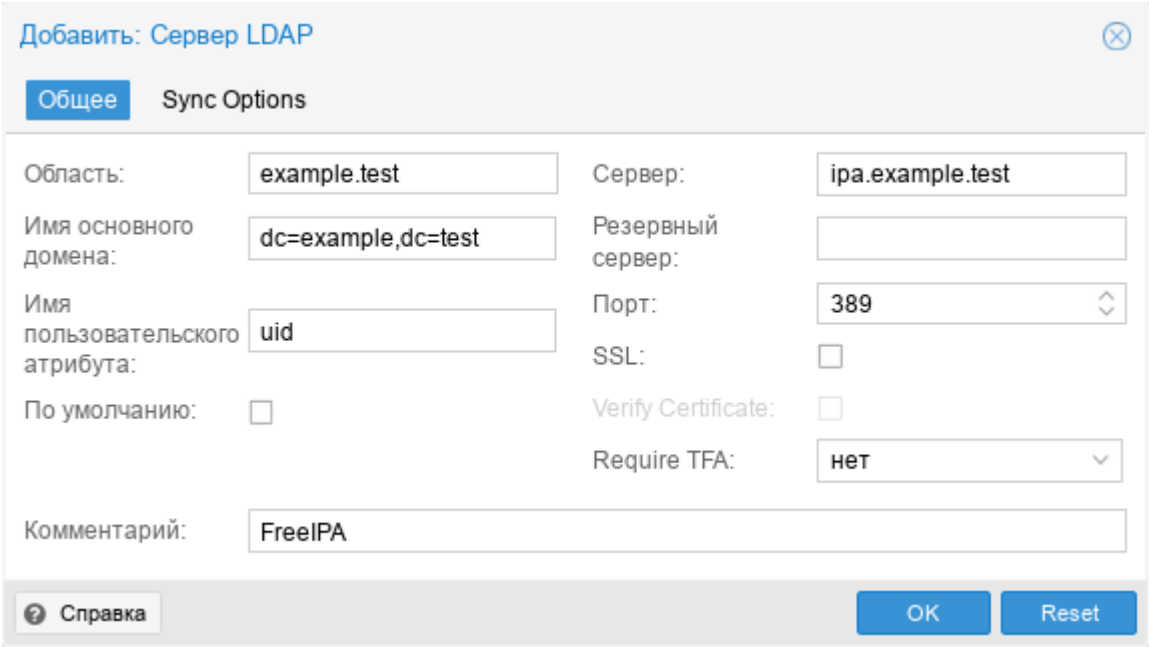


Рис. 199

Настройка LDAP аутентификации (вкладка «Sync Options»)

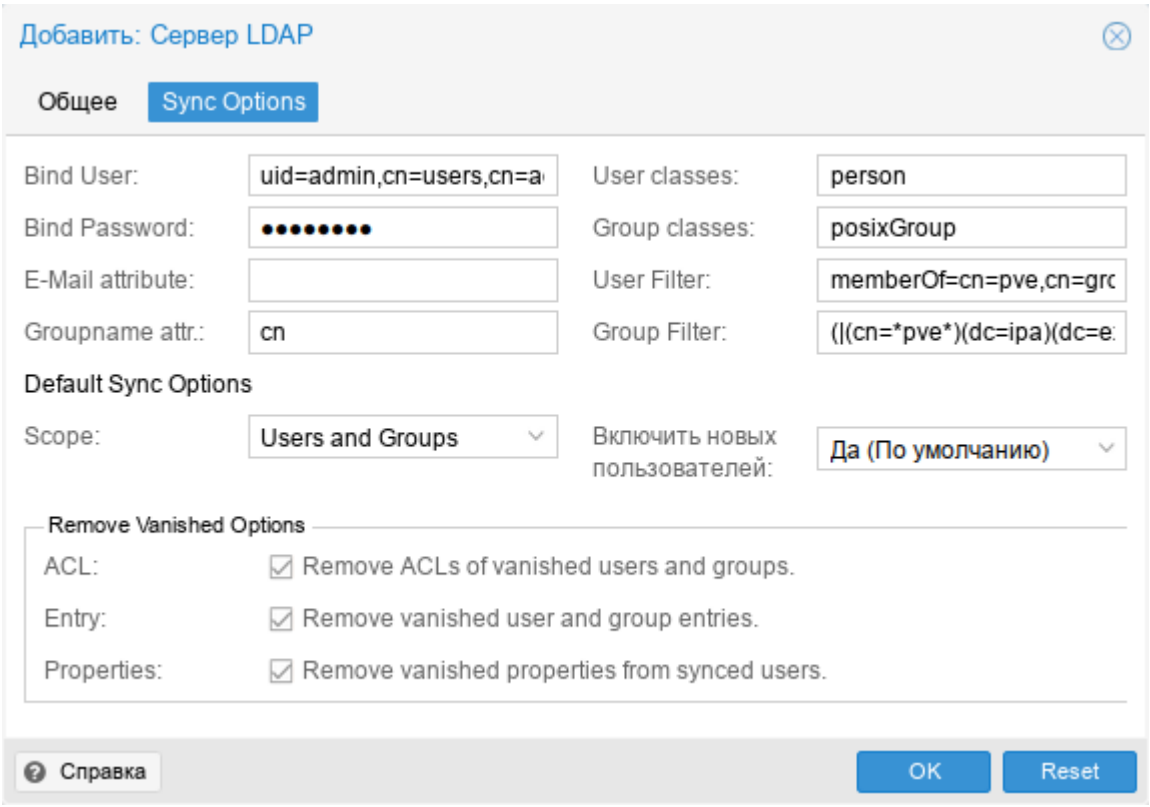


Рис. 200

Кнопка «Sync»

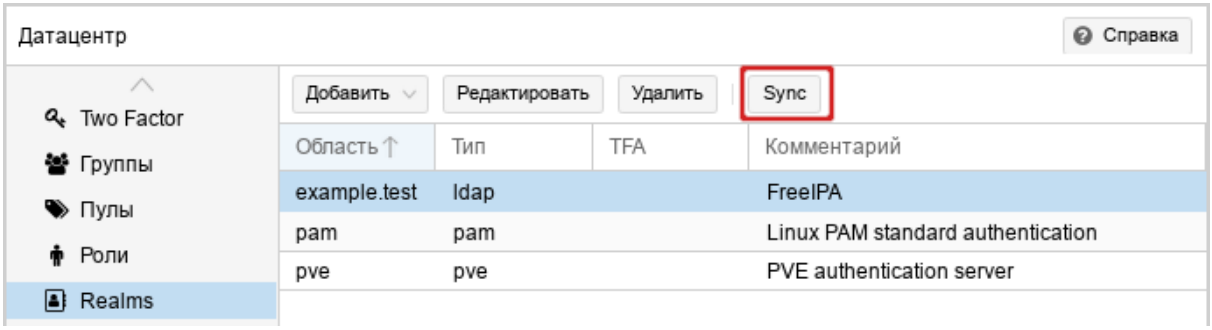


Рис. 201

Параметры синхронизации области аутентификации

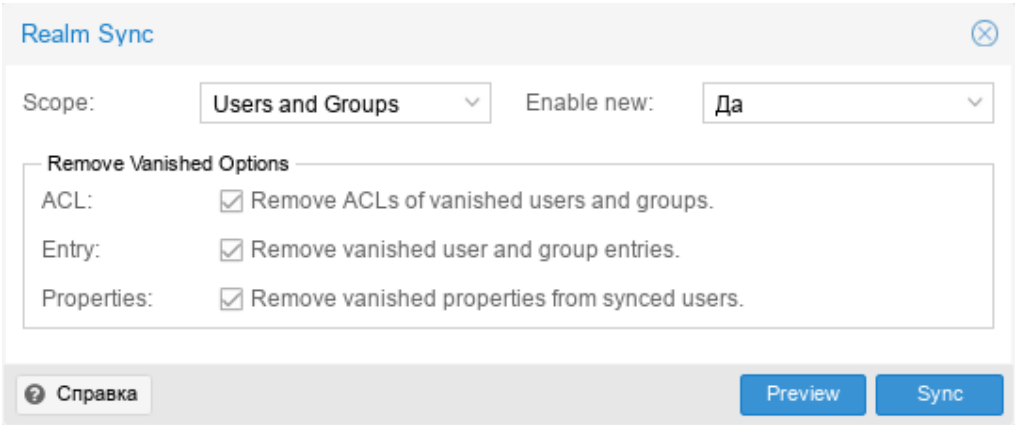


Рис. 202

П р и м е ч а н и е . Команда синхронизации пользователей и групп:

```
# pveum realm sync example.test
```

Для автоматической синхронизации пользователей и групп можно добавить команду синхронизации в планировщик задач.

4.14.1.4 AD аутентификация

В данном разделе приведён пример настройки аутентификации на сервере AD. В примере используются следующие исходные данные:

– dc.test.alt, 192.168.0.122 – сервер AD;
– administrator@test.alt – учётная запись администратора (для большей безопасности рекомендуется создать отдельную учетную запись с доступом только для чтения к объектам домена и не использовать учётную запись администратора);
– office – группа, пользователи которой имеют право аутентифицироваться в PVE.

Для настройки AD аутентификации необходимо выполнить следующие шаги:

1) создать область аутентификации LDAP. Для этого в разделе «Датацентр» → «Разрешения» → «Realms» нажать кнопку «Добавить» → «Сервер LDAP» (*Рис. 198*);

2) на вкладке «Общее» (*Рис. 203*) указать следующие данные:

– «Область» – идентификатор области;
– «Домен» – домен AD (*test.alt*);
– «По умолчанию» – установить область в качестве области по умолчанию для входа в систему;
– «Сервер» – IP-адрес или имя сервера AD (dc.test.alt или 192.168.0.122);
– «Резервный сервер» (опционально) – адрес резервного сервера на случай, если основной сервер недоступен;
– «Порт» – порт, который прослушивает сервер LDAP (обычно 389 без ssl, 636 с ssl);
– «SSL» – использовать ssl;
– «Require TFA» – требовать двухфакторную аутентификацию.

3) на вкладке «Sync Options» (*Рис. 204**Рис. 200*) заполнить параметры синхронизации (в скобках указаны значения, используемые в данном примере):

«Bind User» – имя пользователя (*cn=Administrator,cn=Users,dc=test,dc=alt*);

– «Bind Password» – пароль пользователя;
– «E-Mail attribute (опционально);
– «Groupname attr. » – атрибут имени группы (*cn*);
– «User classes» – класс пользователей LDAP;
– «Group classes» – класс групп LDAP;

– «User Filter» – фильтр пользователей (*(&(objectclass=user)(samaccountname=*)(MemberOf=CN=office,ou=OU,dc=TEST,dc=ALT))*);
– «Group Filter» – фильтр групп (*(|(cn=*office*)(dc=dc)(dc=test)(dc=alt))*);

4) нажать кнопку «Добавить»;

5) выбрать добавленную область и нажать кнопку «Синхронизировать» («Sync»);

6) указать, если необходимо, параметры синхронизации и нажать кнопку «Синхронизировать» («Sync») (Рис. 212).

В результате синхронизации пользователи и группы PVE будут синхронизированы с сервером AD. Сведения о пользователях и группах можно проверить на вкладках «Пользователи» и «Группы».

7) Настроить разрешения для группы/пользователя на вкладке «Разрешения».

Настройка AD аутентификации (вкладка «Общее»)

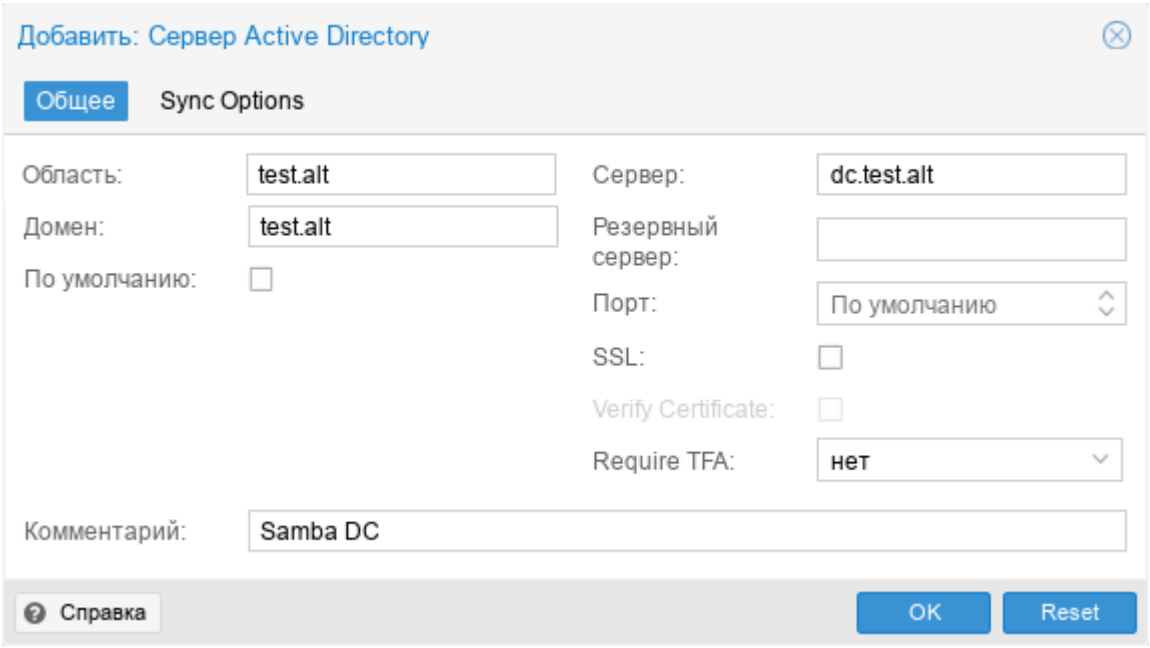


Рис. 203

Примечание. Команда синхронизации пользователей и групп:

```
# pveum realm sync test.alt
```

Для автоматической синхронизации пользователей и групп можно добавить команду синхронизации в планировщик задач.

Настройка AD аутентификации (вкладка «Sync Options»)

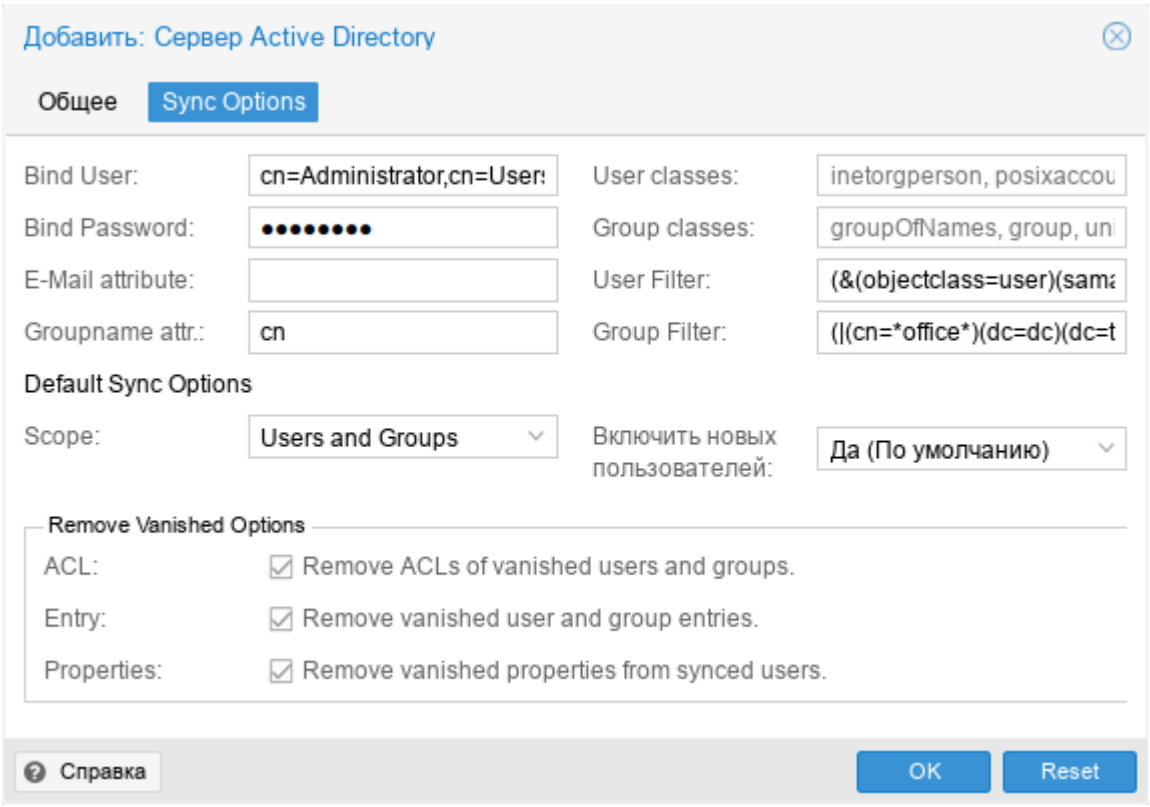


Рис. 204

4.14.2 Двухфакторная аутентификация

В PVE можно настроить двухфакторную аутентификацию двумя способами:

– Двухфакторная аутентификация (TFA) может потребоваться для области аутентификации (либо через TOTP, либо через YubiKey OTP). В этом случае вновь созданному пользователю необходимо сразу добавить свои ключи, так как нет возможности войти в систему без предъявления второго фактора. В случае с TOTP пользователи смогут изменить TOTP позже, при условии, что они смогут сначала войти в систему. Настроить принудительную двухфакторную аутентификацию можно при добавлении или редактировании области аутентификации (Рис. 205). Если в области аутентификации включена TFA, это становится требованием, и только пользователи с настроенным TFA смогут войти в систему.

– Пользователи могут сами настроить двухфакторную аутентификацию (*Рис. 206*), даже если она не требуется в области аутентификации (пункт TFA в выпадающем списке пользователя).

Настройка двухфакторной аутентификации при редактировании области

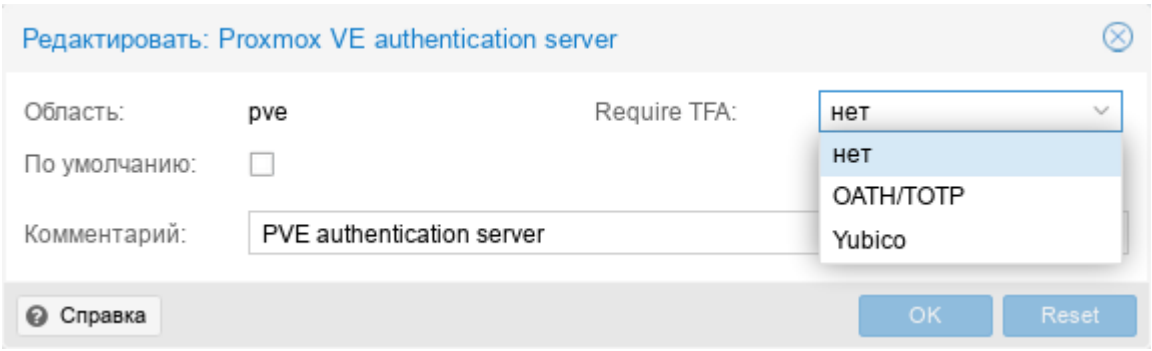


Рис. 205

Настройка двухфакторной аутентификации пользователем

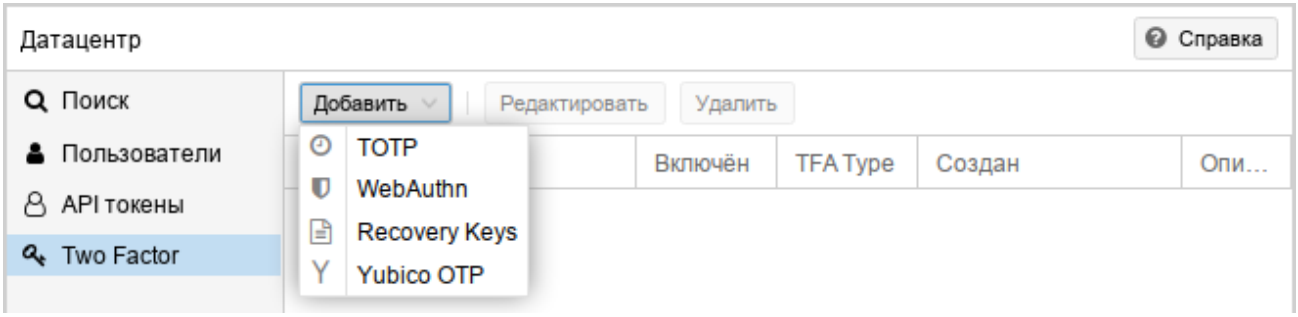


Рис. 206

При добавлении в области аутентификации доступны следующие методы двухфакторной аутентификации (*Рис. 205*):

– «OATH/TOTP» (основанная на времени OATH) – используется стандартный алгоритм HMAC-SHA1, в котором текущее время хэшируется с помощью настроенного пользователем ключа. Параметры временного шага и длины пароля настраиваются (*Рис. 207*).

  У пользователя может быть настроено несколько ключей (разделенных пробелами), и ключи могут быть указаны в Base32 (RFC3548) или в шестнадцатеричном представлении.

  PVE предоставляет инструмент генерации ключей (`oathkeygen`), который печатает случайный ключ в нотации Base32. Этот ключ можно использовать непосредственно с различными инструментами OTP, такими как инструмент командной строки `oathtool`, или приложении FreeOTP и в других подобных приложениях.

– «Yubico» (YubiKey OTP) – для аутентификации с помощью YubiKey необходимо настроить идентификатор API Yubico, ключ API и URL-адрес сервера проверки, а у пользователей должен быть доступен YubiKey. Чтобы получить идентификатор ключа от YubiKey, следует активировать YubiKey после подключения его через USB и скопировать первые 12 символов введенного пароля в поле ID ключа пользователя.

Основанная на времени OATH (TOTP)

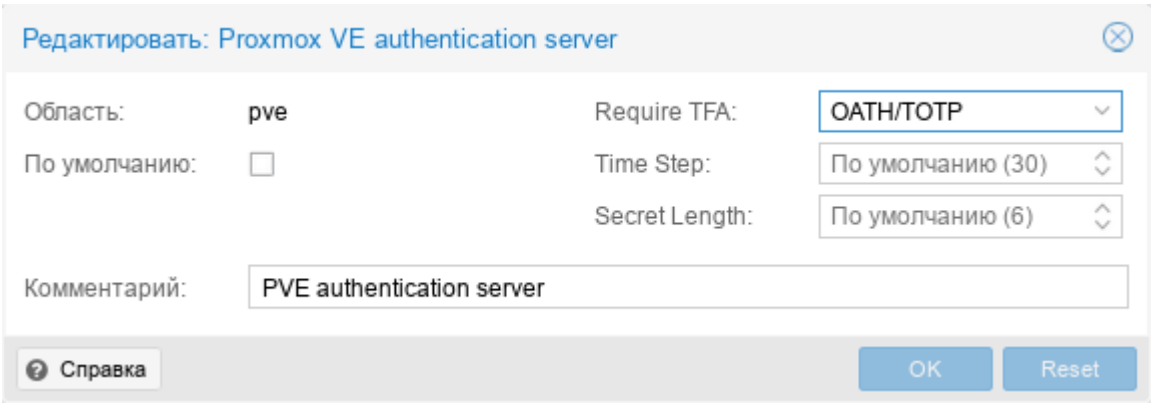


Рис. 207

В дополнение к TOTP и Yubikey OTP пользователям доступны следующие методы двухфакторной аутентификации (*Рис. 206*):

– «TOTP» (одноразовый пароль на основе времени) – для создания этого кода используется алгоритм одноразового пароля с учетом времени входа в систему (код меняется каждые 30 секунд);

– «WebAuthn» (веб-аутентификация) – реализуется с помощью различных устройств безопасности, таких как аппаратные ключи или доверенные платформенные модули (TPM). Для работы веб-аутентификации необходим сертификат HTTPS;

– «Recovery Keys» (одноразовые ключи восстановления) – список ключей, каждый из которых можно использовать только один раз. В каждый момент времени у пользователя может быть только один набор одноразовых ключей. Этот метод аутентификации идеально подходит для того, чтобы гарантировать, что пользователь получит доступ, даже если все остальные вторые факторы потеряны или повреждены.

Примечание. Пользователи могут использовать TOTP или WebAuthn в качестве второго фактора при входе в систему, только если область аутентификацию не применяет YubiKey OTP.

Примечание. Чтобы избежать ситуации, когда потеря электронного ключа навсегда блокирует доступ можно настроить несколько вторых факторов для одной учетной записи (*Рис. 208*).

Несколько настроенных вторых факторов для учётной записи

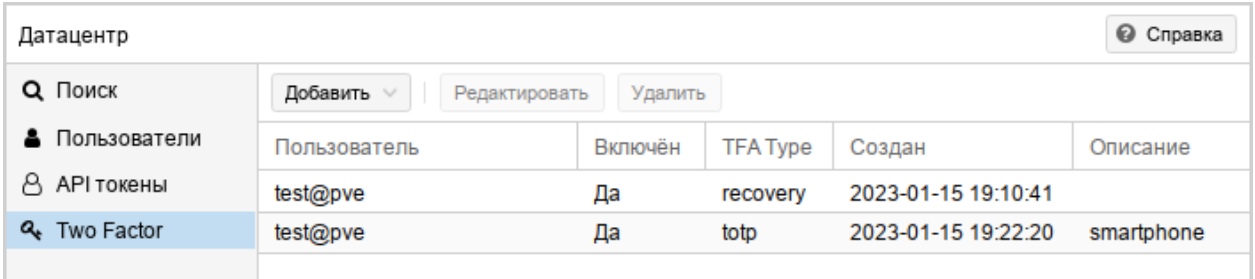


Рис. 208

Процедура добавления аутентификации «TOTP» показана на Рис. 209. При аутентификации пользователя будет запрашиваться второй фактор (Рис. 210).

PVE. Настройка аутентификации TOTP

Рис. 209

Запрос второго фактора (TOTP) при аутентификации пользователя в веб-интерфейсе

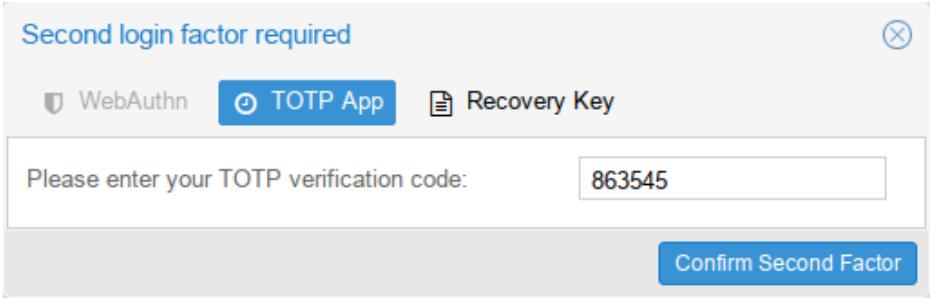


Рис. 210

При настройке аутентификации «Recovery Keys» необходимо создать набор ключей (Рис. 211). При аутентификации пользователя будет запрашиваться второй фактор (Рис. 212).

PVE. Настройка аутентификации Recovery Keys

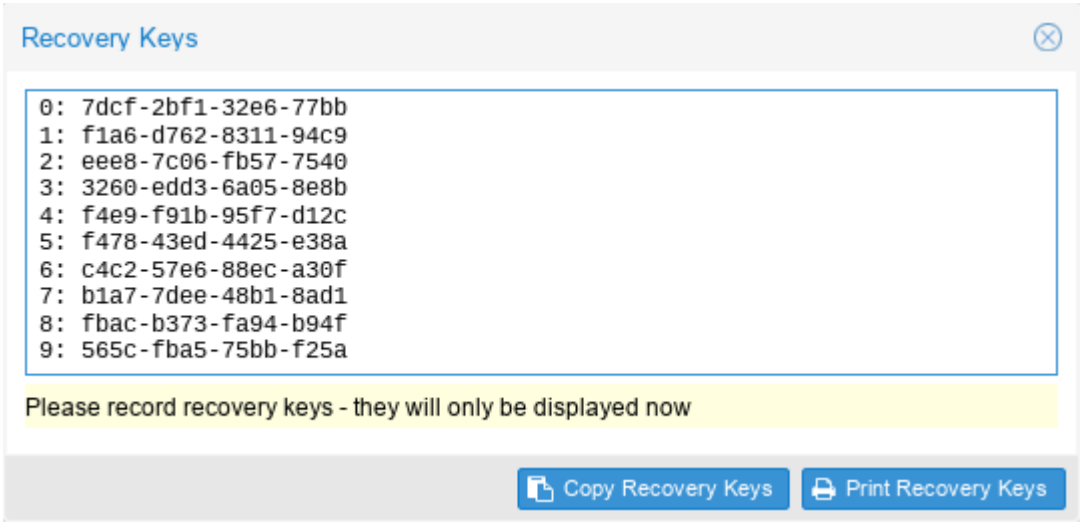


Рис. 211

Запрос второго фактора (Recovery Key) при аутентификации пользователя в веб-интерфейсе

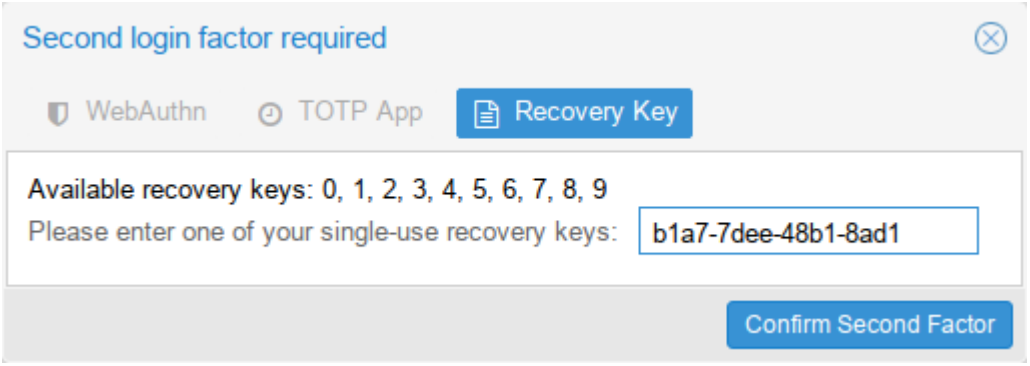


Рис. 212

4.14.3 Управление доступом

Чтобы пользователь мог выполнить какое-либо действие (например, просмотр, изменение или удаление ВМ), ему необходимо иметь соответствующие разрешения.

PVE использует систему управления разрешениями на основе ролей и путей. Запись в таблице разрешений позволяет пользователю или группе играть определенную роль при доступе к объекту или пути. Это означает, что такое правило доступа может быть представлено как тройка (путь, пользователь, роль) или (путь, группа, роль), причем роль содержит набор разрешенных действий, а путь представляет цель этих действий.

Роль – это список привилегий. В PVE предопределён ряд ролей:

– Administrator – имеет все привилегии;
– NoAccess – нет привилегий (используется для запрета доступа);
– PVEAdmin – все привилегии, кроме прав на изменение настроек системы (Sys.PowerMgmt, Sys.Modify, Realm.Allocate);

– PVEAuditor – доступ только для чтения;
– PVEDatastoreAdmin – создание и выделение места для резервного копирования и шаблонов;
– PVEDatastoreUser – выделение места для резервной копии и просмотр хранилища;
– PVEPoolAdmin – выделение пулов;
– PVESysAdmin – ACL пользователя, аудит, системная консоль и системные журналы;
– PVETemplateUser – просмотр и клонирование шаблонов;
– PVEUserAdmin – администрирование пользователей;
– PVEVMAdmin – управление ВМ;
– PVEVMUser – просмотр, резервное копирование, настройка CDROM, консоль ВМ, управление питанием ВМ.

Просмотреть список предопределенных ролей в веб-интерфейсе можно, выбрав «Датацентр» → «Разрешения»→«Роли» (*Рис. 213*).

Добавить новую роль можно как в веб-интерфейсе, так и в командной строке.

Список предопределенных ролей

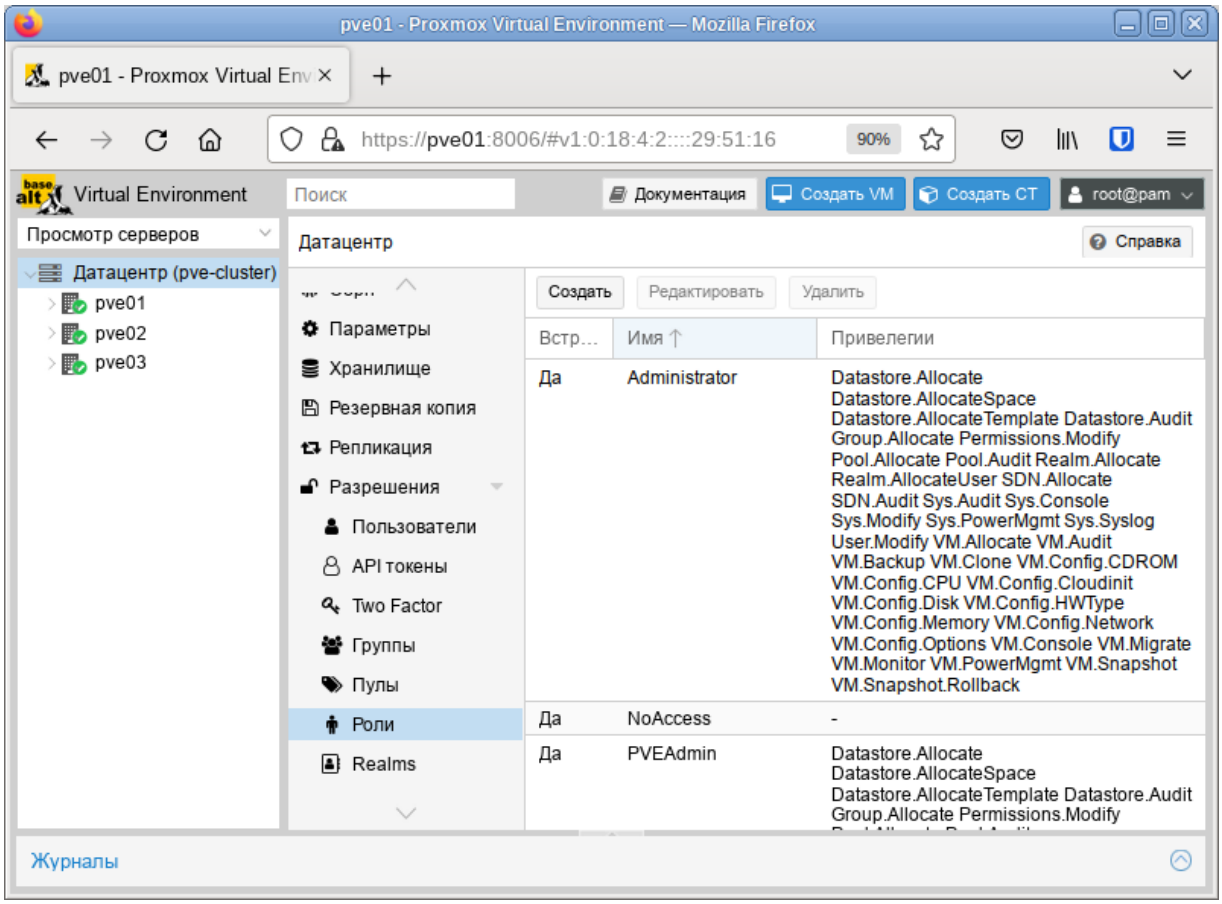


Рис. 213

Привилегия – это право на выполнение определенного действия. Для упрощения управления списки привилегий сгруппированы в роли, которые затем можно использовать в таблице раз-

решений. Привилегии не могут быть напрямую назначены пользователям, не будучи частью роли. Список используемых привилегий приведен в табл. 11.

Пул ресурсов – это набор ВМ, контейнеров и хранилищ. Пул ресурсов удобно использовать для обработки разрешений в случаях, когда определенные пользователи должны иметь контролируемый доступ к определенному набору ресурсов. Пулы ресурсов часто используются в тандеме с группами, чтобы члены группы имели разрешения на набор машин и хранилищ.

Т а б л и ц а 11 –Привилегии используемые в PVE

Привилегия	Описание
Привилегии узла/системы	
Permissions.Modify	Изменение прав доступа
Sys.PowerMgmt	Управление питанием узла (запуск, остановка, сброс, выключение)
Sys.Console	Консольный доступ к узлу
Sys.Syslog	Просмотр Syslog
Sys.Audit	Просмотр состояния/конфигурации узла, конфигурации кластера Corosync и конфигурации HA
Sys.Modify	Создание/удаление/изменение параметров сети узла
Group.Allocate	Создание/удаление/изменение групп
Pool.Allocate	Создание/удаление/изменение пулов
Realm.Allocate	Создание/удаление/изменение областей аутентификации
Realm.AllocateUser	Назначение пользователю области аутентификации
User.Modify	Создание/удаление/изменение пользователя
Права, связанные с ВМ	
VM.Allocate	Создание/удаление ВМ
VM.Migrate	Миграция ВМ на альтернативный сервер в кластере
VM.PowerMgmt	Управление питанием (запуск, остановка, сброс, выключение)
VM.Console	Консольный доступ к ВМ
VM.Monitor	Доступ к монитору виртуальной машины (kvm)
VM.Backup	Резервное копирование/восстановление ВМ
VM.Audit	Просмотр конфигурации ВМ
VM.Clone	Клонирование ВМ
VM.Config.Disk	Добавление/изменение/удаление дисков ВМ
VM.Config.CDROM	Извлечь/изменить CDROM
VM.Config.CPU	Изменение настроек процессора
VM.Config.Memory	Изменение настроек памяти
VM.Config.Network	Добавление/изменение/удаление сетевых устройств
VM.Config.HWType	Изменение типа эмуляции
VM.Config.Options	Изменение любой другой конфигурации ВМ
VM.Snapshot	Создание/удаление снимков ВМ
Права, связанные с хранилищем	
Datastore.Allocate	Создание/удаление/изменение хранилища данных
Datastore.AllocateSpace	Выделить место в хранилище
Datastore.AllocateTemplate	Размещение/загрузка шаблонов контейнеров и ISO-образов
Datastore.Audit	Просмотр хранилища данных

Права доступа назначаются объектам, таким как ВМ, хранилища или пулы ресурсов. PVE использует файловую систему как путь к этим объектам. Эти пути образуют естественное дерево, и права доступа более высоких уровней (более короткий путь) необязательно распространяются вниз по этой иерархии. Примеры:

– /nodes/{node} – доступ к серверам PVE;
– /vms – все ВМ;
– /vms/{vmid} – доступ к определенным ВМ;
– /storage/{storeid} – доступ к хранилищам;
– /access/groups – администрирование групп;

/access/realms/{realmid} – административный доступ.

Для назначения разрешений необходимо в окне «Датацентр» → «Разрешения» нажать кнопку «Добавить», в выпадающем меню выбрать «Разрешения группы», если разрешения назначаются группе пользователей, или «Разрешения пользователя», если разрешения назначаются пользователю. Далее в открывшемся окне (Рис. 214) выбрать путь, группу и роль и нажать кнопку «Добавить».

Добавление разрешений группе

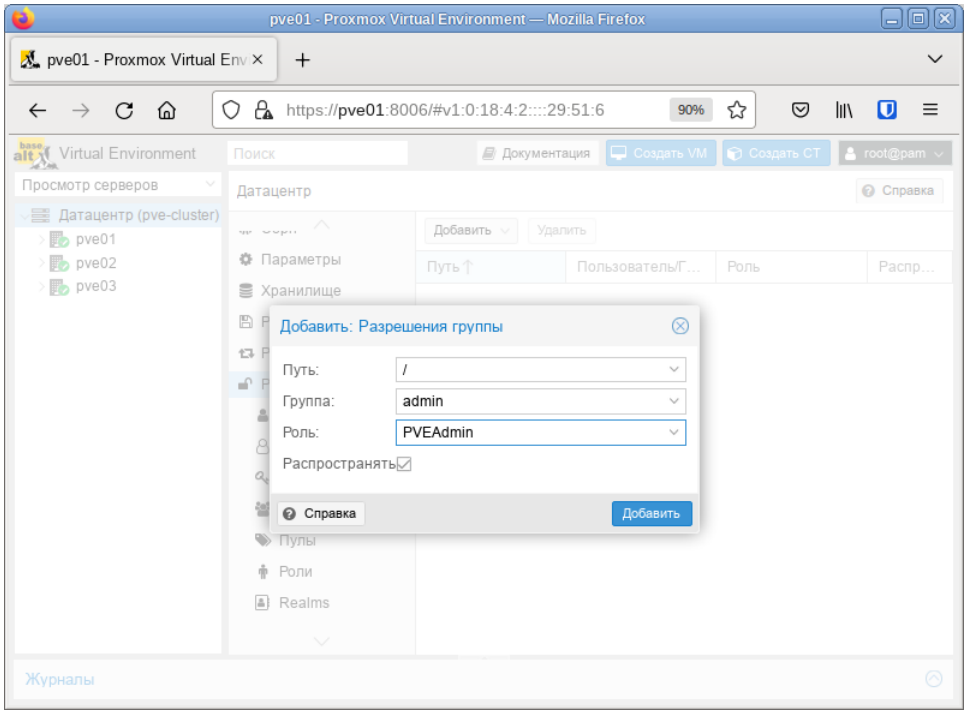


Рис. 214

5 УПРАВЛЕНИЕ ВИРТУАЛИЗАЦИЕЙ НА ОСНОВЕ LIBVIRT

5.1 Установка и настройка libvirt

libvirt – это набор инструментов, предоставляющий единый API к множеству различных технологий виртуализации.

Кроме управления виртуальными машинами/контейнерами libvirt поддерживает управление виртуальными сетями и управление хранением образов.

Для управления из консоли разработан набор утилит virt-install, virt-clone, virsh и других. Для управления из графической оболочки можно воспользоваться virt-manager.

Любой виртуальный ресурс, необходимый для создания ВМ (compute, network, storage) представлен в виде объекта в libvirt. За процесс описания и создания этих объектов отвечает набор различных XML-файлов. Сама ВМ в терминологии libvirt называется доменом (domain). Это тоже объект внутри libvirt, который описывается отдельным XML-файлом.

При первоначальной установке и запуске libvirt по умолчанию создает мост (bridge) virbr0 и его минимальную конфигурацию. Этот мост не будет подключен ни к одному физическому интерфейсу, однако, может быть использован для связи ВМ внутри одного гипервизора.

Примечание. Компоненты libvirt будут установлены в систему, если при установке дистрибутива выбрать профиль «Базовая виртуализация».

Примечание. На этапе «Подготовка диска» рекомендуется выбрать «Generic Server KVM/Docker/LXD/Podman/CRI-O/PVE (large /var)».

Если же развертывание libvirt происходит в уже установленной системе на базе Десятой платформы, достаточно любым штатным способом (apt-get, aptitude, synaptic) установить пакеты:

```
# apt-get update
# apt-get install libvirt virt-install
```

Добавить службу в автозапуск и запустить ее:

```
# systemctl enable --now libvirtd
```

Для непривилегированного доступа (не root) к управлению libvirt, нужно добавить пользователя в группу vmusers:

```
# gpasswd -a user vmusers
```

Сервер виртуализации использует следующие каталоги хостовой файловой системы:

- `/etc/libvirt/` – каталог с файлами конфигурации libvirt;
- `/var/lib/libvirt/` – рабочий каталог сервера виртуализации libvirt;
- `/var/log/libvirt` – файлы журналов libvirt.

5.2 Утилиты управления

Основные утилиты командной строки для управления ВМ:

- qemu-img – управление образами дисков ВМ. Позволяет выполнять операции по созданию образов различных форматов, конвертировать файлы-образы между этими форматами, получать информацию об образах и объединять снимки ВМ для тех форматов, которые это поддерживают;
- virsh – консольный интерфейс управления ВМ, виртуальными дисками и виртуальными сетями;
- virt-clone –клонирование ВМ;
- virt-convert –конвертирования ВМ между различными форматами и программно-аппаратными платформами;
- virt-image – создание ВМ по их XML описанию;
- virt-install – создание ВМ с помощью опций командной строки;
- virt-xml – редактирование XML-файлов описаний ВМ.

5.2.1 Утилита Virsh

virsh – утилита для командной строки, предназначенная для управления ВМ и гипервизорами KVM.

virsh использует libvirt API и служит альтернативой графическому менеджеру виртуальных машин (virt-manager).

С помощью virsh можно сохранять состояние ВМ, переносить ВМ между гипервизорами и управлять виртуальными сетями.

В табл. 12 и табл. 13 приведены основные параметры утилиты командной строки virsh. Получить список доступных команд или параметров, можно используя команду: $ `virsh help`.

Т а б л и ц а 12 – Команды управления виртуальными машинами

Команда	Описание
`help`	Краткая справка
`list`	Просмотр всех ВМ
`dumpxml`	Вывести файл конфигурации XML для заданной ВМ
`create`	Создать ВМ из файла конфигурации XML и ее запуск
`start`	Запустить неактивную ВМ
`destroy`	Принудительно остановить работу ВМ
`define`	Определяет файл конфигурации XML для заданной ВМ
`domid`	Просмотр идентификатора ВМ
`domuuid`	Просмотр UUID ВМ

Команда	Описание
dominfo	Просмотр сведений о ВМ
domname	Просмотр имени ВМ
domstate	Просмотр состояния ВМ
quit	Закрыть интерактивный терминал
reboot	Перезагрузить ВМ
restore	Восстановить сохраненную в файле ВМ
resume	Возобновить работу приостановленной ВМ
save	Сохранить состояние ВМ в файл
shutdown	Корректно завершить работу ВМ
suspend	Приостановить работу ВМ
undefine	Удалить все файлы ВМ
migrate	Перенести ВМ на другой узел

Т а б л и ц а 13 – Параметры управления ресурсами ВМ и гипервизора

Команда	Описание
setmem	Определяет размер выделенной ВМ памяти
setmaxmem	Ограничивает максимально доступный гипервизору объ-ем памяти
setvcpus	Изменяет число предоставленных ВМ виртуальных про-цессоров
vcpuinfo	Просмотр информации о виртуальных процессорах
vcpupin	Настройка соответствий виртуальных процессоров
domblkstat	Просмотр статистики блочных устройств для работающей ВМ
domifstat	Просмотр статистики сетевых интерфейсов для работаю-щей ВМ
attach-device	Подключить определенное в XML-файле устройство к ВМ
attach-disk	Подключить новое дисковое устройство к ВМ
attach-interface	Подключить новый сетевой интерфейс к ВМ
detach-device	Отключить устройство от ВМ (принимает те же опреде-ления XML, что и attach-device)
detach-disk	Отключить дисковое устройство от ВМ
detach-interface	Отключить сетевой интерфейс от ВМ

5.2.2 Утилита virt-install

virt-install – это инструмент для создания ВМ, основанный на командной строке.

Далее подробно рассматриваются возможности создания ВМ при помощи утилиты командной строки virt-install. В табл. 14 приведено описание только наиболее часто используемых опций команды virt-install. Описание всех доступных опций можно получить, выполнив команду:

```
$ man virt-install
```

Утилита virt-install поддерживает как графическую установку операционных систем при помощи VNC и Spice, так и текстовую установку через последовательный порт. Гостевая система может быть настроена на использование нескольких дисков, сетевых интерфейсов, аудиоустройств и физических USB- и PCI-устройств.

Установочный носитель может располагаться как локально, так и удаленно, например, на NFS, HTTP или FTP серверах. В последнем случае virt-install получает минимальный набор файлов для запуска установки и позволяет установщику получить отдельные файлы. Также поддерживается загрузка по сети (PXE) и создание виртуальной машины/контейнера без установки операционной системы.

Утилита virt-install поддерживает большое число опции, позволяющих создать полностью независимую ВМ, готовую к работе, что хорошо подходит для автоматизации установки ВМ.

Т а б л и ц а 14 – Параметры команды virt-install

Команда	Описание
`-n NAME, --name=NAME`	Имя новой ВМ. Это имя должно быть уникально внутри одного гипервизора
`--memory MEMORY`	Определяет размер выделенной ВМ памяти (в МБ)

`--vcpus VCPUS`	Определяет количество виртуальных ЦПУ. Например:
	`--vcpus 5`
	`--vcpus 5,maxvcpus=10,cpuset=1-4,6,8`
	`--vcpus sockets=2,cores=4,threads=2`

`--cpu CPU`	Модель ЦП и его характеристики. Например:
	`--cpu coreduo,+x2apic`
	`--cpu host-passthrough`
	`--cpu host`

`--metadata METADATA`	Метаданные ВМ
Метод установки	
`--cdrom CDROM`	Установочный CD-ROM. Может указывать на файл ISO-образа или на устройство чтения CD/DVD-дисков
`-l LOCATION, --location LOCATION`	Источник установки, например, https://host/path
`--pxe`	Выполнить загрузку из сети используя протокол PXE
`--import`	Пропустить установку ОС, и создать ВМ на основе существующего образа диска

Команда	Описание

`--boot BOOT`	Параметры загрузки ВМ. Например:
	`--boot hd,cdrom,menu=on`
	`--boot init=/sbin/init (для контейнеров)`

`--os-type=DISTRO_TYPE`	Оптимизирует настройки ВМ для заданного типа ОС
`--os-variant=DISTRO_VARIANT`	Дополнительная оптимизация ВМ для конкретного варианта ОС

`--disk DISK`	Настройка пространства хранения данных. Например:
	`--disk size=10 (новый образ на 10 ГБ в выбранном по умолчанию месте)`
	`--disk /my/existing/disk,cache=none`
	`--disk device=cdrom,bus=scsi`
	`--disk=?`

`-w NETWORK, --network NETWORK`	Конфигурация сетевого интерфейса ВМ. Например:
	`--network bridge=mybr0`
	`--network network=my_libvirt_virtual_net`
	`--network network=mynet,model=virtio,mac=00:11...`
	`--network none`

`--graphics GRAPHICS`	Настройки экрана ВМ. Например:
	`--graphics spice`
	`--graphics vnc,port=5901,listen=0.0.0.0`
	`--graphics none`

`--input INPUT`	Конфигурация устройства ввода. Например:
	`--input tablet`
	`--input keyboard,bus=usb`

`--hostdev HOSTDEV`	Конфигурация физических USB/PCI и других устройств хоста для совместного использования ВМ
`-filesystem FILESYSTEM`	Передача каталога хоста гостевой системе. Например: `--filesystem /my/source/dir,/dir/in/guest`
Параметры платформы виртуализации	
`-v, --hvm`	Эта ВМ должна быть полностью виртуализированной
`-p, --paravirt`	Эта ВМ должна быть паравиртуализированной
`--container`	Тип ВМ – контейнер
`--virt-type VIRT_TYPE`	Тип гипервизора (kvm, qemu и т.п.)
`--arch ARCH`	Имитируемая архитектура процессора
`--machine MACHINE`	Имитируемый тип компьютера
Прочие параметры	

Команда	Описание
`--autostart`	Запускать домен автоматически при запуске хоста
`--transient`	Создать временный домен
`--noautoconsole`	Не подключаться к гостевой консоли автоматически
`-q, --quiet`	Подавлять вывод (за исключением ошибок)
`-d, --debug`	Вывести отладочные данные

5.2.3 Утилита qemu-img

qemu-img – инструмент для манипулирования с образами дисков машин QEMU.

Использование:

```
qemu-img command [command options]
```

Для манипуляции с образами используются следующие команды:

- create – создание нового образа диска;
- check – проверка образа диска на ошибки;
- convert – конвертация существующего образа диска в другой формат;
- info – получение информации о существующем образе диска;
- snapshot – управляет снимками состояний (snapshot) существующих образов дисков;
- commit – записывает произведенные изменения на существующий образ диска;
- rebase – создает новый базовый образ на основании существующего.

qemu-img работает со следующими форматами:

- raw – простой формат для дисковых образов, обладающий отличной переносимостью на большинство технологий виртуализации и эмуляции. Только непосредственно записанные секторы будут занимать место на диске. Действительный объем пространства, занимаемый образом, можно определить с помощью команд qemu-img info или ls -ls;
- qcow2 – формат QEMU. Этот формат рекомендуется использовать для небольших образов (в частности, если файловая система не поддерживает фрагментацию), дополнительного шифрования AES, сжатия zlib и поддержки множества снимков ВМ;
- qcow – старый формат QEMU. Используется только в целях обеспечения совместимости со старыми версиями;
- cow – формат COW (Copy On Write). Используется только в целях обеспечения совместимости со старыми версиями;
- vmdk – формат образов, совместимый с VMware 3 и 4;

– cloop – формат CLOOP (Compressed Loop). Его единственное применение состоит в обеспечении повторного использования сжатых напрямую образов CD-ROM, например, Knoppix CD-ROM.

Команда получения сведений о дисковом образе:

```
# qemu-img info /var/lib/libvirt/images/alt-server.qcow2
image: /var/lib/libvirt/images/alt-server.qcow2
file format: qcow2
virtual size: 20 GiB (21474836480 bytes)
disk size: 3.32 MiB
cluster_size: 65536
Format specific information:
    compat: 1.1
    compression type: zlib
    lazy refcounts: true
    refcount bits: 16
    corrupt: false
    extended l2: false
```

В результате будут показаны сведения о запрошенном образе, в том числе зарезервированный объем на диске, а также информация о снимках ВМ.

Команда создания образа для жесткого диска (динамически расширяемый):

```
# qemu-img create -f qcow2 /var/lib/libvirt/images/hdd.qcow2 20G
```

Команда конвертирования образ диска из формата raw в qcow2:

```
# qemu-img convert -f raw -O qcow2 disk_hd.img disk_hd.qcow2
```

5.2.4 Менеджер виртуальных машин virt-manager

Менеджер виртуальных машин virt-manager предоставляет графический интерфейс для доступа к гипервизорам и ВМ в локальной и удаленных системах. С помощью virt-manager можно создавать ВМ. Кроме того, virt-manager выполняет управляющие функции:

– выделение памяти;
– выделение виртуальных процессоров;
– мониторинг производительности;
– сохранение и восстановление, приостановка и возобновление работы, запуск и завершение работы виртуальных машин;
– доступ к текстовой и графической консоли;
– автономная и живая миграция.

Для запуска менеджера виртуальных машин, в меню приложений необходимо выбрать «Система»→ «Менеджер виртуальных машин» («Manage virtual machines»).

П р и м е ч а н и е . На управляющей машине должен быть установлен пакет virt-manager.

В главном окне менеджера (Рис. 215), при наличии подключения к гипервизору, будут показаны все запущенные ВМ. Двойной щелчок на имени ВМ открывает ее консоль.

Главное окно менеджера виртуальных машин

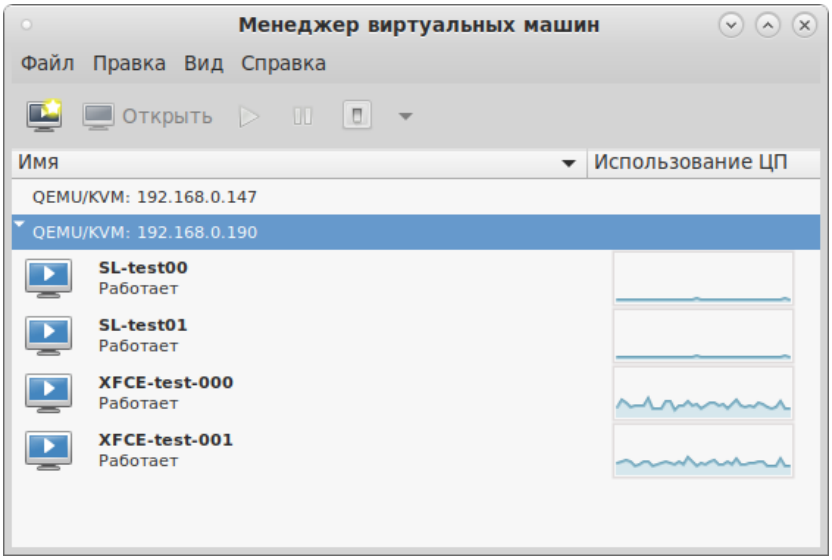


Рис. 215

5.3 Подключение к гипервизору

5.3.1 Управление доступом к libvirt через SSH

В дополнение к аутентификации SSH также необходимо определить управление доступом для службы Libvirt в хост-системе (Рис. 216).

Доступ к libvirt с удаленного узла

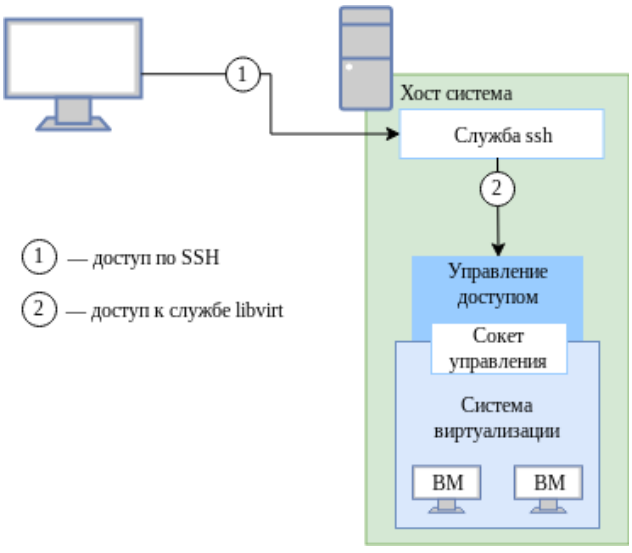


Рис. 216

Для настройки подключения к удаленному серверу виртуализации на узле, с которого будет производиться подключение, необходимо сгенерировать SSH-ключ и скопировать его публичную часть на сервер. Для этого с правами пользователя, от имени которого будет создаваться подключение, требуется выполнить в консоли следующие команды:

```
$ ssh-keygen -t ed25519
$ ssh-copy-id user@192.168.0.147
```

где 192.168.0.147 – IP-адрес сервера с libvirt.

В результате получаем возможность работы с домашними каталогами пользователя user на сервере с libvirt.

Для доступа к libvirt достаточно добавить пользователя user в группу vmusers на сервере, либо скопировать публичный ключ пользователю root и подключаться к серверу по ssh от имени root – root@server

5.3.2 Подключение к сессии гипервизора с помощью virsh

Команда подключения к гипервизору:

```
virsh -c URI
```

Если параметр URI не задан, то libvirt попытается определить наиболее подходящий гипервизор.

Параметр URI может принимать следующие значения:

- qemu:///system – подключиться к службе, которая управляет KVM/QEMU-доменами и запущена под root. Этот вариант используется по умолчанию для пользователей virt-manager;
- qemu:///session – подключиться к службе, которая управляет KVM/QEMU-доменами и запущена от имени непривилегированного пользователя;
- lxc:/// – подключиться к гипервизору для создания LXC контейнеров (должен быть установлен пакет libvirt-lxc).

Чтобы установить соединение только для чтения, к приведенной выше команде следует добавить опцию `--readonly`.

Пример создания локального подключения:

```
$ virsh -c qemu:///system list --all
ID   Имя          Состояние
----------------------------
 -    alt-server   выключен
```

Примечание. Чтобы постоянно не вводить `-c qemu:///system` можно добавить:

```
export LIBVIRT_DEFAULT_URI=qemu:///system
```

Подключение к удаленному гипервизору QEMU через протокол SSH:

```
$ virsh -c qemu+ssh://user@192.168.0.147/system
```

```
Добро пожаловать в virsh — интерактивный терминал виртуализации.

Введите «help» для получения справки по командам
         «quit», чтобы завершить работу и выйти.

virsh #
```

где user – имя пользователя на удаленном хосте, который входит в группу vmusers. 192.168.0.147 – IP-адрес или имя хоста виртуальных машин.

5.3.3 Настройка соединения с удаленным гипервизором в virt-manager

На управляющей системе можно запустить virt-manager, выполнив следующую команду:

```
virt-manager -c qemu+ssh://user@192.168.0.147/system
```

где user – имя пользователя на удаленном хосте, который входит в группу vmusers. 192.168.0.147 – IP-адрес или имя хоста виртуальных машин.

virt-manager позволяет управлять несколькими удаленными хостами ВМ.

Подключение virt-manager к удаленным хостам, также, можно настроить и в графическом интерфейсе менеджера виртуальных машин. Для создания нового подключения необходимо в меню менеджера выбрать «Файл» → «Добавить соединение…».

В открывшемся окне необходимо выбрать сессию гипервизора, отметить пункт «Подключиться к удаленному хосту с помощью SSH», ввести имя пользователя и адрес сервера и нажать кнопку «Подключиться» (Рис. 217).

Окно соединений менеджера виртуальных машин

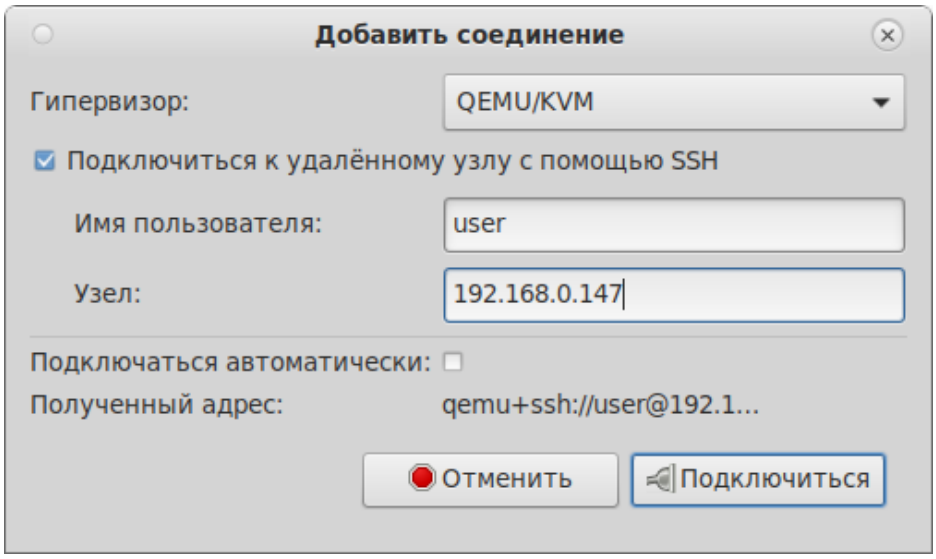


Рис. 217

5.4 Создание виртуальных машин

Наиболее важным этапом в процессе использования виртуализации является создание ВМ. Именно при создании ВМ задается используемый тип виртуализации, способы доступа к ВМ, подключение к локальной сети и другие характеристики виртуального оборудования.

Установка ВМ может быть запущена из командной строки с помощью программ virsh и virt-install или из пользовательского интерфейса программы virt-manager.

5.4.1 Создание виртуальной машины на основе файла конфигурации (утилита virsh)

ВМ могут быть созданы из файлов конфигурации. Для этого конфигурация ВМ должна быть описана в XML формате.

Команда создания ВМ из XML файла:

```
$ virsh -c qemu:///system create guest.xml
Domain 'altK' created from guest.xml
```

Для получения файла конфигурации можно сделать копию существующего XML-файла ранее созданной ВМ, или использовать опцию `dumpxml`:

```
virsh dumpxml <domain>
```

Эта команда выводит XML-файл конфигурации ВМ в стандартный вывод (stdout). Можно сохранить эти данные, отправив вывод в файл.

Пример передачи вывода в файл `guest.xml`:

```
$ virsh -c qemu:///system dumpxml alt-server > guest.xml
```

Можно отредактировать этот файл конфигурации, чтобы настроить дополнительные устройства или развернуть дополнительные ВМ.

5.4.2 Создание ВМ с помощью virt-install

Минимальные требуемые опции для создания ВМ: `--name`, `--ram`, хранилище (`--disk`, `--filesystem` или `--nodisks`) и опции установки.

Чтобы использовать команду virt-install, необходимо сначала загрузить ISO-образ той ОС, которая будет устанавливаться.

Команда создания ВМ:

```
# virt-install --connect qemu:///system
--name alt-server \
--os-type=linux \
--os-variant=alt10.0 \
--cdrom /var/lib/libvirt/boot/alt-server-10.0-x86_64.iso \
--graphics vnc\
--disk pool=default,size=20,bus=virtio,format=qcow2 \
```

```
--ram 2048 \
--vcpus=2 \
--network network=default \
--hvm \
--virt-type=kvm
```

где

--name alt-server – название ВМ;

--os-type=linux – тип ОС;

--os-variant=alt10.0 – версия ОС;

--cdrom /var/lib/libvirt/boot/alt-server-10.0-x86_64.iso – путь к ISO-образу установочного диска ОС;

--graphics vnc – графическая консоль;

--disk pool=default,size=20,bus=virtio,format=qcow2 – ВМ будет создана в пространстве хранения объемом 20 ГБ, которое автоматически выделяется из пула хранилищ default. Образ диска для этой виртуальной машины будет создан в формате qcow2;

--ram 2048 – объем оперативной памяти;

--vcpus=2 – количество процессоров;

--network network=default – виртуальная сеть default;

--hvm – полностью виртуализированная система;

--virt-type=kvm – использовать модуль ядра KVM, который задействует аппаратные возможности виртуализации процессора.

Последние две опции команды virt-install оптимизируют ВМ для использования в качестве полностью виртуализированной системы (--hvm) и указывают, что KVM является базовым гипервизором (--virt-type) для поддержки новой ВМ. Обе этих опции обеспечивают определенную оптимизацию в процессе создания и установки операционной системы; если эти опции не заданы в явном виде, то вышеуказанные значения применяются по умолчанию.

Список доступных вариантов ОС можно получить, выполнив команду:

```
$ osinfo-query os
```

Запуск Live CD в ВМ без дисков:

```
# virt-install \
 --hvm \
 --name demo \
 --ram 500 \
 --nodisks \
 --livecd \
```

```
 --graphics vnc \
 --cdrom /var/lib/libvirt/boot/altlive.iso
```

Запуск /usr/bin/httpd в контейнере (LXC), с ограничением памяти в 512 МБ и двумя ядрами хост-системы:

```
# virt-install \
 --connect lxc:/// \
 --name httpd_guest \
 --ram 512 \
 --vcpus 2 \
 --init /usr/bin/httpd
```

Создать ВМ, используя существующий том хранилища:

```
# virt-install \
 --name demo \
 --ram 512 \
 --disk /home/user/VMs/mydisk.img \
 --import
```

5.4.3 Создание виртуальных машин с помощью virt-manager

Новую ВМ можно создать, нажав кнопку «Создать виртуальную машину» в главном окне virt-manager, либо выбрав в меню «Файл»→ «Создать виртуальную машину».

На первом шаге создания ВМ необходимо выбрать метод установки ОС (Рис. 218) и нажать кнопку «Вперед».

В следующем окне для установки гостевой ОС требуется указать ISO-образ установочного диска ОС или CD/DVD-диск с дистрибутивом (Рис. 219). Данное окно будет выглядеть по-разному в зависимости от выбора, сделанного на предыдущем этапе. Здесь также можно указать версию устанавливаемой ОС.

На третьем шаге необходимо указать размер памяти и количество процессоров для ВМ (Рис. 220). Эти значения влияют на производительность хоста и ВМ.

На следующем этапе настраивается пространство хранения данных (Рис. 221).

Создание ВМ. Выбор метода установки

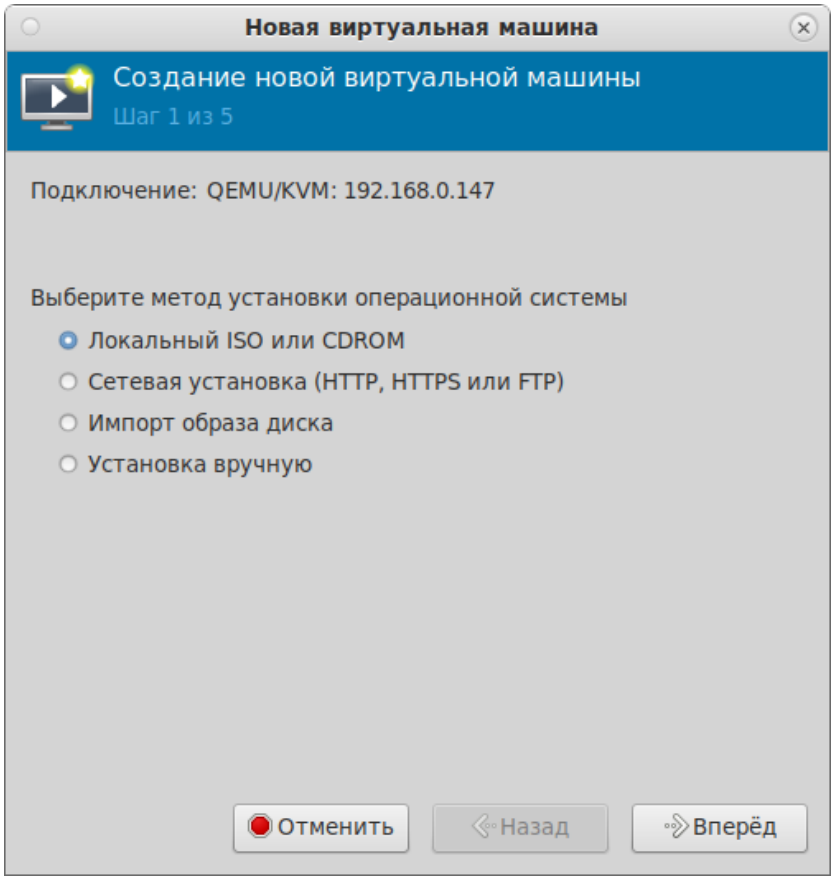


Рис. 218

Создание ВМ. Выбор ISO-образа

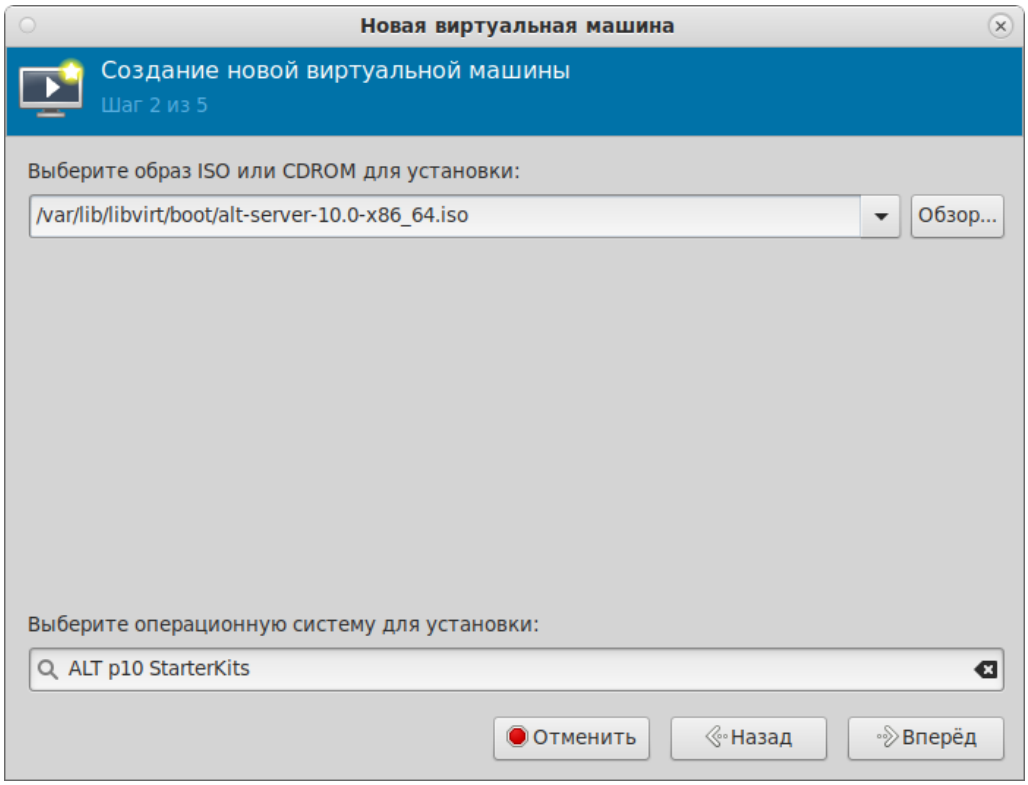


Рис. 219

Создание ВМ. Настройка ОЗУ и ЦПУ для ВМ

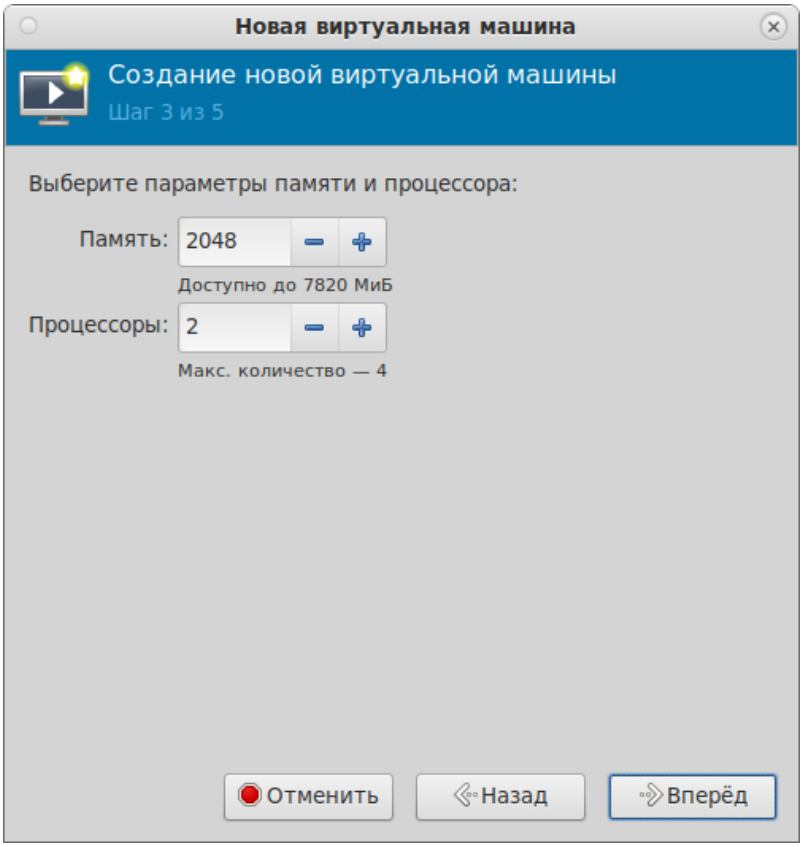


Рис. 220

Создание ВМ. Настройка пространства хранения данных

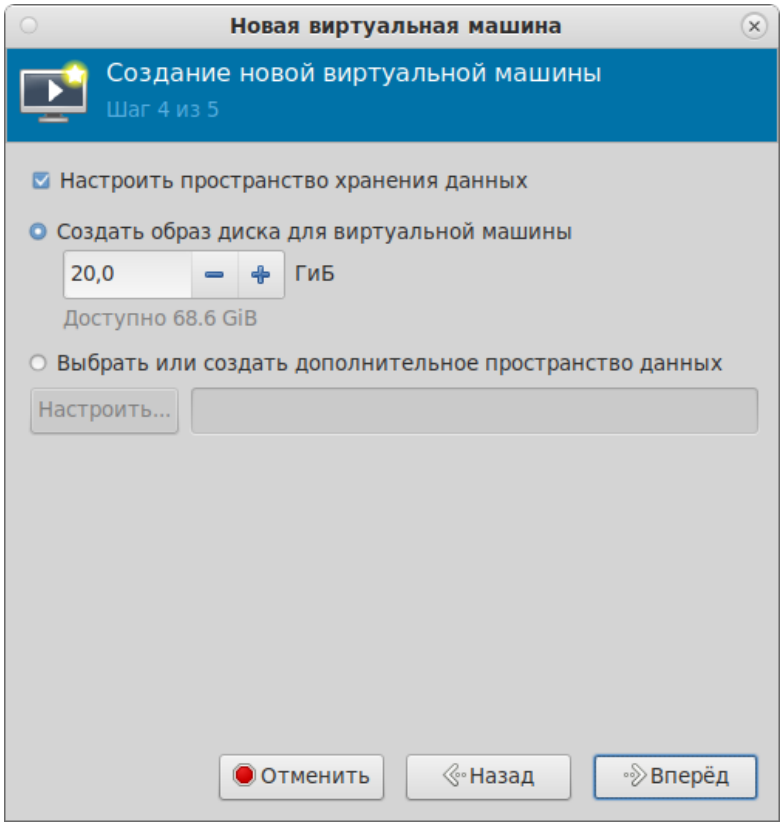


Рис. 221

На последнем этапе (Рис. 222) можно задать название ВМ, выбрать сеть и нажать кнопку «Готово».

Создание ВМ. Выбор сети

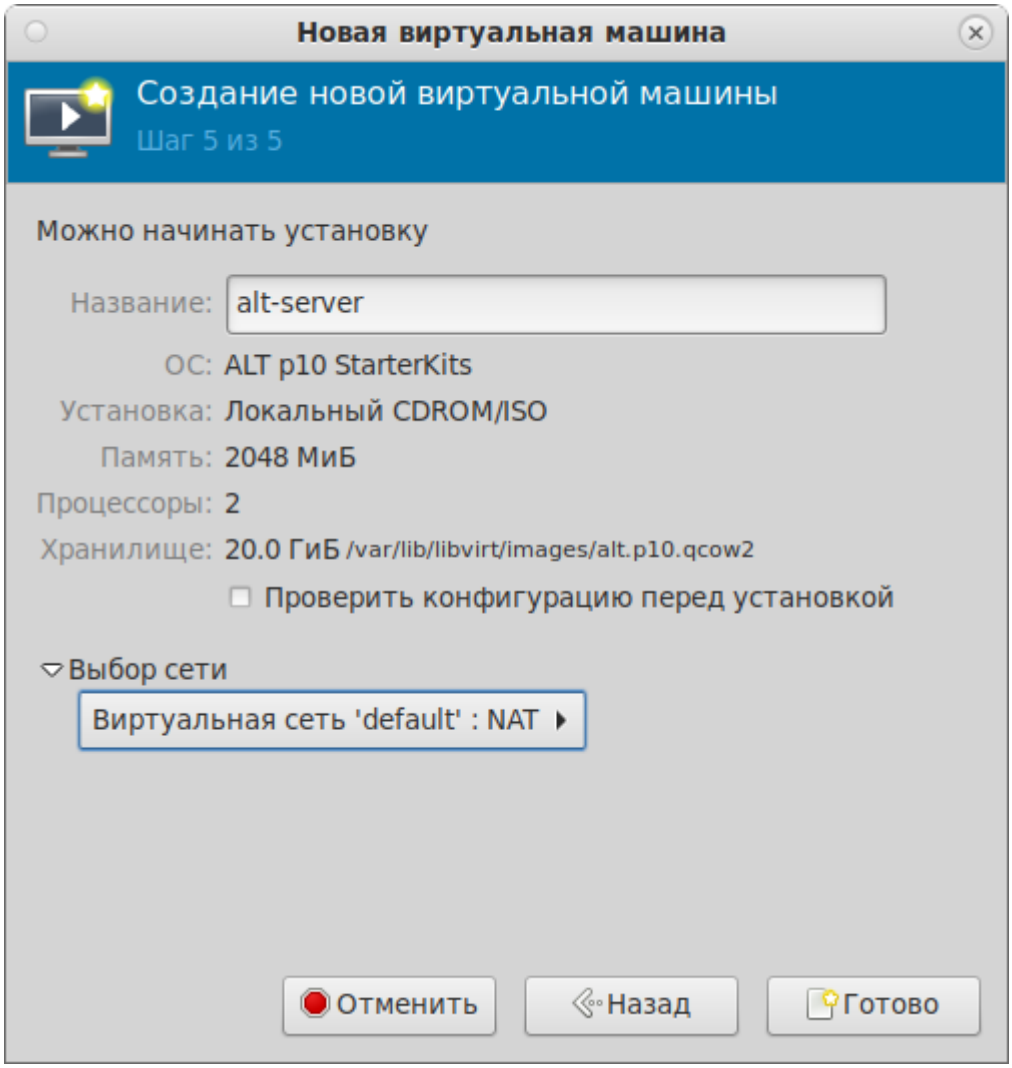


Рис. 222

В результате созданная ВМ будет запущена и после завершения исходной загрузки начнется стандартный процесс установки ОС (Рис. 223).

Окружение локального рабочего стола способно перехватывать комбинации клавиш (например, <Ctrl>+<Alt>+<F11>) для предотвращения их отправки гостевой машине. Чтобы отправить такие последовательности, используется свойство «западания» клавиш virt-manager. Для перевода клавиши в нажатое состояние необходимо нажать клавишу модификатора (<Ctrl> или <Alt>) 3 раза. Клавиша будет считаться нажатой до тех пор, пока не будет нажата любая клавиша, отличная от модификатора. Таким образом, чтобы передать гостевой системе комбинацию <Ctrl>+<Alt>+<F11>, необходимо последовательно нажать <Ctrl>+<Ctrl>+<Ctrl>+<Alt>+<F11> или воспользоваться меню «Отправить комбинацию клавиш».

Установка ОС

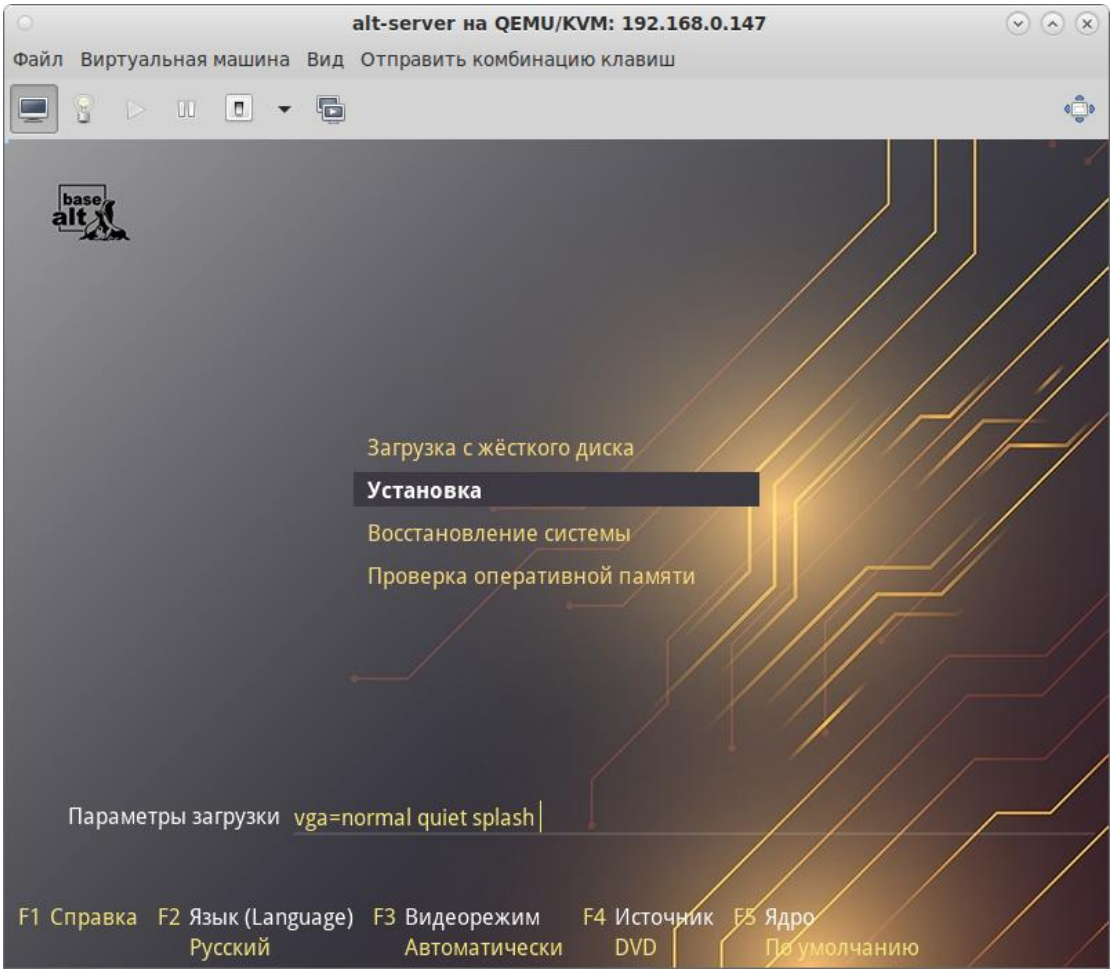


Рис. 223

5.5 Запуск и управление функционированием ВМ

5.5.1 Управление состоянием ВМ в командной строке

Команды управления состоянием ВМ:

- start – запуск ВМ;
- shutdown – завершение работы. Поведение выключаемой ВМ можно контролировать с помощью параметра on_shutdown (в файле конфигурации);
- destroy – принудительная остановка. Использование virsh destroy может повредить гостевые файловые системы. Рекомендуется использовать опцию shutdown;
- reboot – перезагрузка ВМ. Поведение перезагружаемой ВМ можно контролировать с помощью параметра on_reboot (в файле конфигурации);
- suspend – приостановить ВМ. Когда ВМ находится в приостановленном состоянии, она потребляет системную оперативную память, но не ресурсы процессора;
- resume – возобновить работу приостановленной ВМ;

– save – сохранение текущего состояния ВМ. Эта команда останавливает ВМ, сохраняет данные в файл, что может занять некоторое время (зависит от объема ОЗУ ВМ);
– restore – восстановление ВМ, ранее сохраненной с помощью команды `virsh save`. Сохраненная машина будет восстановлена из файла и перезапущена (это может занять некоторое время). Имя и идентификатор UUID ВМ останутся неизменными, но будет предоставлен новый идентификатор домена;
– undefine – удалить ВМ (конфигурационный файл тоже удаляется);
– autostart – добавить ВМ в автозагрузку;
– autostart --disable – удалить из автозагрузки.

В результате выполнения следующих команд, ВМ alt-server будет остановлена и затем удалена:

```
# virsh destroy alt-server
# virsh undefine alt-server
```

5.5.2 Управление состоянием ВМ в менеджере виртуальных машин

Для запуска ВМ в менеджере виртуальных машин virt-manager, необходимо выбрать ВМ из списка и нажать на кнопку «Включить виртуальную машину» (Рис. 224).

Включение ВМ

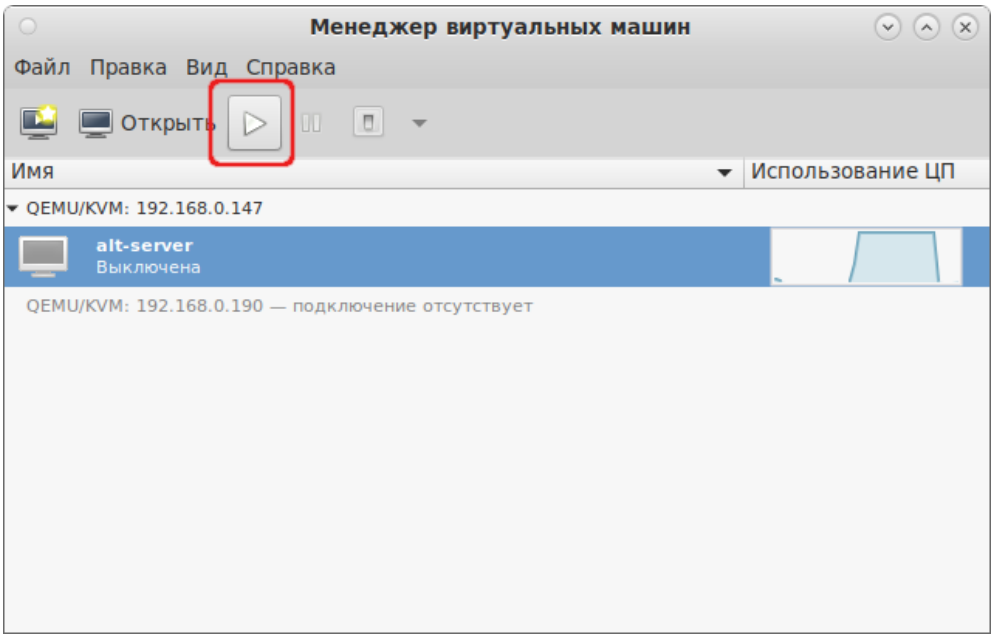


Рис. 224

Для управления запущенной ВМ используются соответствующие кнопки панели инструментов virt-manager (Рис. 225).

Кнопки управления состоянием ВМ

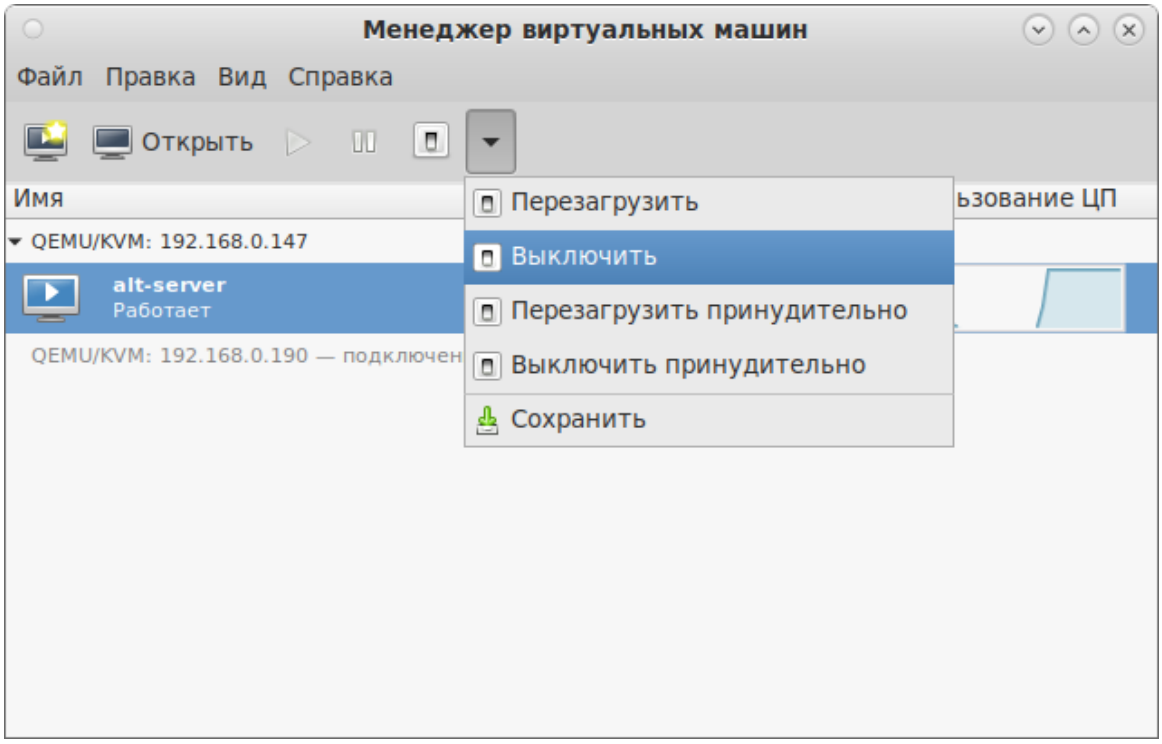


Рис. 225

Управлять состоянием ВМ можно также выбрав соответствующий пункт в контекстном меню ВМ (Рис. 226).

Контекстное меню ВМ

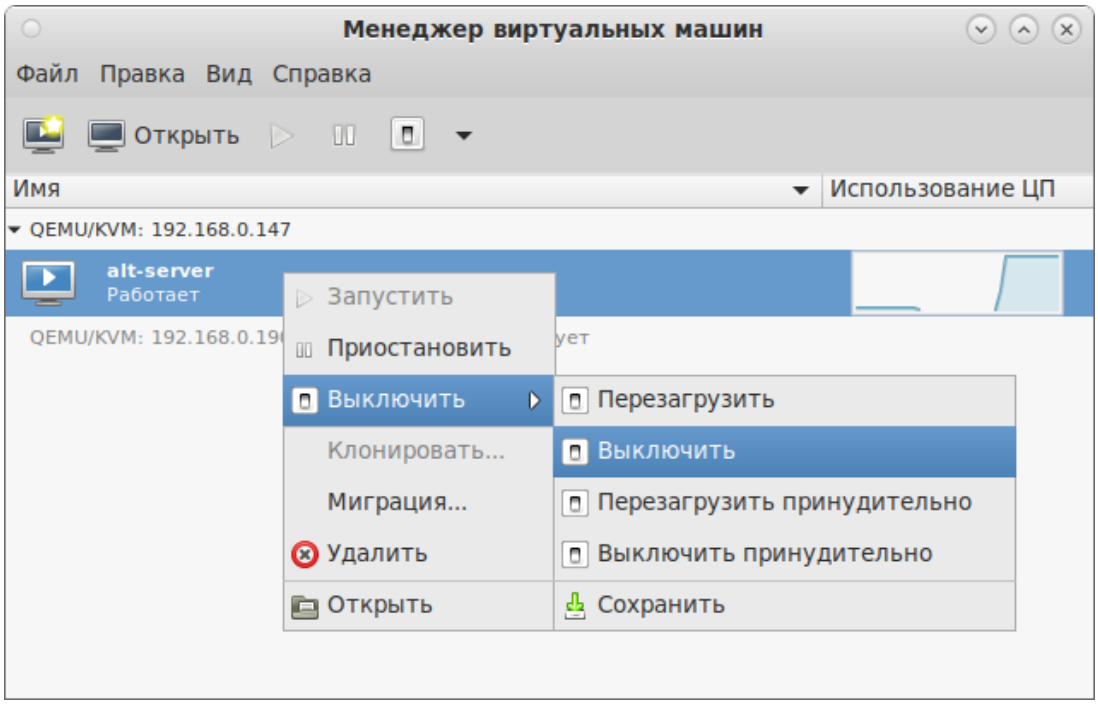


Рис. 226

5.6 Подключение к виртуальному монитору ВМ

Доступ к рабочему столу ВМ может быть организован по протоколам VNC и SPICE.

К каждой из ВМ можно подключиться, используя один IP-адрес и разные порты. Порт доступа к ВМ может быть назначен вручную или автоматически. Удаленный доступ к ВМ можно защитить паролем.

5.6.1 Использование протокола SPICE

Чтобы добавить поддержку SPICE в существующую ВМ, необходимо отредактировать её конфигурацию:

```
#  virsh edit alt-server
```

Добавить графический элемент SPICE, например:

```
<graphics type='spice' port='5900' autoport='yes' listen='127.0.0.1'>
    <listen type='address' address='127.0.0.1'/>
</graphics>
```

Добавить видеоустройство QXL:

```
<video>
    <model type='qxl'/>
</video>
```

После остановки и перезапуска ВМ она должна быть доступна через SPICE.

Проверка параметров подключения к ВМ:

```
# virsh domdisplay alt-server
spice://127.0.0.1:5900
```

В данном примере доступ к ВМ будет возможен только с локального адреса (127.0.0.1). Для удаленного подключения к ВМ SPICE-сервер должен обслуживать запросы с общедоступных сетевых интерфейсов. Для возможности подключения с других машин в конфигурации ВМ необходимо указать адрес 0.0.0.0:

```
<graphics type='spice' port='5900' autoport='yes' listen='0.0.0.0' passwd='mypasswd'>
    <listen type='address' address='0.0.0.0'/>
</graphics>
```

Пример настроек доступа к рабочему столу по протоколу SPICE в менеджере ВМ показан на Рис. 227.

Менеджер ВМ. Вкладка «Дисплей Spice»

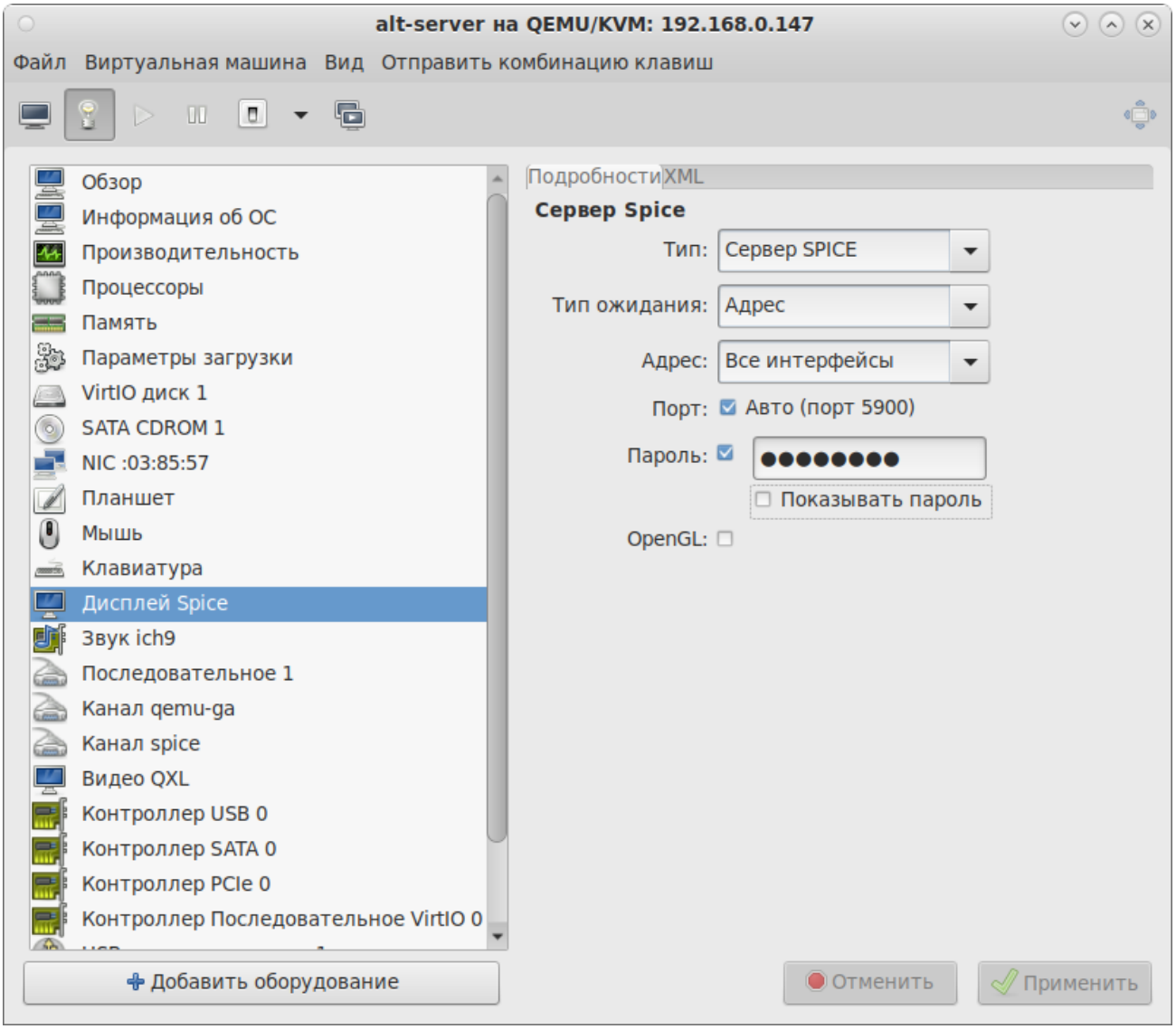


Рис. 227

Для подключения к SPICE-серверу может использоваться встроенный в virt-manager просмотрщик или любой SPICE-клиент. Примеры подключений (на хосте, с которого происходит подключение, должен быть установлен пакет virt-viewer):

```
$ virt-viewer -c qemu+ssh://user@192.168.0.147/system -d alt-server

$ remote-viewer "spice://192.168.0.147:5900"
```

Примечание. При использовании любого SPICE-клиента подключение происходит к порту и адресу хоста KVM, а не к фактическому имени/адресу ВМ.

5.6.2 Использование протокола VNC

Пример настройки доступа к рабочему столу ВМ по протоколу VNC, в файле конфигурации ВМ:

```
<graphics type='vnc' port='5900' autoport='no' listen='0.0.0.0' passwd='mypasswd'>
    <listen type='address' address='0.0.0.0'/>
```

```
</graphics>
```

Пример настроек доступа к рабочему столу по протоколу VNC в менеджере ВМ показан на Рис. 228.

Менеджер ВМ. Вкладка «Дисплей VNC»

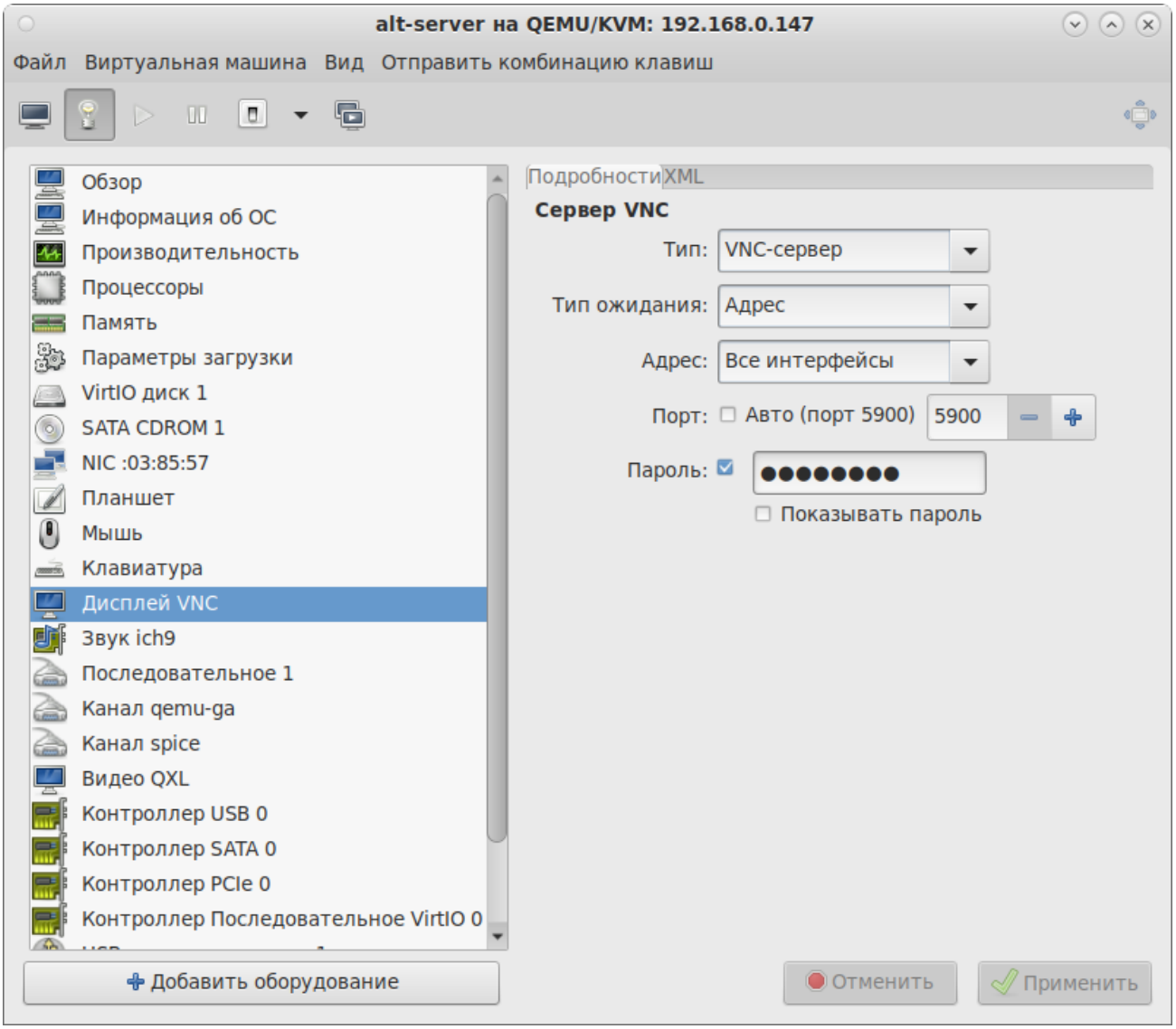


Рис. 228

Проверка параметров подключения к ВМ:

```
# virsh domdisplay alt-server
vnc://localhost:0
```

Для подключения к VNC-серверу может использоваться встроенный в virt-manager просмотрщик или любой VNC -клиент. Примеры подключений (на хосте, с которого происходит подключение, должны быть соответственно установлены пакеты virt-viewer или tigervnc):

```
$ virt-viewer -c qemu+ssh://user@192.168.0.147/system -d alt-server
$ vncviewer 192.168.0.147:5900
```

5.7 Управление ВМ

5.7.1 Управление конфигурацией ВМ

5.7.1.1 Редактирование файла конфигурации ВМ

ВМ могут редактироваться либо во время работы, либо в автономном режиме. Эту функциональность предоставляет команда `virsh edit`. Например, команда редактирования ВМ с именем alt-server:

```
# virsh edit alt-server
```

В результате выполнения этой команды откроется окно текстового редактора, заданного переменной оболочки $EDITOR.

5.7.1.2 Получение информации о ВМ

Команда для получения информации о ВМ:

```
virsh dominfo <domain>
```

где [--domain] <строка> – имя, ID или UUID домена

Пример вывода `virsh dominfo`:

```
$ virsh dominfo alt-server
ID:             3
Имя:          alt-server
UUID:           ccb6bf9e-1f8d-448e-b5f7-fa274703500b
Тип ОС:     hvm
Состояние: работает
CPU:            1
Время CPU: 702,1s
Макс.память: 1048576 KiB
Занято памяти: 1048576 KiB
Постоянство: yes
Автозапуск: выкл.
Управляемое сохранение: no
Модель безопасности: none
DOI безопасности: 0
```

Получение информации об узле:

```
$ virsh nodeinfo
Модель процессора: x86_64
CPU:                 4
Частота процессора: 1337 MHz
```

```
Сокеты:        1
Ядер на сокет: 2
Потоков на ядро: 2
Ячейки NUMA:   1
Объём памяти: 8007952 КiB
```

Просмотр списка ВМ:

```
virsh list
```

Опции команды `virsh list`:

- `--inactive` – показать список неактивных доменов;
- `--all` – показать все ВМ независимо от их состояния.

Пример вывода `virsh list`:

```
$ virsh list --all
 ID   Имя          Состояние
--------------------------
 3    alt-server   работает
```

Столбец «Статус» может содержать следующие значения:

- работает (running) – работающие ВМ, то есть те машины, которые используют ресурсы процессора в момент выполнения команды;
- blocked – заблокированные, неработающие машины. Такой статус может быть вызван ожиданием ввода/вывода или пребыванием машины в спящем режиме;
- приостановлен (paused) – приостановленные домены. В это состояние они переходят, если администратор нажал кнопку паузы в окне менеджера ВМ или выполнил команду virsh suspend. В приостановленном состоянии ВМ продолжает потреблять ресурсы, но не может занимать больше процессорных ресурсов;
- выключен (shutdown) – ВМ, завершающие свою работу. При получении ВМ сигнала завершения работы, она начнет завершать все процессы (некоторые операционные системы не отвечают на такие сигналы);
- dying – сбойные домены и домены, которые не смогли корректно завершить свою работу;
- crashed – сбойные домены, работа которых была прервана. В этом состоянии домены находятся, если не была настроена их перезагрузка в случае сбоя.

Команда получения информации о виртуальных процессорах:

```
virsh vcpuinfo <domain>
```

Пример вывода:

```
# virsh vcpuinfo alt-server
```

```
Виртуальный процессор:: 0
CPU:            1
Состояние: работает
Время CPU: 572,8s
Соответствие ЦП: yyyy
```

Команда сопоставления виртуальных процессоров физическим:

```
virsh vcpupin <domain> [--vcpu <число>] [--cpulist <строка>] [--config] [--live] [--current]
```

Здесь:

[--domain] <строка> – имя, ID или UUID домена;

--vcpu <число> – номер виртуального процессора;

--cpulist <строка> – номера физических процессоров. Если номера не указаны, команда вернет текущий список процессоров;

--config – с сохранением после перезагрузки;

--live – применить к работающему домену;

--current – применить к текущему домену.

Пример вывода:

```
 # virsh vcpupin alt-server
 Виртуальный процессор:   Соответствие ЦП
----------------------------------------
 0                        0-3
```

Команда изменения числа процессоров для домена (заданное число не может превышать значение, определенное при создании ВМ):

```
virsh setvcpus <domain> <count> [--maximum] [--config] [--live] [--current] [--guest] [--hotpluggable]
```

где

[--domain] <строка> – имя, ID или UUID домена;

[--count] <число> – число виртуальных процессоров;

--maximum – установить максимальное ограничение на количество виртуальных процессоров, которые могут быть подключены после следующей перезагрузки домена;

--config – с сохранением после перезагрузки;

--live – применить к работающему домену;

--current – применить к текущему домену;

--guest – состояние процессоров ограничивается гостевым доменом.

Команда изменения выделенного ВМ объема памяти:

```
virsh setmem <domain> <size> [--config] [--live] [--current]
```

где

[--domain] <строка> – имя, ID или UUID домена;

[--size] <число> – целое значение нового размера памяти (по умолчанию в КБ);

--config – с сохранением после перезагрузки;

--live – применить к работающему домену;

--current – применить к текущему домену.

Объем памяти, определяемый заданным числом, должен быть указан в килобайтах. Объем не может превышать значение, определенное при создании ВМ, но в то же время не должен быть меньше 64 мегабайт. Изменение максимального объема памяти может оказать влияние на функциональность ВМ только в том случае, если указанный размер меньше исходного. В таком случае использование памяти будет ограничено.

Команда изменения максимальное ограничение памяти:

```
virsh setmaxmem <domain> <size> [--config] [--live] [--current]
```

где

[--domain] <строка> – имя, ID или UUID домена;

[--size] <число> – целое значение максимально допустимого размера памяти (по умолчанию в КБ);

--config – с сохранением после перезагрузки;

--live – применить к работающему домену;

--current – применить к текущему домену.

Примеры изменения размера оперативной памяти и количества виртуальных процессоров соответственно:

```
# virsh setmaxmem --size 624000 alt-server
# virsh setmem --size 52240 alt-server
# virsh setvcpus --config alt-server 3 --maximum
```

Команда для получения информации о блочных устройствах работающей ВМ:

```
virsh domblkstat <domain> [--device <строка>] [--human]
```

где:

[--domain] <строка> – имя, ID или UUID домена;

--device <строка> – блочное устройство;

--human – форматировать вывод.

Команда для получения информации о сетевых интерфейсах работающей ВМ:

```
virsh domifstat <domain> <interface>
```

где:

[--domain] <строка> – имя, ID или UUID домена;

[--interface] <строка> – устройство интерфейса, указанное по имени или MAC-адресу.

5.7.1.3 Конфигурирование ВМ в менеджере виртуальных машин

С помощью менеджера виртуальных машин можно получить доступ к подробной информации о всех ВМ, для этого следует:

1) в главном окне менеджера выбрать ВМ;
2) нажать кнопку «Открыть» (Рис. 229);
3) в открывшемся окне нажать кнопку «Показать виртуальное оборудование» (Рис. 230);
4) появится окно просмотра сведений ВМ.

Окно менеджера виртуальных машин

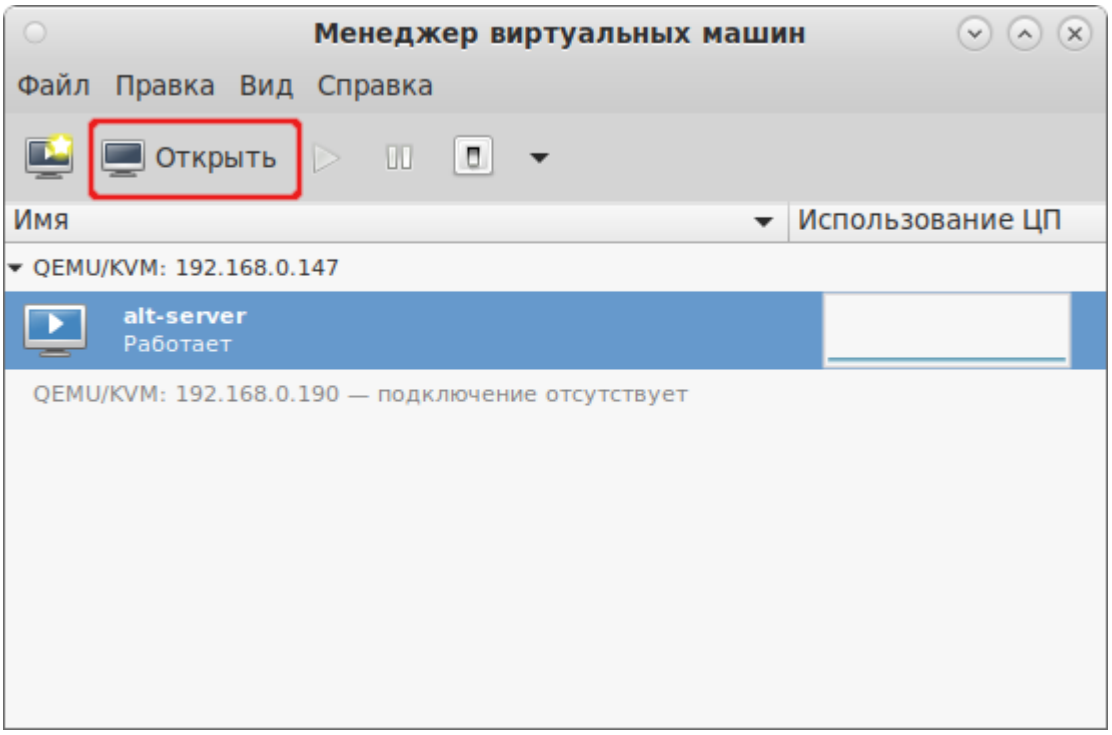


Рис. 229

Для изменения требуемого параметра необходимо перейти на нужную вкладку, внести изменения и подтвердить операцию, нажав кнопку «Применить» (Рис. 231 – Рис. 232).

Окно параметров ВМ

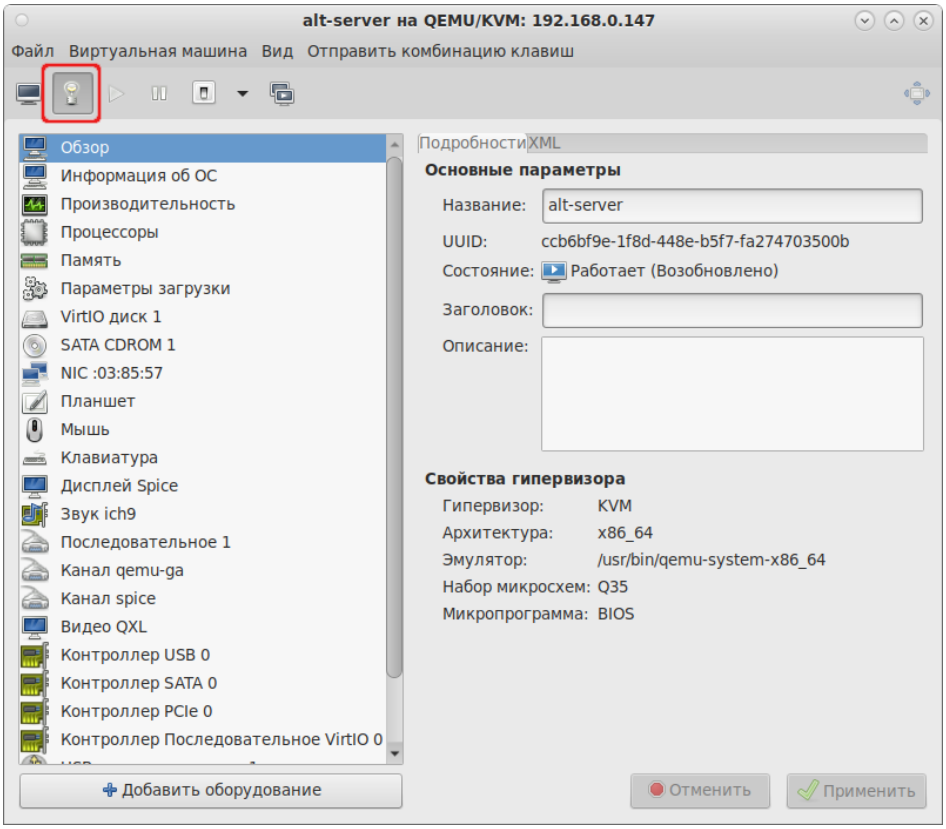


Рис. 230

Вкладка «Память»

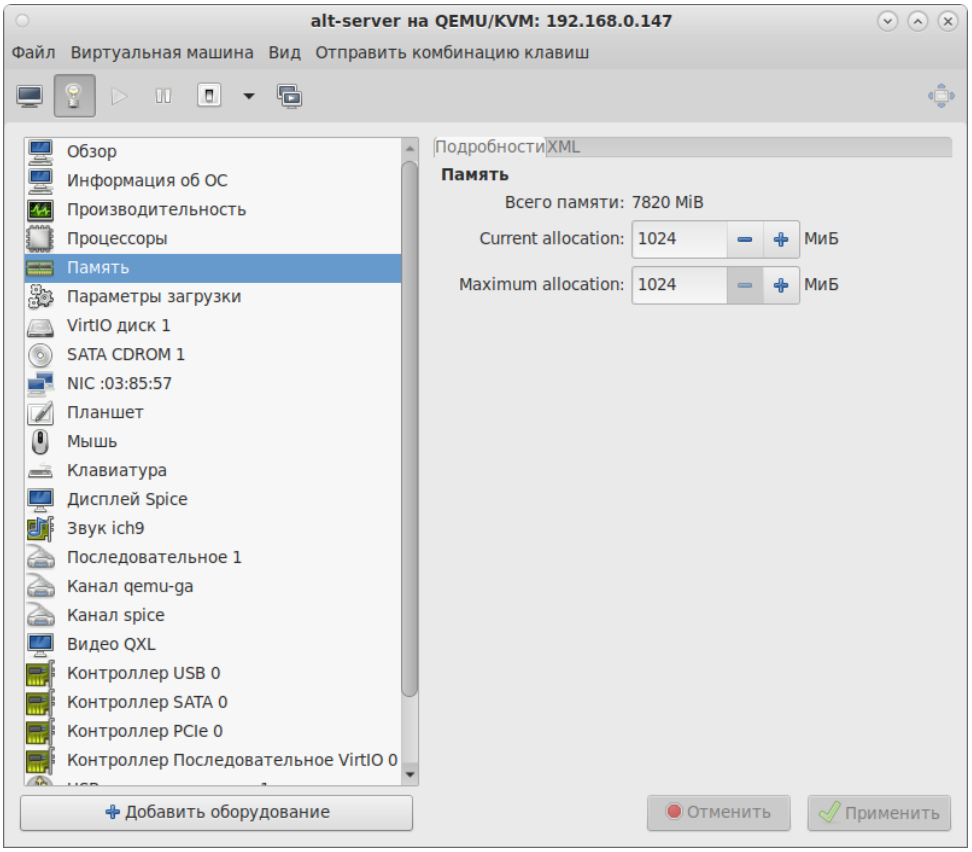


Рис. 231

Вкладка «Процессоры»

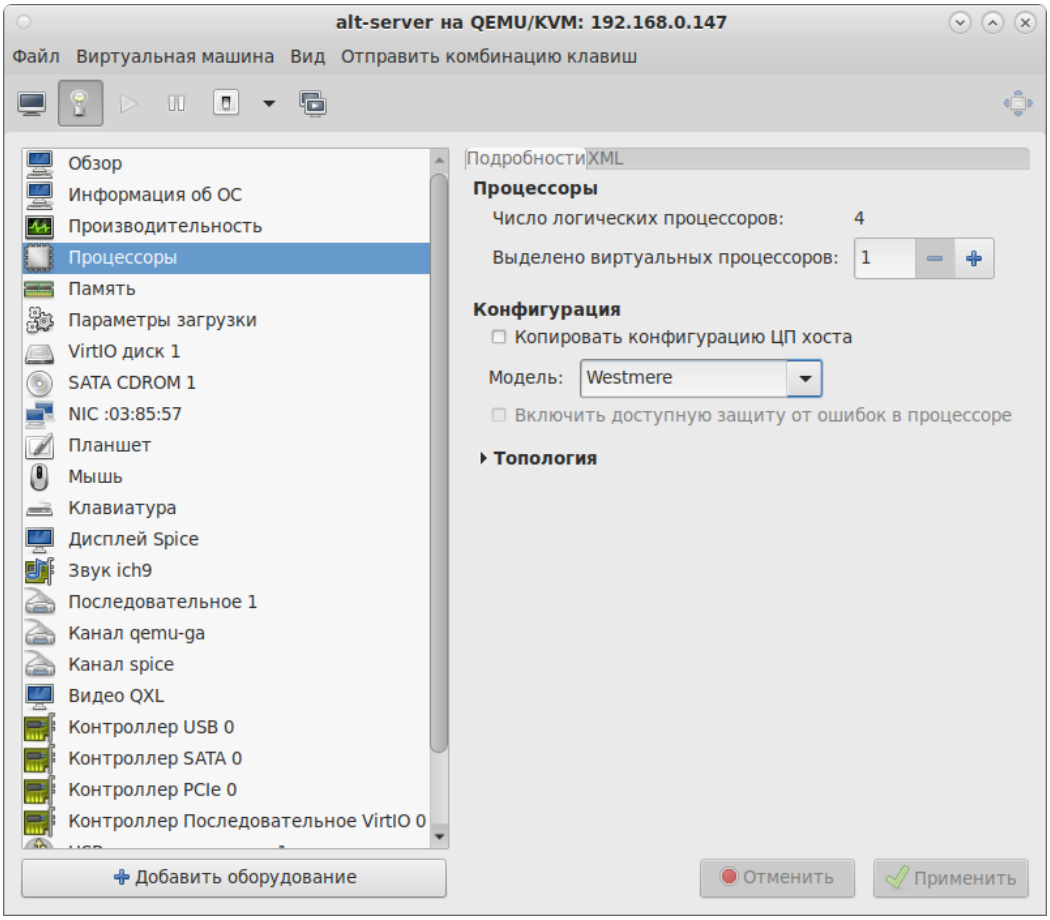


Рис. 232

5.7.1.4 Мониторинг состояния

С помощью менеджера виртуальных машин можно изменить настройки контроля состояния ВМ.

Для этого в меню «Правка» следует выбрать пункт «Параметры», в открывшемся окне «Настройки» на вкладке «Статистика» можно задать время обновления состояния ВМ в секундах (Рис. 233).

Вкладка «Статистика»

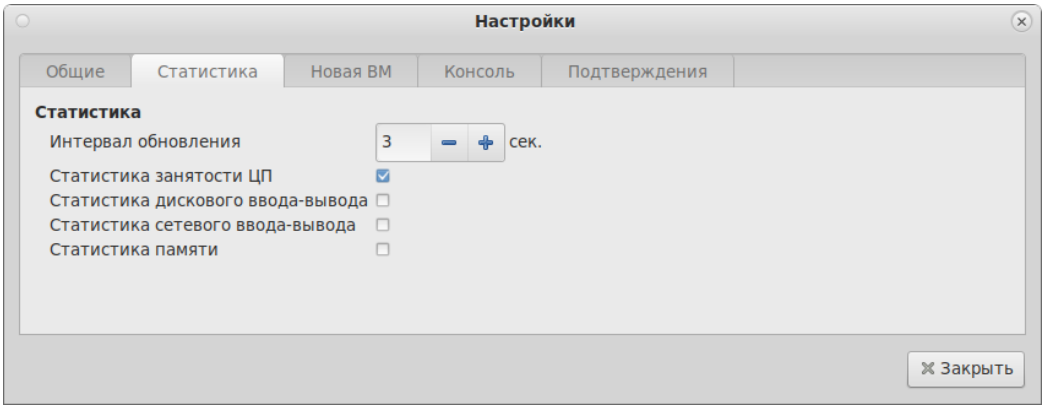


Рис. 233

Во вкладке «Консоль» (Рис. 234) можно выбрать, как открывать консоль, и указать устройство ввода.

Вкладка «Консоль»

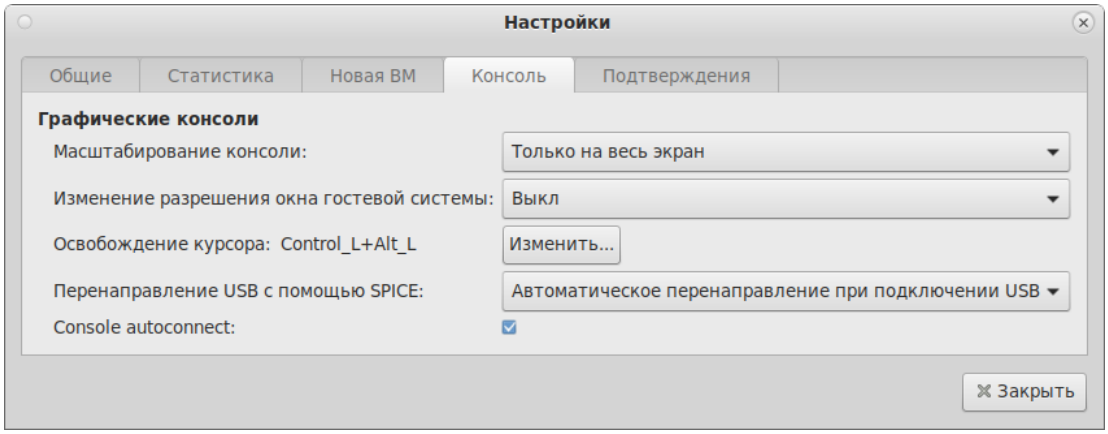


Рис. 234

5.7.2 Управление виртуальными сетевыми интерфейсами и сетями

При базовых настройках используется виртуальная сеть недоступная извне.

Доступ по IP может быть осуществлен с компьютера, на котором поднят KVM. Изнутри доступ происходит через NAT.

Возможные варианты настройки сети:

- NAT – это вариант по умолчанию. Внутренняя сеть, предоставляющая доступ к внешней сети с автоматическим применением NAT;
- Маршрутизация (Routed) – аналогично режиму NAT внутренняя сеть, предоставляющая доступ к внешней сети, но без NAT. Предполагает дополнительные настройки таблиц маршрутизации во внешней сети;
- Изолированная IPv4/IPv6 сеть (Isolated) – в этом режиме ВМ, подключенные к виртуальному коммутатору, могут общаться между собой и с хостом. При этом их трафик не будет выходить за пределы хоста;
- Bridge – подключение типа мост. Позволяет реализовать множество различных конфигураций, в том числе и назначение IP из реальной сети;
- SR-IOV pool (Single-root IOV) – перенаправление одной PCI сетевых карт хост-машины на ВМ. Технология SR-IOV повышает производительность сетевой виртуализации, избавляя гипервизор от обязанности организовывать совместное использование физического адаптера и перекладывая задачу реализации мультиплексирования на сам адаптер. В этом случае обеспечивается прямая пересылка ввода/вывода с ВМ непосредственно на адаптер.

5.7.2.1 Управление виртуальными сетями в командной строке

Команды управления виртуальными сетями:

- virsh net-autostart имя_сети – автоматический запуск заданной сети;
- virsh net-create файл_XML – создание и запуск новой сети на основе существующего XML-файла;
- virsh net-define файл_XML – создание нового сетевого устройства на основе существующего XML-файла (устройство не будет запущено);
- virsh net-list – просмотр списка виртуальных сетей;
- virsh net-dumpxml имя_сети – просмотр информации о заданной виртуальной сети;
- virsh net-destroy имя_сети – удаление заданной сети;
- virsh net-name UUID_сети – преобразование заданного идентификатора в имя сети;
- virsh net-uuid имя_сети – преобразование заданного имени в идентификатор UUID;
- virsh net-update имя_сети – обновить существующую конфигурацию сети;
- virsh net-start имя_неактивной_сети – запуск неактивной сети;
- virsh net-undefine имя_неактивной_сети — удаление определения неактивной сети.

```
# virsh net-list --all
Имя       Состояние    Автозапуск   Постоянный
----------------------------------------------
 default   не активен   no           yes

# virsh net-start default
Сеть default запущен

# virsh net-autostart default
Добавлена метка автоматического запуска сети default

# virsh net-list
 Имя       Состояние   Автозапуск   Постоянный
----------------------------------------------
 default   активен     yes          yes

# virsh net-dumpxml default
<network>
  <name>default</name>
  <uuid>0b37eff3-2234-4929-8a42-04a9cf35d3aa</uuid>
```

```
  <forward mode='nat'/>
  <bridge name='virbr0' stp='on' delay='0'/>
  <mac address='52:54:00:d2:30:b6'/>
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

Иногда бывает полезно выдавать клиенту один и тот же IP-адрес независимо от момента обращения. Пример добавления статического сопоставления MAC- и IP-адреса ВМ:

1) получить MAC-адрес ВМ (alt-server – имя ВМ):

```
# virsh dumpxml alt-server | grep 'mac address'
   <mac address='52:54:00:ba:f2:76'/>
```

2) отредактировать XML-конфигурацию сети (default – имя сети):

```
# virsh net-edit default
```

после строки:

```
<range start='192.168.122.2' end='192.168.122.254'/>
```

вставить строки с MAC-адресами виртуальных адаптеров:

```
<host mac='52:54:00:ba:f2:76' name='alt-server' ip='192.168.122.50'/>
```

3) сохранить изменения и перезапустить виртуальную сеть:

```
# virsh net-destroy default
# virsh net-start default
```

Изменения внесённые с помощью команды `virsh net-edit` не вступят в силу в силу до тех пор, пока сеть не будет перезапущена, что приведет к потере всеми ВМ сетевого подключения к хосту до тех пор, пока их сетевые интерфейсы повторно не подключаться.

Изменения в конфигурацию сети, можно внести с помощью команды `virsh net-update`, которая требует немедленного применения изменений. Например, чтобы добавить запись статического хоста, можно использовать команду:

```
# virsh net-update default add ip-dhcp-host \
   "<host           mac='52:54:00:ba:f2:76'           name='alt-server'
ip='192.168.122.50' />" \
   --live --config
```

5.7.2.2 Управление виртуальными сетями в менеджере виртуальных машин

В менеджере виртуальных машин virt-manager существует возможность настройки виртуальных сетей для обеспечения сетевого взаимодействия ВМ как между собой, так и с хостовой ОС.

Для настройки виртуальной сети с помощью virt-manager необходимо:

1) в меню «Правка» выбрать «Свойства подключения» (Рис. 235);
2) в открывшемся окне перейти на вкладку «Виртуальные сети» (Рис. 236);
3) доступные виртуальные сети будут перечислены в левой части окна. Для доступа к настройкам сети необходимо выбрать сеть.

Меню «Правка»

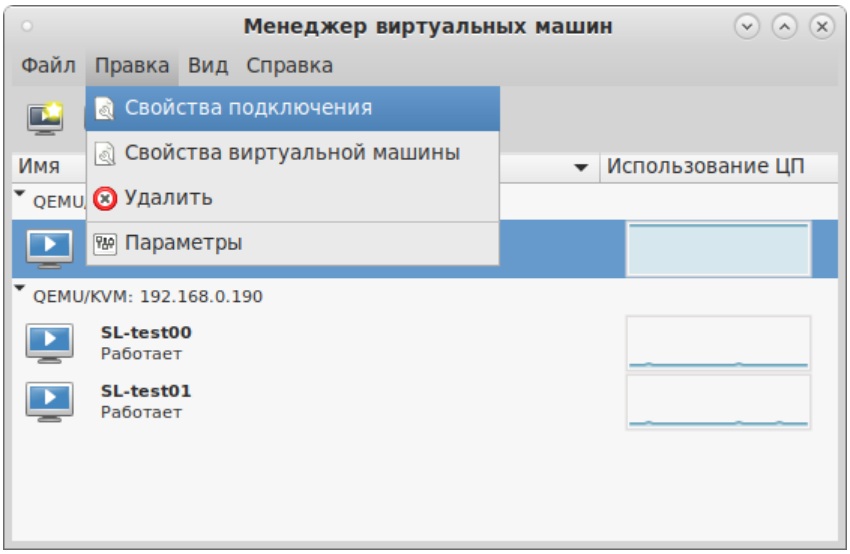


Рис. 235

Окно параметров виртуальной сети

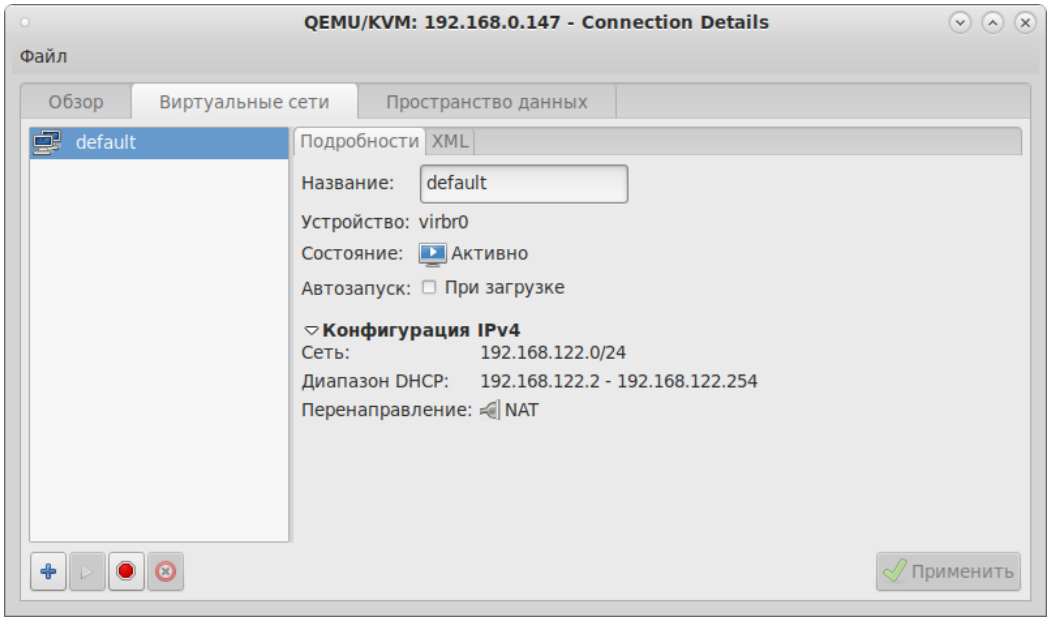


Рис. 236

Для добавления новой виртуальной сети следует нажать кнопку «Добавить сеть» («+»), расположенную в нижнем левом углу диалогового окна «Свойства соединения» (Рис. 236). В открывшемся окне (Рис. 237) следует ввести имя для новой сети и задать необходимые настройки: выбрать способ подключения виртуальной сети к физической, ввести пространство адресов IPv4 для виртуальной сети, указать диапазон DHCP, задав начальный и конечный адрес и нажать кнопку «Готово».

Создание новой виртуальной сети

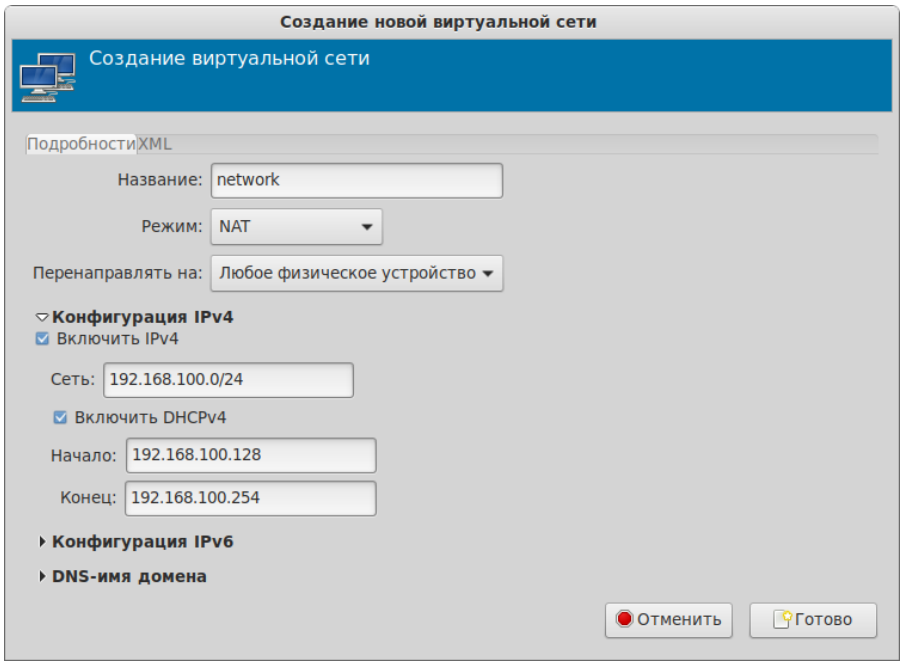


Рис. 237

5.7.3 Управление хранилищами

API-интерфейс libvirt обеспечивает удобную абстракцию для размещения образов ВМ и файловых систем, которая носит название storage pools (пул хранилищ). Пул хранилищ – это локальный каталог, локальное устройство хранения данных (физический диск, логический том или хранилище на основе хост-адаптера шины SCSI [SCSI HBA]), файловая система NFS (network file system), либо сетевое хранилище блочного уровня, управляемое посредством libvirt и позволяющее создать и хранить один или более образов виртуальных машин.

По умолчанию команды на базе libvirt используют в качестве исходного пула хранилищ для каталога файловой системы каталог `/var/lib/libvirt/images` на хосте виртуализации.

Образ диска – это снимок данных диска виртуальной машины, сохраненный в том или ином формате. libvirt понимает несколько форматов образов. Так же возможна работа с образами CD/DVD дисков. Каждый образ хранится в том или ином хранилище.

Типы хранилищ, с которыми работает libvirt:

dir – каталог в файловой системе;

disk – физический диск;

fs – отформатированное блочное устройство;

gluster – файловая система Gluster;

isci – хранилище iSCSI;

logical – группа томов LVM;

mpath – регистратор многопутевых устройств;

netfs – экспорт каталога из сети;

rbd – блочное устройство RADOS/Ceph;

scsi – хост-адаптер SCSI;

sheepdog – файловая система Sheepdog;

zfs – пул ZFS.

5.7.3.1 Управление хранилищами в командной строке

Команды управления хранилищами:

- pool-define – определить неактивный постоянный пул носителей на основе файла XML;
- pool-create – оздать пул из файла XML;
- pool-define-as – определить пул на основе набора аргументов;
- pool-create-as – создать пул на основе набора аргументов;
- pool-dumpxml – вывести файл конфигурации XML для заданного пула;
- pool-list – вывести список пулов;
- pool-build – собрать пул;
- pool-start – запустить ранее определённый неактивный пул;
- pool-autostart – автозапуск пула;
- pool-destroy – разрушить (остановить) пул;
- pool-delete – удалить пул;
- pool-edit – редактировать XML-конфигурацию пула носителей;
- pool-info – просмотр информации о пуле носителей;
- pool-refresh – обновить пул;
- pool-undefine – удалить определение неактивного пула.

Команда `virsh pool-define-as` создаст файл конфигурации для постоянного пула хранения. Позже этот пул можно запустить командой `virsh pool-start`, настроить его на автоматический запуск при загрузке хоста, остановить командой `virsh pool-destroy`.

Команда `virsh pool-create-as` создаст временный пул хранения (файл конфигурации не будет создан), который будет сразу запущен. Этот пул хранения будет удалён командой

`virsh pool-destory`. Временный пул хранения нельзя запустить автоматически при загрузке. Преобразовать существующий временный пул в постоянный, можно создав файл XML-описания:

```
virsh pool-dumpxml имя_пула > имя_пула.xml && virsh pool-define имя_пула.xml
```

Пример создания пула хранения на основе NFS (netfs):

```
# virsh pool-create-as NFS-POOL netfs \
--source-host 192.168.0.105 \
--source-path /export/storage \
--target /var/lib/libvirt/images/NFS-POOL
Пул NFS-POOL создан
```

Первый аргумент (NFS-POOL) идентифицирует имя нового пула, второй аргумент идентифицирует тип создаваемого пула. Аргумент опции `--source-host` идентифицирует хост, который экспортирует каталог пула хранилищ посредством NFS. Аргумент опции `--source-path` определяет имя экспортируемого каталога на этом хосте. Аргумент опции `--target` идентифицирует локальную точку монтирования, которая будет использоваться для обращения к пулу хранилищ (этот каталог должен существовать).

Примечание. Для возможности монтирования NFS хранилища должен быть запущен nfs-client:

```
# systemctl enable --now nfs-client.target
```

После создания нового пула хранилищ он будет указан в выходной информации команды virsh pool-list:

```
virsh pool-list --all --details
 Имя        Состояние   Автозапуск   Постоянный   Размер      Распределение   Доступно
-------------------------------------------------------------------------------------
 default    работает    yes          yes          69,12 GiB   485,35 MiB      68,65 GiB
 NFS-POOL   работает    no           no           29,40 GiB   7,26 GiB        22,14 GiB
```

В выводе команды видно, что опция «Автозапуск» («Autostart») для пула NFS-POOL имеет значение no (нет), т. е. после перезапуска системы этот пул не будет автоматически доступен для использования, и что опция «Постоянный» («Persistent») также имеет значение «no», т. е. после перезапуска системы этот пул вообще не будет определен. Пул хранилищ является постоянным только в том случае, если он сопровождается XML-описанием пула хранилищ, которое находится в каталоге `/etc/libvirt/storage`. XML-файл описания пула хранилищ имеет такое же имя, как у пула хранилищ, с которым он ассоциирован.

Чтобы создать файл XML-описания для сформированного в ручном режиме пула, следует воспользоваться командой `virsh pool-dumpxml`, указав в качестве ее заключительного аргумента имя пула, для которого нужно получить XML-описание. Эта команда осуществляет

запись в стандартное устройство вывода, поэтому необходимо перенаправить ее выходную информацию в соответствующий файл.

Например, следующая команда создаст файл XML-описания для созданного ранее пула NFS-POOL и определит постоянный пул на основе этого файла:

```
# virsh pool-dumpxml NFS-POOL > NFS-POOL.xml && virsh pool-define NFS-POOL.xml
Пул NFS-POOL определён на основе NFS-POOL.xml
```

Чтобы задать для пула хранилищ опцию «Автозапуск» («Autostart»), можно воспользоваться командой `virsh pool-autostart`:

```
# virsh pool-autostart NFS-POOL
Добавлена метка автоматического запуска пула NFS-POOL
```

Маркировка пула хранилищ как автозапускаемого говорит о том, что этот пул хранилищ будет доступен после любого перезапуска хоста виртуализации (каталог `/etc/libvirt/storage/autostart` будет содержать символьную ссылку на XML-описание этого пула хранилищ).

Пример создания постоянного локального пула:

```
# virsh pool-define-as boot --type dir --target /var/lib/libvirt/boot
Пул boot определён

# virsh pool-list --all
 Имя         Состояние    Автозапуск
-----------------------------------
 boot        не активен   no
 default     активен      yes
 NFS-POOL    активен      yes

# virsh pool-build boot
Пул boot собран

# virsh pool-start boot
Пул boot запущен

# virsh pool-autostart boot
Добавлена метка автоматического запуска пула newpool

# virsh pool-list --all
 Имя         Состояние    Автозапуск
-----------------------------------
 boot        активен     yes
 default     активен     yes
 NFS-POOL    активен     yes
```

5.7.3.2 Настройка хранилищ в менеджере виртуальных машин

Для настройки хранилищ с помощью virt-manager необходимо:

1) в меню «Правка» выбрать «Свойства подключения» (Рис. 235);
4) в открывшемся окне перейти на вкладку «Пространство данных» (Рис. 238).

Вкладка «Пространство данных»

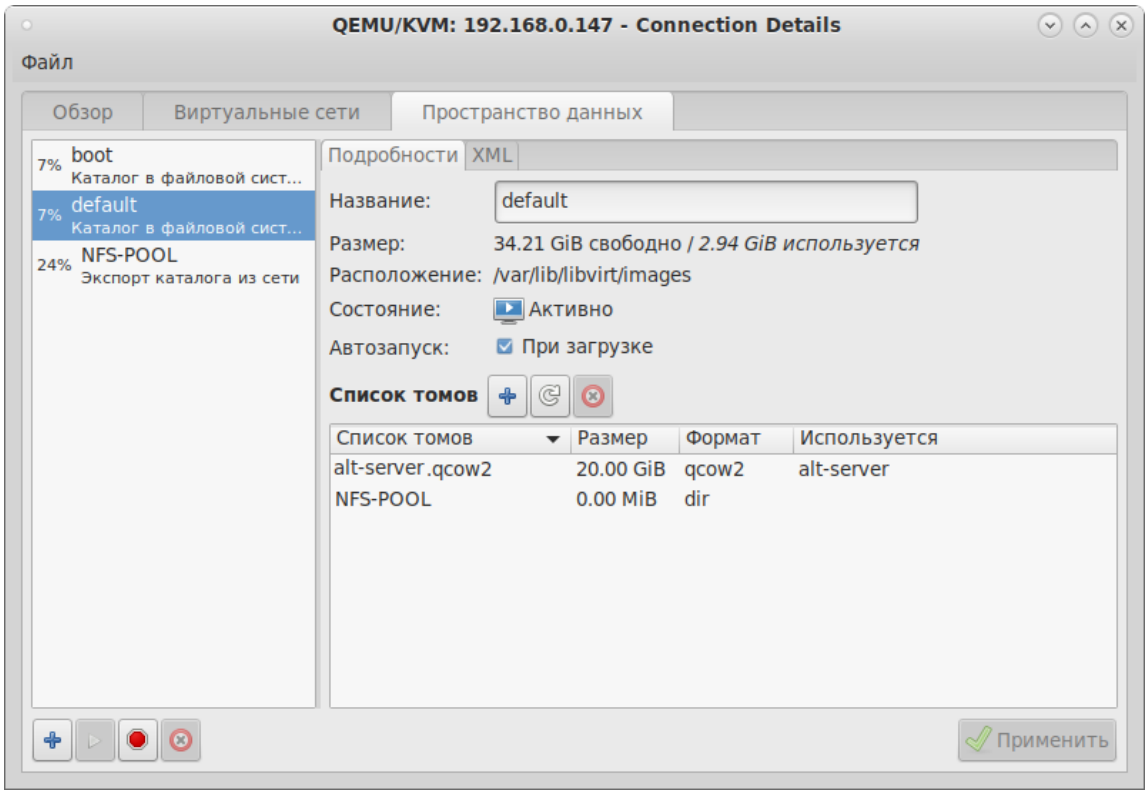


Рис. 238

Для добавления пула следует нажать кнопку «Добавить пул», расположенную в нижнем левом углу диалогового окна «Свойства соединения» (Рис. 238). В открывшемся окне (Рис. 239) следует выбрать тип пула, далее необходимо задать параметры пула (Рис. 240).

Создание пула хранения. Выбор типа пула

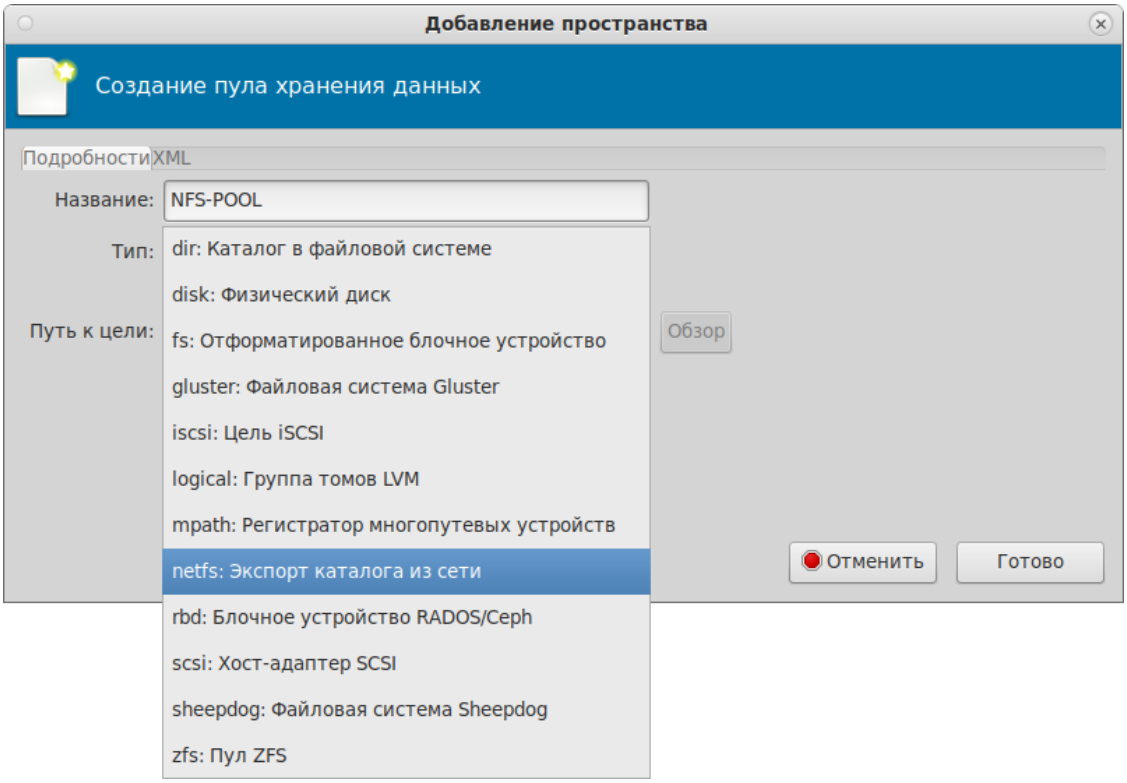


Рис. 239

Создание пула хранения. Ввод параметров

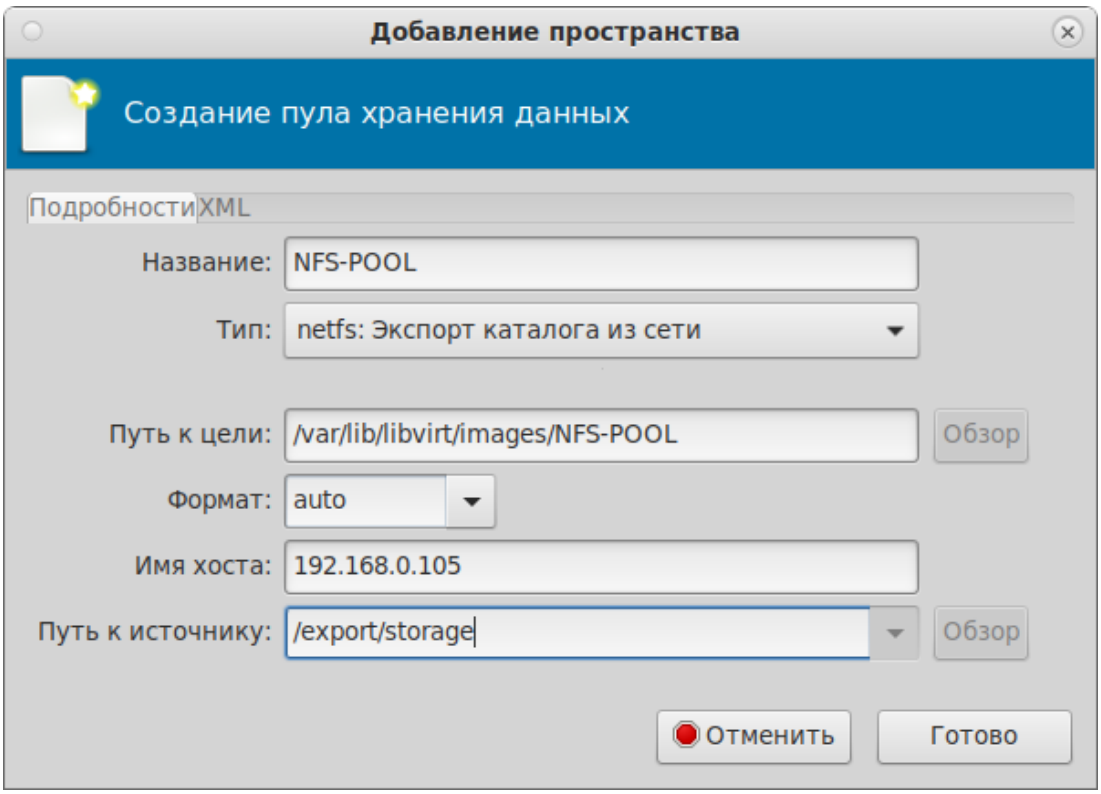


Рис. 240

5.8 Миграция ВМ

Под миграцией понимается процесс переноса ВМ с одного узла на другой.

Живая миграция позволяет перенести работу ВМ с одного физического хоста на другой без остановки ее работы.

Для возможности миграции ВМ, ВМ должна быть создана с использованием общего пула хранилищ (NFS, ISCSI, GlusterFS, CEPH).

Примечание. Живая миграция возможна даже без общего хранилища данных (с опцией `--copy-storage-all`). Но это приведет к большому трафику при копировании образа ВМ между серверами виртуализации и к заметному простою сервиса. Что бы миграция была по-настоящему «живой» с незаметным простоем необходимо использовать общее хранилище.

5.8.1 Миграция с помощью virsh

ВМ можно перенести на другой узел с помощью команды virsh. Для выполнения живой миграции нужно указать параметр `--live`. Команда переноса:

```
# virsh migrate --live VMName DestinationURL
```

где VMName – имя перемещаемой ВМ;

DestinationURL – URL или имя хоста узла назначения. Узел назначения должен использовать тот же гипервизор и служба libvirt на нем должна быть запущена.

После ввода команды будет запрошен пароль администратора узла назначения.

Для выполнения живой миграции ВМ alt-server на узел 192.168.0.190 с помощью virsh, необходимо выполнить следующие действия:

1) убедиться, что ВМ запущена:

```
# virsh list
ID   Имя          Состояние
------------------------------
 7    alt-server   работает
```

2) выполнить следующую команду, чтобы начать перенос ВМ на узел 192.168.0.190 (после ввода команды будет запрошен пароль пользователя root системы назначения):

```
# virsh migrate --live alt-server qemu+ssh://192.168.0.190/system
```

3) процесс миграции может занять некоторое время в зависимости от нагрузки и размера ВМ. virsh будет сообщать только об ошибках. ВМ будет продолжать работу на исходном узле до завершения переноса;

4) проверить результат переноса, выполнив на узле назначения команду:

```
# virsh list
```

5.8.2 Миграция с помощью virt-manager

Менеджер виртуальных машин virt-manager поддерживает возможность миграции ВМ между серверами виртуализации.

Для выполнения миграции, в virt-manager необходимо выполнить следующие действия:

1) подключить второй сервер виртуализации («Файл»→ «Добавить соединение…»);
2) в контекстном меню ВМ (она должна быть запущена) (Рис. 241) выбрать пункт «Миграция»;
3) в открывшемся окне (Рис. 242) выбрать конечный узел и нажать кнопку «Миграция».

Пункт «Миграция» в контекстном меню ВМ

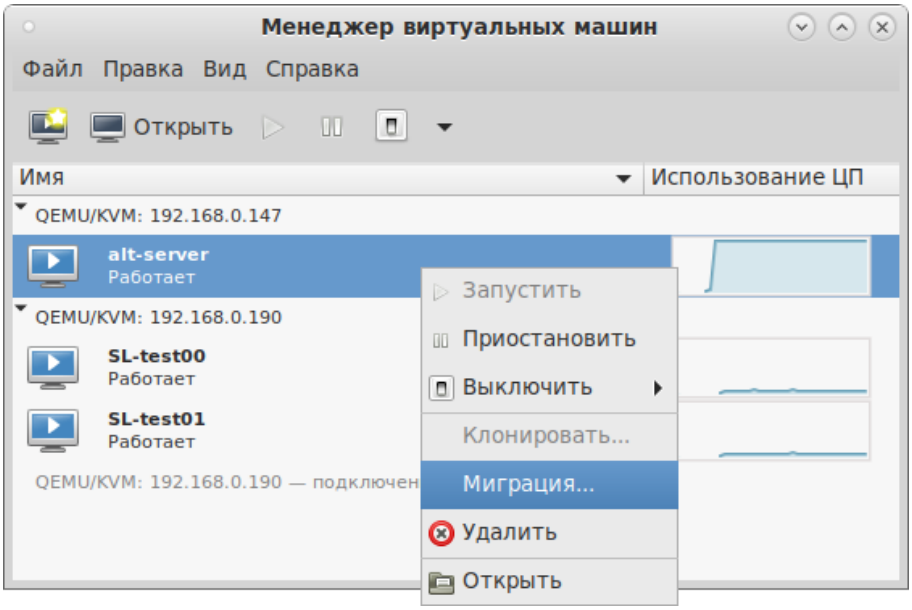


Рис. 241

Миграция ВМ

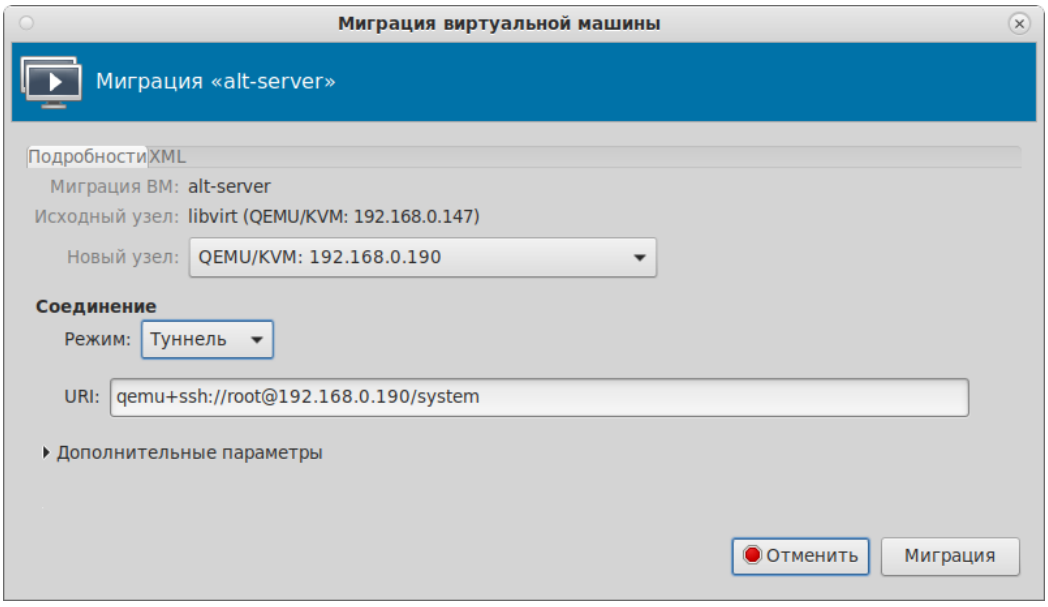


Рис. 242

При этом конфигурационный файл перемещаемой машины не перемещается на новый узел, поэтому при выключении ВМ она вновь появится на старом хосте. В связи с этим, для совершения полной живой миграции, при котором конфигурация ВМ будет перемещена на новый узел, необходимо воспользоваться утилитой командной строки virsh:

```
virsh migrate --live --persistent --undefinesource \
alt-server qemu+ssh://192.168.0.190/system
```

5.9 Снимки машины

Примечание. Снимок (snapshot) текущего состояния машины можно создать только если виртуальный жесткий диск в формате *.qcow2.

5.9.1 Управления снимками ВМ в консоли

Команда создания снимка (ОЗУ и диск) из файла XML:

```
virsh snapshot-create <domain> [--xmlfile <строка>] [--disk-only] [--live]...
```

Команда создания снимка (ОЗУ и диск) напрямую из набора параметров:

```
virsh snapshot-create-as <domain> [--name <строка>] [--disk-only] [--live]...
```

Пример создания снимка ВМ:

```
# virsh snapshot-create-as --domain alt-server --name alt-server-17dec2021
Снимок домена alt-server-17dec2021 создан
```

где

alt-server – имя ВМ;

alt-server-17dec2021 – название снимка.

После того как снимок ВМ будет сделан, резервные копии файлов конфигураций будут находиться в каталоге `/var/lib/libvirt/qemu/snapshot/`.

Пример создания снимка диска ВМ:

```
# virsh snapshot-create-as --domain alt-server --name 05dec2021 -diskspec vda,file=/var/lib/libvirt/images/sn1.qcow2 --disk-only --atomic
Снимок домена 05dec2021 создан
```

Просмотр существующих снимков для домена alt-server:

```
# virsh snapshot-list --domain alt-server
Имя                    Время создания              Состояние
------------------------------------------------------------------
 05dec2021              2021-12-17 09:47:24 +0200   disk-snapshot
 alt-server-17dec2021   2021-12-17 09:45:29 +0200   running
```

Восстановить ВМ из снимка:

```
# virsh snapshot-revert --domain alt-server --snapshotname 05may2021 –running
```

Удалить снимок:

```
# virsh snapshot-delete --domain alt-server --snapshotname 05may2021
```

5.9.2 Управления снимками ВМ virt-manager

Для управления снимками ВМ в менеджере виртуальных машин virt-manager, необходимо:

1) в главном окне менеджера выбрать ВМ;
2) нажать кнопку «Открыть»;
3) в открывшемся окне нажать кнопку «Управление снимками» (Рис. 243). Появится окно управления снимками ВМ.

Управление снимками ВМ

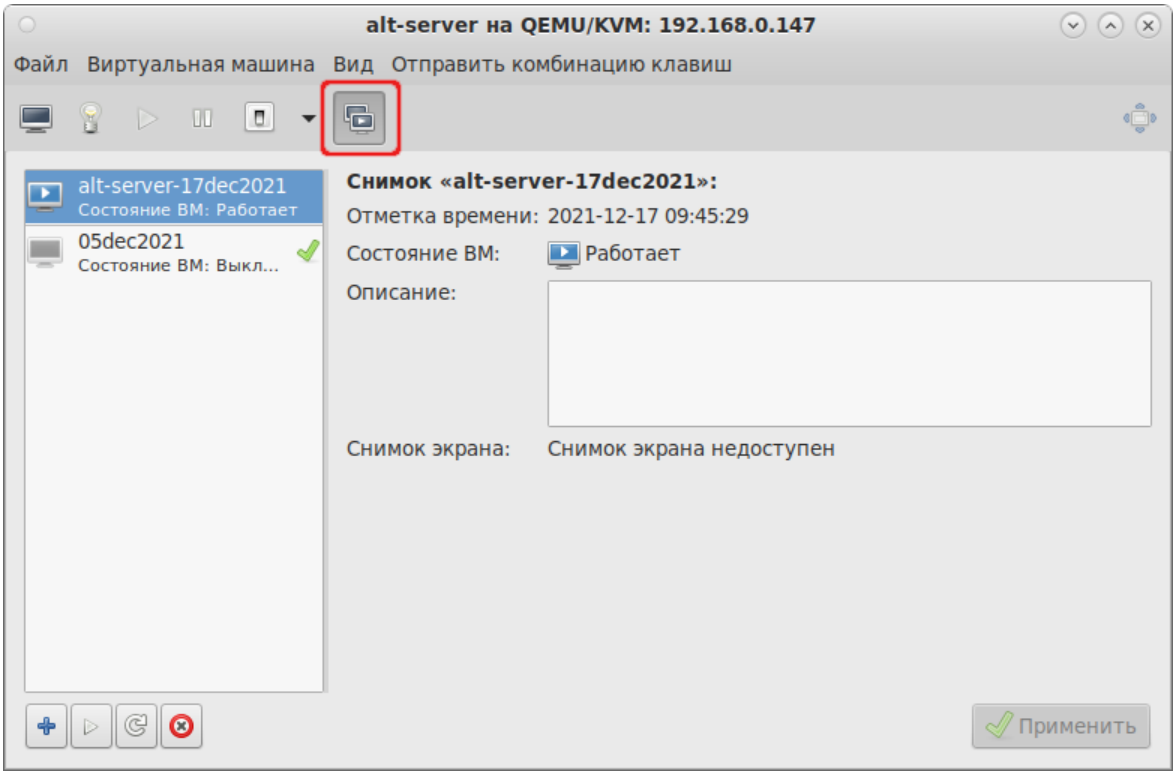


Рис. 243

Для создания нового снимка следует нажать кнопку «Создать новый снимок», расположенную в нижнем левом углу окна управления снимками ВМ. В открывшемся окне (Рис. 244) следует указать название снимка и нажать кнопку «Готово».

Для того чтобы восстановить ВМ из снимка или удалить снимок, следует воспользоваться контекстным меню снимка (Рис. 245).

Создание снимка

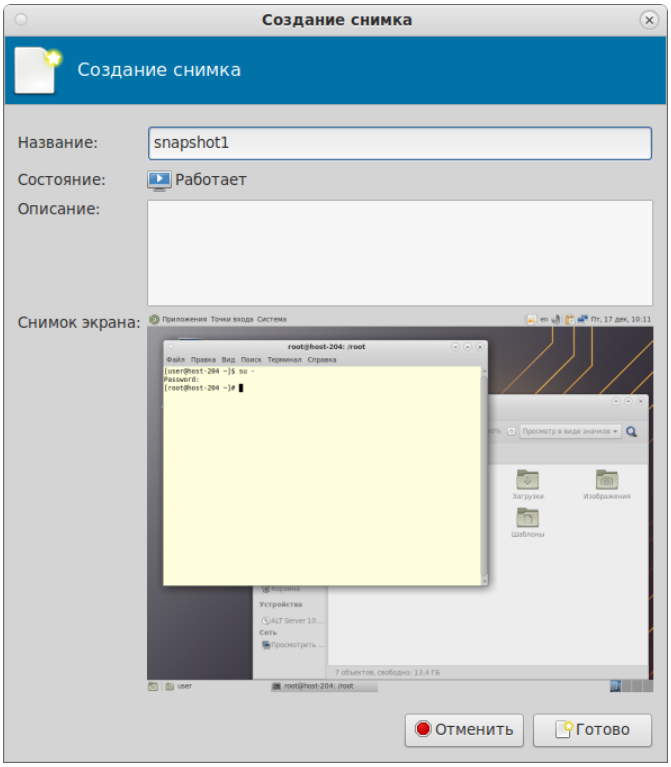


Рис. 244

Контекстное меню снимка

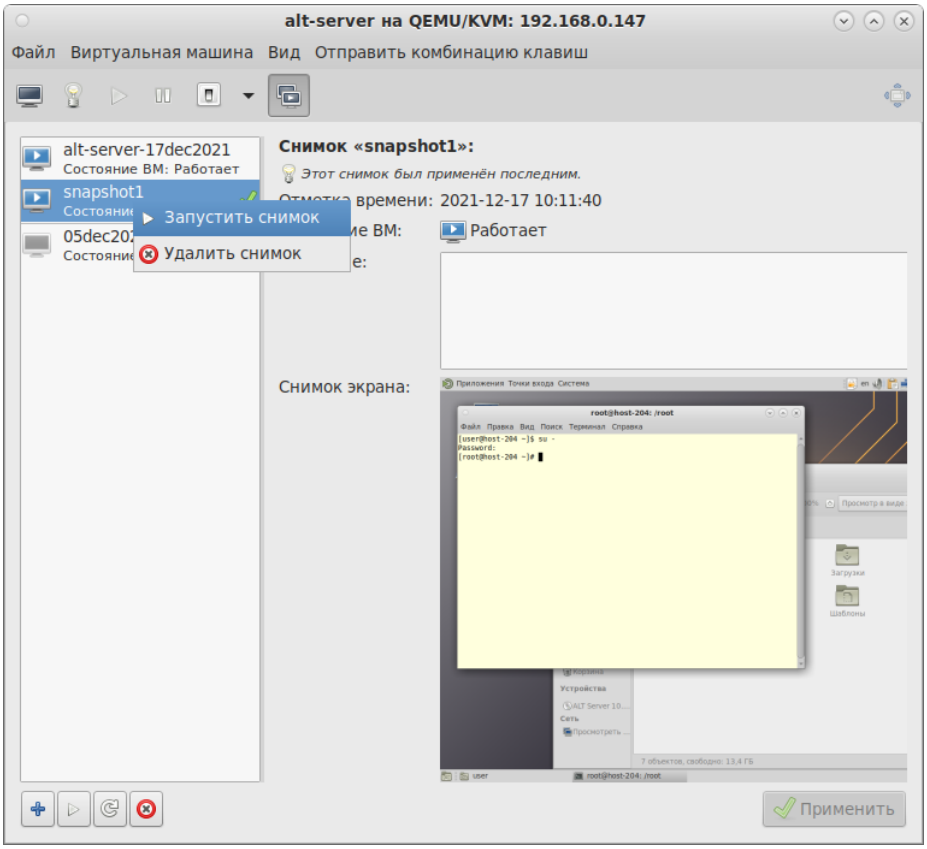


Рис. 245

5.10 Регистрация событий libvirt

Настройка регистрации событий в libvirt, осуществляется в файле `/etc/libvirt/libvirtd.conf`. Логи сохраняются в каталоге `/var/log/libvirt`.

Функция журналирования в libvirt основана на трех ключевых понятиях:

- сообщения журнала;
- фильтры;
- формат ввода.

Сообщения журнала – это информация, полученная во время работы libvirt. Каждое сообщение включает в себя уровень приоритета (отладочное сообщение – 1, информационное – 2, предупреждение – 3, ошибка – 4). По умолчанию, log_level=1, т. е. журналируются все сообщения.

Фильтры – это набор шаблонов и для записи сообщений в журнал. Если категория сообщения совпадает с фильтром, приоритет сообщения сравнивается с приоритетом фильтра, если она ниже сообщение отбрасывается, иначе сообщение записывается в журнал. Если сообщение не соответствует ни одному фильтру, то применяется общий уровень. Это позволяет, например, захватить все отладочные сообщения для Qemu, а для остальных, только сообщения об ошибках.

Формат для фильтра:

```
x:name  (log message only)
x:+name (log message + stack trace)
```

где:

- name – строка, которая сравнивается с заданной категорией, например, remote, qemu, или util.json;
- + – записывать каждое сообщение с данным именем;
- x – минимальный уровень ошибки (1, 2, 3, 4).

Пример фильтра:

```
log_filtrers="3:remote 4:event"
```

Как только сообщение прошло через фильтрацию набора выходных данных, формат вывода определяет, куда отправить сообщение. Формат вывода также может фильтровать на основе приоритета, например, он может быть полезен для вывода всех сообщений в файл отладки.

Формат вывода может быть:

- x:stderr – вывод в STDERR;
- x:syslog:name – использовать системный журнал для вывода и использовать данное имя в качестве идентификатора;
- x:file:file_path – вывод в файл, с соответствующим filepath;

– x:journal – вывод в systemd журнал.

Пример:

```
log_outputs="3:syslog:libvirtd 1:file:/tmp/libvirt.log"
```

Журналы работы виртуальных машин под KVM хранятся в `/var/log/libvirt/qemu/`. В этом каталоге libvirt хранит журнал для каждой ВМ. Например, для машины alt-server журнал будет находиться по адресу: `/var/log/libvirt/qemu/alt-server.log`.

5.11 Управление доступом в виртуальной инфраструктуре

Права пользователя могут управляться с помощью правил polkit.

В каталоге `/usr/share/polkit-1/actions/` имеются два файла с описанием возможных действий для работы с ВМ, предоставленные разработчиками libvirt:

– файл org.libvirt.unix.policy описывает мониторинг ВМ и управление ими;

– в файле org.libvirt.api.policy перечислены конкретные действия (остановка, перезапуск и т. д.), которые возможны, если предыдущая проверка пройдена.

Перечисление конкретных свойств с комментариями доступно в файле `/usr/share/polkit-1/actions/org.libvirt.api.policy`.

В libvirt названия объектов и разрешений отображаются в имена polkit действий, по схеме:

```
org.libvirt.api.$объект.$разрешение
```

Например, разрешение search-storage-vols на объекте storage_pool отображено к действию polkit:

```
org.libvirt.api.storage-pool.search-storage-vols
```

Чтобы определить правила авторизации, polkit должен однозначно определить объект. Libvirt предоставляет ряд атрибутов для определения объектов при выполнении проверки прав доступа. Набор атрибутов изменяется в зависимости от типа объекта.

Пример тонкой настройки

Есть две виртуальные машины: alt1, alt2. Необходимо разрешить пользователю test (должен быть в группе vmusers) действия только с доменом alt1. Для этого необходимо выполнить следующие действия:

1) раскомментировать в файле `/etc/libvirt/libvirtd.conf` строку:

```
access_drivers = [ "polkit" ]
```

2) перезапустить libvirt:

```
# systemctl restart libvirtd
```

3) создать файл `/etc/polkit-1/rules.d/100-libvirt-acl.rules` (имя произвольно) следующего вида:

```
==========================
```

```
polkit.addRule(function(action, subject) {
   // разрешить пользователю test действия с доменом "alt1"
   if (action.id.indexOf("org.libvirt.api.domain.") ==0  &&
       subject.user == "test") {
         if (action.lookup("domain_name") == 'alt1') {
           return polkit.Result.YES;
         }
         else { return polkit.Result.NO; }
   }
   else {
   // разрешить пользователю test действия с
   //подключениями, хранилищем и прочим
if (action.id.indexOf("org.libvirt.api.") == 0 &&
       subject.user == "test") {
       polkit.log("org.libvirt.api.Yes");
           return polkit.Result.YES;
         }
         else { return polkit.Result.NO; }
   }})
================================
```

4) перелогиниться.

В результате выполненных действий пользователю test машина alt1 видна, а машина alt2 – нет.

Права можно настраивать более тонко, например, разрешив пользователю test запускать ВМ, но запретить ему все остальные действия с ней, для этого надо разрешить действие org.libvirt.api.domain.start:

```
=========================
polkit.addRule(function(action, subject) {
   // разрешить пользователю test только запускать ВМ в
   // домене "alt1"
   if (action.id. == "org.libvirt.api.domain.start") &&
       subject.user == "test") {
         if (action.lookup("domain_name") == 'alt1') {
           return polkit.Result.YES;
         }
```

```
         else { return polkit.Result.NO; }
}

   });
=========================
```

Предоставить право запускать ВМ, только пользователям группы wheel:

```
if (action.id == "org.libvirt.api.domain.start") {
        if (subject.isInGroup("wheel")) {
            return polkit.Result.YES;
        } else {
            return polkit.Result.NO;
        }
    };
```

Предоставить право останавливать ВМ, только пользователям группы wheel:

```
if (action.id == "org.libvirt.api.domain.stop") {
        if (subject.isInGroup("wheel")) {
            return polkit.Result.YES;
        } else {
            return polkit.Result.NO;
        }
    };
```

Можно также вести файл журнала, используя правила polkit. Например, делать запись в журнал при старте ВМ:

```
if (action.id.match("org.libvirt.api.domain.start") ) {
           polkit.log("action=" + action);
           polkit.log("subject=" + subject);
           return polkit.Result.YES;
         }
```

Запись в журнал при останове ВМ:

```
if (action.id.match("org.libvirt.api.domain.stop") ) {
           polkit.log("action=" + action);
           polkit.log("subject=" + subject);
           return polkit.Result.YES;
         }
```

6 KUBERNETES

6.1 Краткое описание возможностей

Kubernetes – это система для автоматизации развёртывания, масштабирования и управления контейнеризированными приложениями. Поддерживает основные технологии контейнеризации (Docker, Rocket) и аппаратную виртуализацию.

Основные задачи Kubernetes:

- развертывание контейнеров и все операции для запуска необходимой конфигурации (перезапуск остановившихся контейнеров, перемещение контейнеров для выделения ресурсов на новые контейнеры и т.д.);
- масштабирование и запуск нескольких контейнеров одновременно на большом количестве хостов;
- балансировка множества контейнеров в процессе запуска. Для этого Kubernetes использует API, задача которого заключается в логическом группировании контейнеров.

Утилиты для создания и управления кластером Kubernetes:

- `kubectl` – создание и настройка объектов в кластере;
- `kubelet` – запуск контейнеров на узлах;
- `kubeadm` – настройка компонентов, составляющие кластер.

6.2 Установка и настройка Kubernetes

Для создания управляющего или вычислительного узла, при установке дистрибутива в группе «Контейнеры» следует соответственно отметить пункт «Сервисы Kubernetes для управляющего хоста» или «Сервисы Kubernetes для вычислительного хоста» (Рис. 246).

Примечание. На этапе «Подготовка диска» рекомендуется выбрать «Server KVM/Docker/LXD/Podman/CRI-O (large /var/lib/)» и не создавать раздел Swap.

Примечание. В данном руководстве рассмотрен процесс разворачивания кластера с использованием CRI-O.

Установка Kubernetes при установке системы

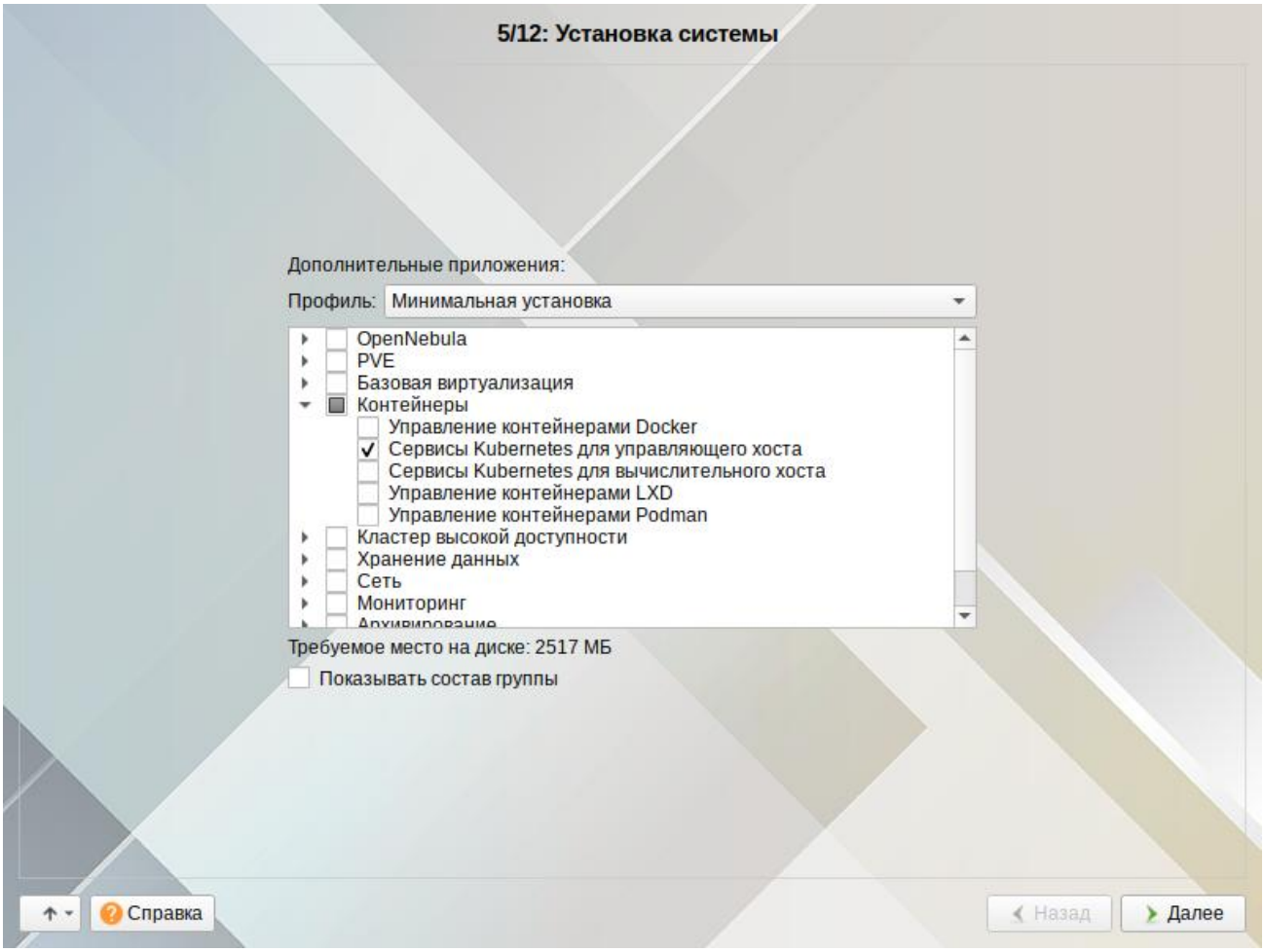


Рис. 246

6.2.1 Создание кластера Kubernetes

Для создания кластера необходимо несколько машин (nodes), одна из которых будет мастером. Системные требования:

- 2 ГБ или больше ОЗУ на машину;
- 2 ядра процессора или больше;
- все машины должны быть доступны по сети друг для друга;
- все машины должны успешно разрешать имена hostname друг друга (через DNS или hosts);
- Swap должен быть выключен.

Примечание. Для отключения Swap нужно выполнить команду:

```
# swapoff -a
```

и удалить соответствующую строку в `/etc/fstab`.

6.2.1.1 Инициализация кластера

Для инициализации кластера запустить одну из двух следующих команд (на мастере):

- для настройки сети с использованием Flannel:

```
# kubeadm init --pod-network-cidr=10.244.0.0/16
```

– для настройки сети с использованием Calico:

```
# kubeadm init --pod-network-cidr=10.168.0.0/16
```

где:

– `--pod-network-cidr=10.244.0.0/16` – адрес внутренней (разворачиваемой Kubernetes) сети, рекомендуется оставить данное значение для правильной работы Flannel;

– `--pod-network-cidr=10.168.0.0/16` – адрес внутренней (разворачиваемой Kubernetes) сети, рекомендуется оставить данное значение для правильной работы Calico.

Если все сделано правильно, на экране отобразится команда, позволяющая присоединить остальные ноды кластера к мастеру:

```
…
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as
root:

kubeadm join 192.168.0.102:6443 --token rcbuiz.o0brh8chmu0i7ljw \
    --discovery-token-ca-cert-hash \
    sha256:b80186633ae51261c29ed4f5c2da68907b1e344f48a52022de413b3bd24191ce
```

Настроить kubernetes для работы от пользователя (на мастер-ноде):

1) создать каталог `~/.kube` (с правами администратора):

```
$ mkdir ~/.kube
```

2) скопировать конфигурацию (с правами администратора):

```
# cp /etc/kubernetes/admin.conf ~<пользователь>/.kube/config
```

3) изменить владельца конфигурационного файла (с правами администратора):

```
# chown <пользователь>: ~<пользователь>/.kube/config
```

6.2.1.2 Настройка сети

Развернуть сеть (Container Network Interface), запустив один из двух наборов команд (на мастер-ноде):

– для Flannel:

```
$ kubectl apply -f https://raw.githubusercontent.com/flannel-
io/flannel/master/Documentation/kube-flannel.yml
```

– для Calico:

```
$ kubectl apply -f https://projectcalico.docs.tigera.io/manifests/tigera-
operator.yaml
$ kubectl apply -f https://projectcalico.docs.tigera.io/manifests/custom-
resources.yaml
```

В выводе будут отображены имена всех созданных ресурсов.

Проверить, что всё работает:

```
$ kubectl get pods --namespace kube-system
NAME                                READY   STATUS    RESTARTS   AGE
coredns-74ff55c5b-5rgmk             1/1     Running   0          38m
coredns-74ff55c5b-wjq4r             1/1     Running   0          38m
etcd-master01                       1/1     Running   0          37m
kube-apiserver-master01             1/1     Running   0          37m
kube-controller-manager-master01    1/1     Running   0          37m
kube-flannel-ds-2gl6g               1/1     Running   0          92s
kube-proxy-gknv7                    1/1     Running   0          15m
kube-scheduler-master01             1/1     Running   0          37m
```

coredns должны находиться в состоянии Running. Количество kube-flannel и kube-proxy зависит от общего числа нод.

6.2.1.3 Добавление узлов (нод) в кластер

Подключить остальные узлы (ноды) в кластер. Для этого на узле выполнить команду:

```
# kubeadm join <ip адрес>:<порт> --token <токен> \
    --discovery-token-ca-cert-hash sha256:<хэш> \
    --ignore-preflight-errors=SystemVerification
```

Данная команда была выведена при выполнении команды `kubeadm init` на мастер-ноде. В данном случае:

```
# kubeadm join 192.168.0.102:6443 --token rcbuiz.o0brh8chmu0i7ljw \
    --discovery-token-ca-cert-hash \
    sha256:b80186633ae51261c29ed4f5c2da68907b1e344f48a52022de413b3bd24191ce
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
```

```
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm
kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file
"/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

Примечание. Получить токен, если его нет, можно выполнив команду (на мастер-ноде):

```
$ kubeadm token list
TOKEN                     TTL      EXPIRES                USAGES
rcbuiz.o0brh8chmu0i7ljw   22h      2021-12-18T11:49:53Z   authentication,signing
```

По умолчанию, срок действия токена – 24 часа. Если требуется добавить новый узел в кластер по окончанию этого периода, можно создать новый токен:

```
$ kubeadm token create
```

Если значение параметра `--discovery-token-ca-cert-hash` неизвестно, его можно получить, выполнив команду (на мастер-ноде):

```
$ openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | \
   openssl rsa -pubin -outform der 2>/dev/null | \
   openssl dgst -sha256 -hex | sed 's/^.* //'
b80186633ae51261c29ed4f5c2da68907b1e344f48a52022de413b3bd24191ce
```

Для ввода IPv6-адреса в параметр `<control-plane-host>:<control-plane-port>`, адрес должен быть заключен в квадратные скобки:

```
[fd00::101]:2073
```

Проверить наличие нод (на мастер-ноде):

```
$ kubectl get nodes
NAME       STATUS   ROLES                  AGE    VERSION
master01   Ready    control-plane,master   42m     v1.22.5
kube02     Ready    <none>                 2m43s   v1.22.5
kube03     Ready    <none>                 24s     v1.22.5
```

или:

```
$ kubectl get nodes -o wide
```

Информация о кластере:

```
$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.0.102:6443
```

```
KubeDNS is running at https://192.168.0.102:6443/api/v1/namespaces/kube-
system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

Посмотреть подробную информацию о ноде:

```
$ kubectl describe node docker03
```

6.2.2 Тестовый запуск nginx

Deployment – это объект Kubernetes, представляющий работающее приложение в кластере.

Создать Deployment с nginx:

```
$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

Список подов:

```
$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-66b6c48dd5-89lq4   1/1     Running   1          9s
nginx-deployment-66b6c48dd5-bt7zp   1/1     Running   1          9s
```

Создать сервис, с помощью которого можно получить доступ к приложению из внешней сети. Для этого создать файл `nginx-service.yaml`, со следующим содержимым:

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  type: NodePort
  ports:
  - port: 80
    targetPort: 80
  selector:
    app: nginx
```

Запустить новый сервис:

```
$ kubectl apply -f nginx-service.yaml
service/nginx created
```

Просмотреть порт сервиса nginx:

```
$ kubectl get svc nginx
NAME    TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
nginx   NodePort   10.98.167.146   <none>        80:31868/TCP   17s
```

Проверить работу nginx, выполнив команду (сервер должен вернуть код 200):

```
$ curl -I <ip адрес>:<порт>
```

где <ip адрес> – это адрес любой из нод (не мастер-ноды), а <порт> – это порт сервиса, полученный с помощью предыдущей команды. В данном кластере возможна команда:

```
$ curl -I 192.168.0.102:31868
HTTP/1.1 200 OK
Server: nginx/1.14.2
```

6.3 Кластер высокой доступности Kubernetes

Kubernetes предлагает два основных способа реализации HA-кластера:

- со стековой топологией (узлы etcd размещаются вместе с узлами плоскости управления);
- с внешней etcd-топологией (etcd работает на отдельных узлах от плоскости управления).

6.3.1 Стековая (составная) топология etcd

Стековая (составная) топология etcd – это топология, в которой распределенный кластер хранения данных, предоставляемый etcd, размещается на вершине кластера, образованного узлами, управляемыми kubeadm, которые запускают компоненты плоскости управления.

Каждый узел плоскости управления запускает экземпляр kube-apiserver, kube-scheduler и kube-controller-manager. Kube-apiserver предоставляется рабочим узлам с помощью балансировщика нагрузки.

Каждый узел уровня управления создает локальный etcd, и этот etcd взаимодействует только с kube-apiserver этого узла. То же самое относится к локальным экземплярам kube-controller-manager и kube-scheduler.

Эту топологию проще настроить, чем кластер с внешними узлами etcd, и проще управлять репликацией. Но если один узел выходит из строя, будут потеряны и etcd, и экземпляр уровня управления, и избыточность нарушится. Этот риск можно снизить, добавив больше узлов плоскости управления. Поэтому для HA-кластера следует запустить как минимум три сгруппированных узла плоскости управления.

Это топология используется по умолчанию в kubeadm. Локальный etcd создается автоматически на узлах плоскости управления при использовании kubeadm init и kubeadm join --control-plane.

6.3.2 Внешняя etcd-топология

Внешняя etcd-топология – это топология, в которой кластер распределенного хранения данных, предоставляемый etcd, является внешним по отношению к кластеру, сформированному узлами, на которых выполняются компоненты плоскости управления.

Каждый узел плоскости управления во внешней топологии etcd запускает экземпляр kube-apiserver, kube-scheduler и kube-controller-manager. И kube-apiserver предоставляется рабочим

узлам с помощью балансировщика нагрузки. Однако etcd работают на отдельных хостах, и каждый хост etcd взаимодействует с kube-apiserver каждого узла плоскости управления.

Эта топология разделяет плоскость управления и элемент etcd. Таким образом обеспечивается настройка HA, при которой потеря экземпляра уровня управления или etcd оказывает меньшее влияние и не влияет на избыточность кластера в такой степени, как многослойная топология.

Но для этой топологии требуется вдвое больше хостов, чем для многослойной топологии. Для HA-кластера с этой топологией требуется как минимум три хоста для узлов плоскости управления и три хоста для узлов etcd.

6.3.3 Создание HA-кластера с помощью kubeadm

Рекомендации:

- три или более управляющих узла;
- три или более вычислительных узла;
- все узлы должны быть доступны по сети друг для друга;
- на всех узлах должны быть установлены kubeadm, kubelet и среда выполнения контейнера;
- каждый узел должен иметь доступ к реестру образов контейнера Kubernetes (k8s.gcr.io);
- возможность доступа по ssh с одного узла ко всем узлам в системе.

Для создания HA-кластера etcd к вышеперечисленным требованиям дополнительно требуется три или более узла, которые станут членами кластера etcd.

Примечание. В данных примерах рассмотрена настройка сети с использованием Flannel.

6.3.3.1 Настройка HA-кластера etcd с помощью kubeadm

На первом управляющем узле необходимо выполнить следующие действия:

1) инициализировать кластер, выполнив команду:

```
# kubeadm init --control-plane-endpoint 192.168.0.201:6443 --upload-certs --pod-network-cidr=10.244.0.0/16
```

где

- `--control-plane-endpoint` – указывает адрес и порт балансировщика нагрузки;
- `--upload-certs` – используется для загрузки в кластер сертификатов, которые должны быть общими для всех управляющих узлов;
- `--pod-network-cidr=10.244.0.0/16` – адрес внутренней (разворачиваемой Kubernetes) сети, рекомендуется оставить данное значение для правильной работы Flannel.

Если все сделано правильно, на экране отобразится команда, позволяющая присоединить остальные узлы кластера к управляющему узлу:

```
…
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of the control-plane node running the following command
on each as root:

kubeadm join 192.168.0.201:6443 --token cvvui8.lz82ufip6cz89ar9 \
      --discovery-token-ca-cert-hash
sha256:3ee0c550746a4a8e0abb6b59311f0fc301cdfeec00af8b26ed4598116c4d8184 \
      --control-plane --certificate-key
e0cbf1dc4e282bf517e23887dace30b411cd739b1aab037b056f0c23e5b0a222

Please note that the certificate-key gives access to cluster sensitive data, keep it
secret!
As a safeguard, uploaded-certs will be deleted in two hours; If necessary, you can
use
"kubeadm init phase upload-certs --upload-certs" to reload certs afterward.

Then you can join any number of worker nodes by running the following on each as
root:

kubeadm join 192.168.0.201:6443 --token cvvui8.lz82ufip6cz89ar9 \
      --discovery-token-ca-cert-hash
sha256:3ee0c550746a4a8e0abb6b59311f0fc301cdfeec00af8b26ed4598116c4d8184
```

2) настроить kubernetes для работы от пользователя:

– создать каталог ~/.kube (с правами пользователя):

```
$ mkdir ~/.kube
```

– скопировать конфигурацию (с правами администратора):

```
# cp /etc/kubernetes/admin.conf ~<пользователь>/.kube/config
```

– изменить владельца конфигурационного файла (с правами администратора):

```
# chown <пользователь>: ~<пользователь>/.kube/config
```

3) развернуть сеть (CNI):

```
$ kubectl apply -f https://raw.githubusercontent.com/flannel-
io/flannel/master/Documentation/kube-flannel.yml
```

4) проверить, что всё работает:

```
$ kubectl get pod -n kube-system -w
NAME                                READY   STATUS              RESTARTS   AGE
coredns-78fcd69978-c5swn            0/1     ContainerCreating   0          11m
coredns-78fcd69978-zdbp8            0/1     ContainerCreating   0          11m
etcd-master01                       1/1     Running             0          11m
kube-apiserver-master01             1/1     Running             0          11m
kube-controller-manager-master01    1/1     Running             0          11m
kube-flannel-ds-qfzbw               1/1     Running             0          116s
kube-proxy-r6kj9                    1/1     Running             0          11m
kube-scheduler-master01             1/1     Running             0          11m
```

На остальных управляющих узлах выполнить команду подключения узла к кластеру (данная команда была выведена при выполнении команды `kubeadm init` на первом управляющем узле):

```
# kubeadm join 192.168.0.201:6443 --token cvvui8.lz82ufip6cz89ar9 \
--discovery-token-ca-cert-hash
sha256:3ee0c550746a4a8e0abb6b59311f0fc301cdfeec00af8b26ed4598116c4d8184 \
--control-plane --certificate-key
e0cbf1dc4e282bf517e23887dace30b411cd739b1aab037b056f0c23e5b0a222
```

Подключить вычислительные узлы к кластеру (данная команда была выведена при выполнении команды `kubeadm init` на первом управляющем узле):

```
# kubeadm join 192.168.0.201:6443 --token cvvui8.lz82ufip6cz89ar9 \
--discovery-token-ca-cert-hash
sha256:3ee0c550746a4a8e0abb6b59311f0fc301cdfeec00af8b26ed4598116c4d8184
```

Проверить наличие нод (на управляющем узле):

```
$ kubectl get nodes
NAME       STATUS   ROLES                  AGE     VERSION
kube01     Ready    <none>                 23m     v1.22.5
kube02     Ready    <none>                 15m     v1.22.5
kube03     Ready    <none>                 2m30s   v1.22.5
master01   Ready    control-plane,master   82m     v1.22.5
master02   Ready    control-plane,master   66m     v1.22.5
master03   Ready    control-plane,master   39m     v1.22.5
```

6.3.3.2 Настройка HA-кластера etcd с помощью kubeadm

Настройка HA-кластера с внешней etcd-топологией аналогична процедуре, используемой для стековой топологии etcd, за исключением того, что предварительно необходимо настроить etcd и передать информацию etcd в конфигурационный файл kubeadm.

В данном примере рассматривается процесс создания HA-кластера etcd состоящего из трех узлов. Узлы должны иметь возможность общаться друг с другом через порты 2379 и 2380.

Основная идея при таком способе настройки кластера, состоит в том, чтобы генерировать все сертификаты на одном узле и распространять только необходимые файлы на другие узлы.

Примечание. kubeadm содержит все необходимое криптографические механизмы для создания сертификатов, никаких других криптографических инструментов для данного примера не требуется.

Настройка etcd кластера:

1) настроить kubelet в качестве диспетчера служб для etcd. Для этого на всех etcd узлах нужно добавить новый файл конфигурации systemd для модуля kubelet с более высоким приоритетом, чем предоставленный kubeadm файл модуля kubelet:

```
# cat <<EOF > /etc/systemd/system/kubelet.service.d/20-etcd-service-manager.conf
[Service]
ExecStart=
# Replace "systemd" with the cgroup driver of your container runtime. The default
value in the kubelet is "cgroupfs".
# Replace the value of "--container-runtime-endpoint" for a different container
runtime if needed.
ExecStart=/usr/bin/kubelet --address=127.0.0.1 --pod-manifest-
path=/etc/kubernetes/manifests --cgroup-driver=systemd --container-runtime=remote --
container-runtime-endpoint=unix:///var/run/crio/crio.sock
Restart=always
EOF

# systemctl daemon-reload
# systemctl restart kubelet
```

Убедиться, что kubelet запущен:

```
# systemctl status kubelet
```

2) на первом узле etcd создать файлы конфигурации kubeadm для всех узлов etcd. Для этого создать и запустить скрипт:

```
#!/bin/sh
# HOST0, HOST1, и HOST2 - IP-адреса узлов
export HOST0=192.168.0.205
export HOST1=192.168.0.206
```

```
export HOST2=192.168.0.207

# NAME0, NAME1 и NAME2 - имена узлов
export NAME0="etc01"
export NAME1="etc02"
export NAME2="etc03"

# Создать временные каталоги
mkdir -p /tmp/${HOST0}/ /tmp/${HOST1}/ /tmp/${HOST2}/

HOSTS=(${HOST0} ${HOST1} ${HOST2})
NAMES=(${NAME0} ${NAME1} ${NAME2})

for i in "${!HOSTS[@]}"; do
HOST=${HOSTS[$i]}
NAME=${NAMES[$i]}
cat >> EOF < /tmp/${HOST}/kubeadmcfg.yaml
---
apiVersion: "kubeadm.k8s.io/v1beta3"
kind: InitConfiguration
nodeRegistration:
    name: ${NAME}
localAPIEndpoint:
    advertiseAddress: ${HOST}
---
apiVersion: "kubeadm.k8s.io/v1beta3"
kind: ClusterConfiguration
etcd:
    local:
        serverCertSANs:
        - "${HOST}"
        peerCertSANs:
        - "${HOST}"
        extraArgs:
            initial-cluster:
${NAMES[0]}=https://${HOSTS[0]}:2380,${NAMES[1]}=https://${HOSTS[1]}:2380,${NAMES[2]}
=https://${HOSTS[2]}:2380
            initial-cluster-state: new
            name: ${NAME}
            listen-peer-urls: https://${HOST}:2380
            listen-client-urls: https://${HOST}:2379
            advertise-client-urls: https://${HOST}:2379
```

```
            initial-advertise-peer-urls: https://${HOST}:2380
EOF
done
```

3) создать центр сертификации (CA).

Примечание. Если у вас уже есть CA, то необходимо скопировать сертификат (crt) и ключ CA в /etc/kubernetes/pki/etcd/ca.crt и /etc/kubernetes/pki/etcd/ca.key. После этого можно перейти к следующему шагу.

Если у вас еще нет CA, следует на узле, где были сгенерированы файлы конфигурации kubeadm, запустить команду:

```
# kubeadm init phase certs etcd-ca
[certs] Generating "etcd/ca" certificate and key
```

Эта команда создаст два файла: /etc/kubernetes/pki/etcd/ca.crt и /etc/kubernetes/pki/etcd/ca.key.

4) сгенерировать сертификаты для всех etcd узлов. Для этого создать и запустить скрипт (на первом etcd узле):

```
#!/bin/sh
# HOST0, HOST1, и HOST2 - IP-адреса узлов
export HOST0=192.168.0.205
export HOST1=192.168.0.206
export HOST2=192.168.0.207
kubeadm init phase certs etcd-server --config=/tmp/${HOST2}/kubeadmcfg.yaml
kubeadm init phase certs etcd-peer --config=/tmp/${HOST2}/kubeadmcfg.yaml
kubeadm init phase certs etcd-healthcheck-client --
config=/tmp/${HOST2}/kubeadmcfg.yaml
kubeadm init phase certs apiserver-etcd-client --config=/tmp/${HOST2}/kubeadmcfg.yaml
cp -R /etc/kubernetes/pki /tmp/${HOST2}/
# cleanup non-reusable certificates
find /etc/kubernetes/pki -not -name ca.crt -not -name ca.key -type f -delete

kubeadm init phase certs etcd-server --config=/tmp/${HOST1}/kubeadmcfg.yaml
kubeadm init phase certs etcd-peer --config=/tmp/${HOST1}/kubeadmcfg.yaml
kubeadm init phase certs etcd-healthcheck-client --
config=/tmp/${HOST1}/kubeadmcfg.yaml
kubeadm init phase certs apiserver-etcd-client --config=/tmp/${HOST1}/kubeadmcfg.yaml
cp -R /etc/kubernetes/pki /tmp/${HOST1}/
find /etc/kubernetes/pki -not -name ca.crt -not -name ca.key -type f -delete

kubeadm init phase certs etcd-server --config=/tmp/${HOST0}/kubeadmcfg.yaml
kubeadm init phase certs etcd-peer --config=/tmp/${HOST0}/kubeadmcfg.yaml
kubeadm init phase certs etcd-healthcheck-client --
config=/tmp/${HOST0}/kubeadmcfg.yaml
```

```
kubeadm init phase certs apiserver-etcd-client --config=/tmp/${HOST0}/kubeadmcfg.yaml
# No need to move the certs because they are for HOST0

# clean up certs that should not be copied off this host
find /tmp/${HOST2} -name ca.key -type f -delete
find /tmp/${HOST1} -name ca.key -type f -delete
```

5) скопировать сертификаты и файлы конфигурации kubeadm на на второй и третий узлы etcd:

```
HOST1=192.168.0.206
HOST2=192.168.0.207
USER=user
# scp -r /tmp/${HOST1}/* ${USER}@${HOST1}:
# ssh ${USER}@${HOST1}
$ su -
# chown -R root:root /home/user/pki
# mv /home/user/pki /etc/kubernetes/
# exit
$ exit
# scp -r /tmp/${HOST2}/* ${USER}@${HOST2}:
# ssh ${USER}@${HOST2}
$ su -
# chown -R root:root /home/user/pki
# mv /home/user/pki /etc/kubernetes/
# exit
$ exit
```

6) в итоге должны существовать следующие файлы:

– на первом узле etcd (там, где были сгенерированы файлы конфигурации для kubeadm и сертификаты):

```
/tmp/${HOST0}
└── kubeadmcfg.yaml
---
/etc/kubernetes/pki
├── apiserver-etcd-client.crt
├── apiserver-etcd-client.key
└── etcd
    ├── ca.crt
    ├── ca.key
    ├── healthcheck-client.crt
    ├── healthcheck-client.key
    ├── peer.crt
    ├── peer.key
```

```
    ├── server.crt
    └── server.key
```

– на втором узле etcd:

```
$HOME
└── kubeadmcfg.yaml
---
/etc/kubernetes/pki
├── apiserver-etcd-client.crt
├── apiserver-etcd-client.key
└── etcd
    ├── ca.crt
    ├── healthcheck-client.crt
    ├── healthcheck-client.key
    ├── peer.crt
    ├── peer.key
    ├── server.crt
    └── server.key
```

– на третьем узле etcd:

```
$HOME
└── kubeadmcfg.yaml
---
/etc/kubernetes/pki
├── apiserver-etcd-client.crt
├── apiserver-etcd-client.key
└── etcd
    ├── ca.crt
    ├── healthcheck-client.crt
    ├── healthcheck-client.key
    ├── peer.crt
    ├── peer.key
    ├── server.crt
    └── server.key
```

7) на каждом etcd узле запустить команду kubeadm, чтобы сгенерировать статический манифест для etcd:

– на первом узле etcd (там, где были сгенерированы файлы конфигурации kubeadm и сертификаты):

```
# kubeadm init phase etcd local --config=/tmp/192.168.0.205/kubeadmcfg.yaml
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests
```

– на втором и третьем узлах etcd:

```
# kubeadm init phase etcd local --config=/home/user/kubeadmcfg.yaml
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
```

Настроить первый управляющий узел кластера:

1) скопировать сертификаты и ключ с первого узла etcd на первый управляющий узел:

```
export CONTROL_PLANE="user@192.168.0.201"
# scp /etc/kubernetes/pki/etcd/ca.crt "${CONTROL_PLANE}":
# scp /etc/kubernetes/pki/apiserver-etcd-client.crt "${CONTROL_PLANE}":
# scp /etc/kubernetes/pki/apiserver-etcd-client.key "${CONTROL_PLANE}":
```

2) создать на первом управляющем узле файл kubeadm-config.yaml:

```
---
apiVersion: kubeadm.k8s.io/v1beta3
kind: ClusterConfiguration
kubernetesVersion: stable
networking:
    podSubnet: "10.244.0.0/16"
controlPlaneEndpoint: "192.168.0.201:6443" # IP-адрес, порт балансировщика нагрузки
etcd:
   external:
      endpoints:
          - https://192.168.0.205:2379 # IP-адрес ETCD01
          - https://192.168.0.206:2379 # IP-адрес ETCD02
          - https://192.168.0.207:2379 # IP-адрес ETCD03
     caFile: /etc/kubernetes/pki/etcd/ca.crt
     certFile: /etc/kubernetes/pki/apiserver-etcd-client.crt
     keyFile: /etc/kubernetes/pki/apiserver-etcd-client.key
```

3) переместить ранее скопированные сертификаты и ключ в соответствующий каталог на первом управляющем узле:

```
# mkdir -p /etc/kubernetes/pki/etcd/
# cp /home/user/ca.crt /etc/kubernetes/pki/etcd/
# cp /home/user/apiserver-etcd-client.* /etc/kubernetes/pki/
```

4) создать первый управляющий узел:

```
# kubeadm init --config kubeadm-config.yaml --upload-certs
…
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:
```

```
  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of the control-plane node running the following command
on each as root:

  kubeadm join 192.168.0.201:6443 --token 7onha1.afzqd41s8dzr1wj1 \
      --discovery-token-ca-cert-hash
sha256:ec2be69db54b2ae13c175765ddd058801fd70054508c0e118020896a1d4c9ec3 \
      --control-plane --certificate-key
eb1fabf70e994c061f749f13c0f26baef64764e813d5f0eaa7b09d5279a492c4

Please note that the certificate-key gives access to cluster sensitive data, keep it
secret!
As a safeguard, uploaded-certs will be deleted in two hours; If necessary, you can
use
"kubeadm init phase upload-certs --upload-certs" to reload certs afterward.

Then you can join any number of worker nodes by running the following on each as
root:

kubeadm join 192.168.0.201:6443 --token 7onha1.afzqd41s8dzr1wj1 \
      --discovery-token-ca-cert-hash
sha256:ec2be69db54b2ae13c175765ddd058801fd70054508c0e118020896a1d4c9ec3
```

Следует сохранить этот вывод, т.к. этот токен будет использоваться для присоединения к кластеру остальных управляющих и вычислительных узлов.

5) настроить kubernetes для работы от пользователя:

– создать каталог ~/.kube (с правами пользователя):

```
$ mkdir ~/.kube
```

– скопировать конфигурацию (с правами администратора):

```
# cp /etc/kubernetes/admin.conf ~<пользователь>/.kube/config
```

– изменить владельца конфигурационного файла (с правами администратора):

```
# chown <пользователь>: ~<пользователь>/.kube/config
```

6) развернуть сеть (CNI):

```
$ kubectl apply -f https://raw.githubusercontent.com/flannel-
io/flannel/master/Documentation/kube-flannel.yml
```

7) проверить, что всё работает:

```
$ kubectl get pod -n kube-system -w
```

На остальных управляющих узлах выполнить команду подключения узла к кластеру (данная команда была выведена при выполнении команды kubeadm init на первом управляющем узле):

```
# kubeadm join 192.168.0.201:6443 --token 7onha1.afzqd41s8dzr1wj1 \
      --discovery-token-ca-cert-hash
sha256:ec2be69db54b2ae13c175765ddd058801fd70054508c0e118020896a1d4c9ec3 \
      --control-plane --certificate-key
eb1fabf70e994c061f749f13c0f26baef64764e813d5f0eaa7b09d5279a492c4
```

Подключить вычислительные узлы к кластеру (данная команда была выведена при выполнении команды kubeadm init на первом управляющем узле):

```
# kubeadm join 192.168.0.201:6443 --token 7onha1.afzqd41s8dzr1wj1 \
      --discovery-token-ca-cert-hash
sha256:ec2be69db54b2ae13c175765ddd058801fd70054508c0e118020896a1d4c9ec3
```

Проверить наличие нод (на управляющем узле):

```
$ kubectl get nodes
```

7 НАСТРОЙКА СИСТЕМЫ

7.1 Центр управления системой

Для управления настройками установленной системы можно использовать Центр управления системой. Центр управления системой (ЦУС) представляет собой удобный интерфейс для выполнения наиболее востребованных административных задач: добавление и удаление пользователей, настройка сетевых подключений, просмотр информации о состоянии системы и т.п. ЦУС имеет веб-ориентированный интерфейс, позволяющий управлять сервером с любого компьютера сети.

Центр управления системой состоит из нескольких независимых диалогов-модулей. Каждый модуль отвечает за настройку определённой функции или свойства системы. Модули центра управления системой имеют справочную информацию.

7.1.1 Применение ЦУС

ЦУС можно использовать для разных целей, например:

- настройка даты и времени;
- управление системными службами;
- просмотр системных журналов;
- управление выключением удаленного компьютера;
- настройка ограничений выделяемых ресурсов памяти пользователям (квоты);
- настройка ограничений на использование внешних носителей;
- конфигурирование сетевых интерфейсов;
- настройка межсетевого экрана;
- изменения пароля администратора системы (root);
- создание, удаление и редактирование учётных записей пользователей.

7.1.2 Использование веб-ориентированного ЦУС

ЦУС имеет веб-ориентированный интерфейс, позволяющий управлять данным компьютером с любого другого компьютера сети.

Для запуска веб-ориентированного интерфейса должен быть запущен сервис ahttpd и alteratord:

```
# systemctl enable --now ahttpd
# systemctl enable --now alteratord
```

Работа с ЦУС может происходить из любого веб-браузера. Для начала работы необходимо перейти по адресу https://ip-адрес:8080/.

Если для сервера задан IP-адрес 192.168.0.122, то интерфейс управления будет доступен по адресу: https://192.168.0.122:8080/.

Примечание. IP-адрес сервера можно узнать, введя на сервере команду:

```
$ ip addr
```

При запуске ЦУС необходимо ввести в соответствующие поля имя пользователя (root) и пароль пользователя root (Рис. 247).

Запуск веб-ориентированного центра управления системой

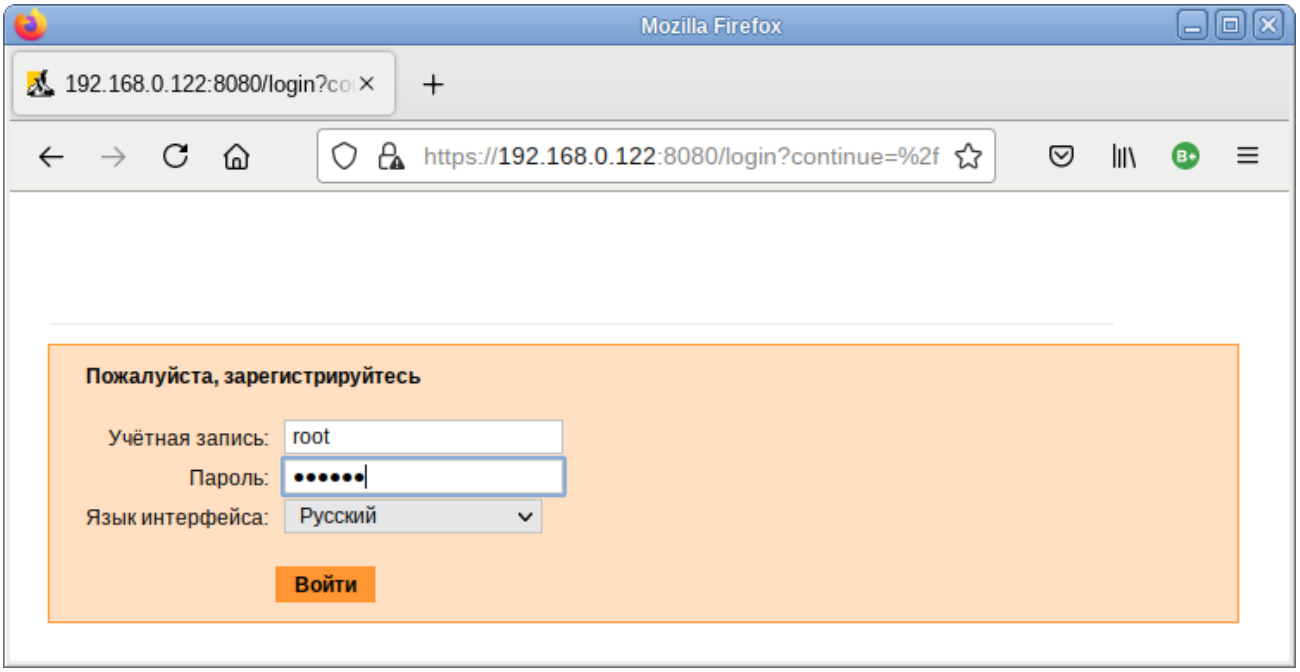


Рис. 247

После этого будут доступны все возможности ЦУС на той машине, к которой было произведено подключение через веб-интерфейс (Рис. 248).

Веб-интерфейс ЦУС можно настроить (кнопка «Режим эксперта»), выбрав один из режимов:

- основной режим;
- режим эксперта.

Выбор режима влияет на количество отображаемых модулей. В режиме эксперта отображаются все модули, а в основном режиме только наиболее используемые.

ЦУС содержит справочную информацию по включённым в него модулям. Об использовании самого интерфейса системы управления можно прочитать, нажав, на кнопку «Справка» на начальной странице центра управления системой (Рис. 248).

После работы с ЦУС, в целях безопасности, не следует оставлять открытым браузер. Необходимо обязательно выйти из сеанса работы с ЦУС, нажав на кнопку «Выйти».

Веб-ориентированный центр управления системой

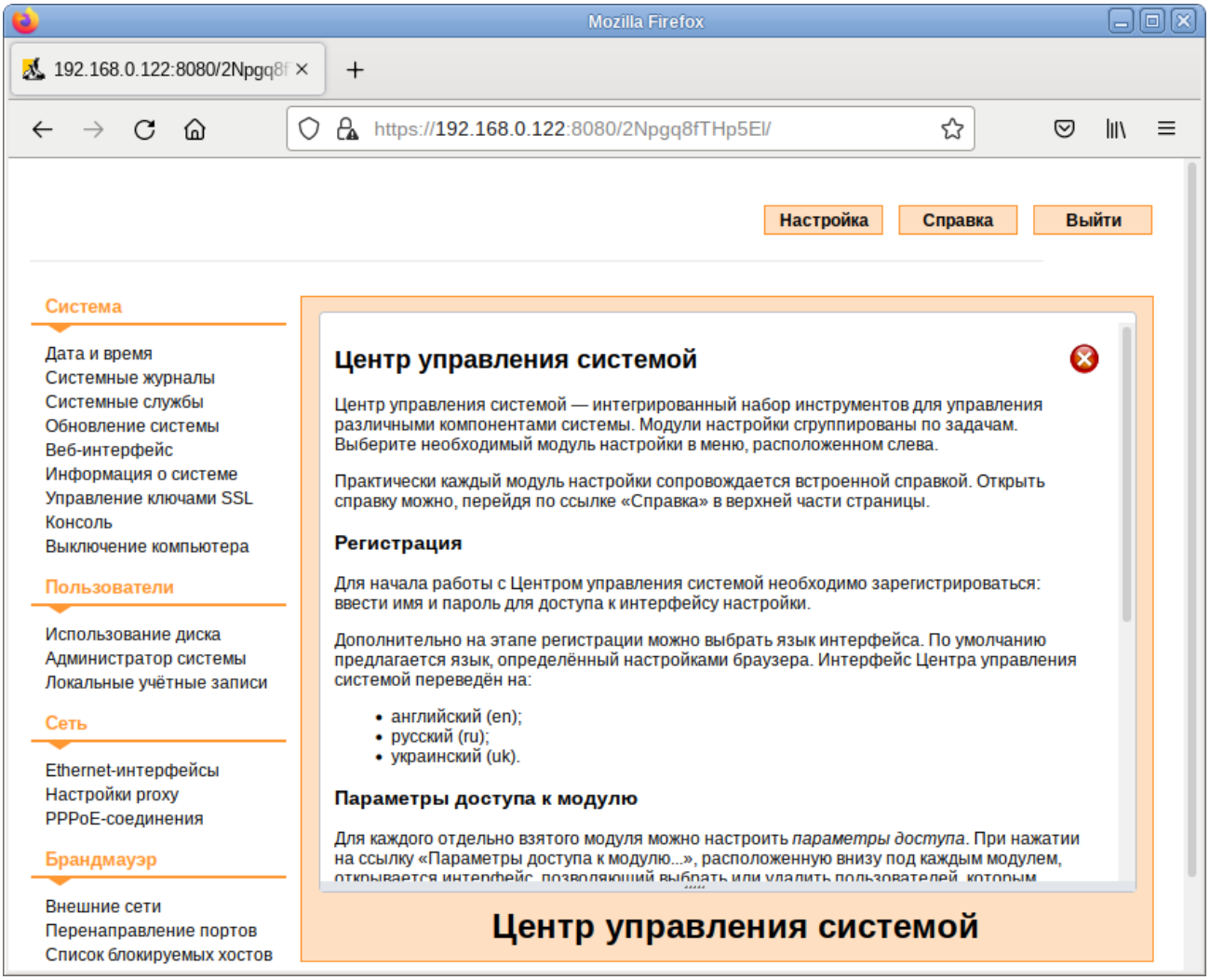


Рис. 248

Дальнейшие разделы описывают некоторые возможности использования ОС «Альт Сервер Виртуализации», настраиваемые в ЦУС.

7.2 Конфигурирование сетевых интерфейсов

ОС «Альт Сервер Виртуализации» поддерживает самые разные способы подключения к сети Интернет:

– Ethernet;
– PPTP;
– PPPoE;
– и т.д.

Для настройки подключения можно воспользоваться одним из разделов ЦУС «Сеть»:

– Ethernet-интерфейсы;
– PPTP-соединения;

– PPPoE-соединения;
– OpenVPN-соединения.

Конфигурирование сетевых интерфейсов осуществляется в модуле ЦУС «Ethernet-интерфейсы» (пакет alterator-net-eth) из раздела раздел «Сеть» (Рис. 249).

Настройка модуля «Ethernet-интерфейсы»

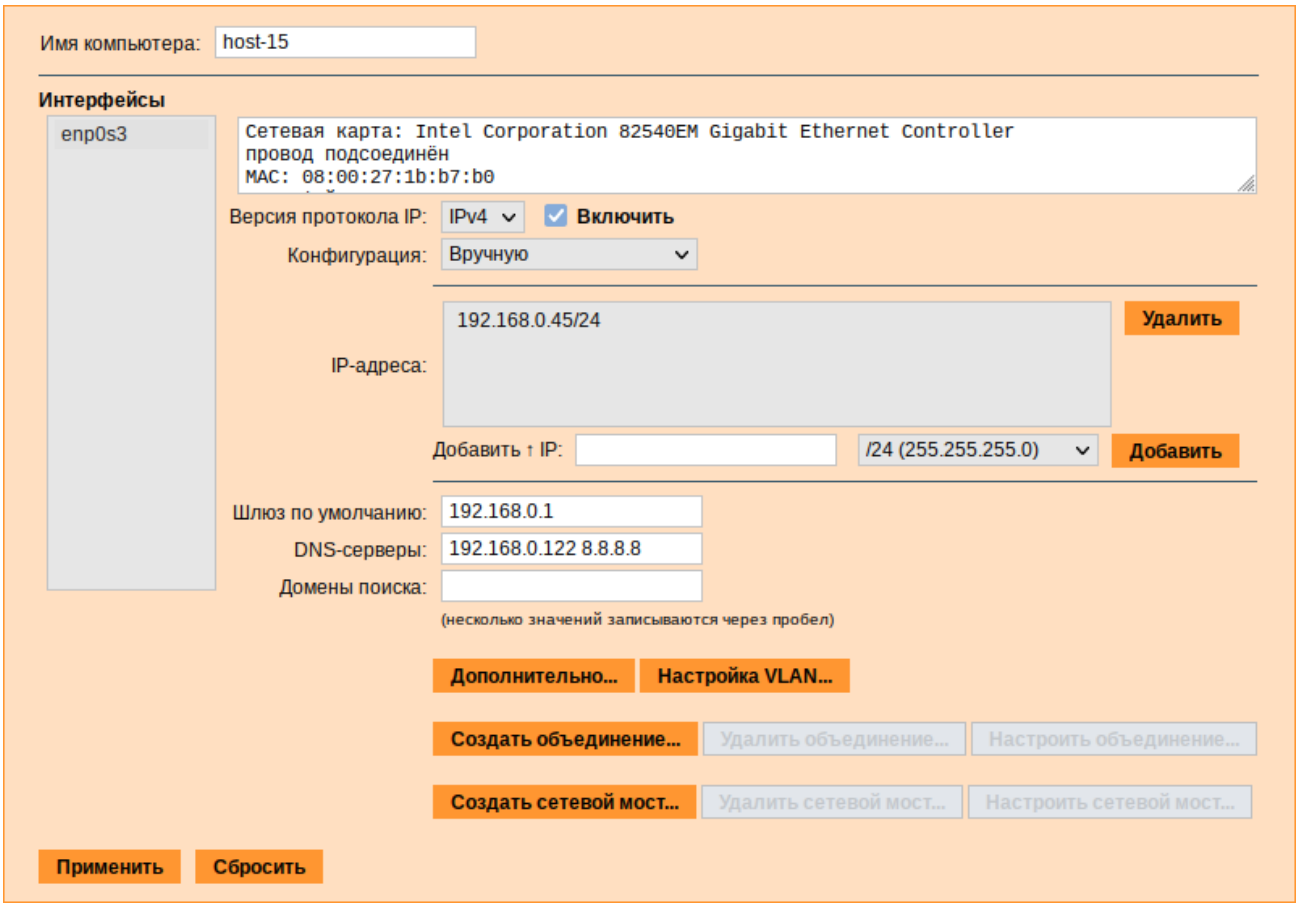


Рис. 249

В модуле «Ethernet-интерфейсы» можно заполнить следующие поля:

– «Имя компьютера» – указать сетевое имя ПЭВМ в поле для ввода имени компьютера (это общий сетевой параметр, не привязанный к какому либо конкретному интерфейсу). Имя компьютера, в отличие от традиционного имени хоста в Unix (hostname), не содержит названия сетевого домена;
– «Интерфейсы» – выбрать доступный сетевой интерфейс, для которого будут выполняться настройки;
– «Версия протокола IP» – указать в выпадающем списке версию используемого протокола IP (IPv4, IPv6) и убедиться, что пункт «Включить», обеспечивающий поддержку работы протокола, отмечен;
– «Конфигурация» – выбрать способ назначения IP-адресов (службы DHCP, Zeroconf, вручную);

- «IP-адреса» – пул назначенных IP-адресов из поля «IP», выбранные адреса можно удалить нажатием кнопки «Удалить»;
- «Добавить ↑ IP» – ввести IP-адрес вручную и выбрать в выпадающем поле предпочтительную маску сети, затем нажать кнопку «Добавить» для переноса адреса в пул поля «IP-адреса»;
- «Шлюз по умолчанию» – в поле для ввода необходимо ввести адрес шлюза, который будет использоваться сетью по умолчанию;
- «DNS-серверы» – в поле для ввода необходимо ввести список предпочтительных DNS-серверов, которые будут получать информацию о доменах, выполнять маршрутизацию почты и управлять обслуживающими узлами для протоколов в домене;
- «Домены поиска» – в поле для ввода необходимо ввести список предпочтительных доменов, по которым будет выполняться поиск.

«IP-адрес» и «Маска сети» – обязательные параметры каждого узла IP-сети. Первый параметр – уникальный идентификатор машины, от второго напрямую зависит, к каким машинам локальной сети данная машина будет иметь доступ. Если требуется выход во внешнюю сеть, то необходимо указать параметр «Шлюз по умолчанию».

В случае наличия DHCP-сервера можно все вышеперечисленные параметры получить автоматически – выбрав в списке «Конфигурация» пункт «Использовать DHCP» (Рис. 250).

Автоматическое получение настроек от DHCP-сервера

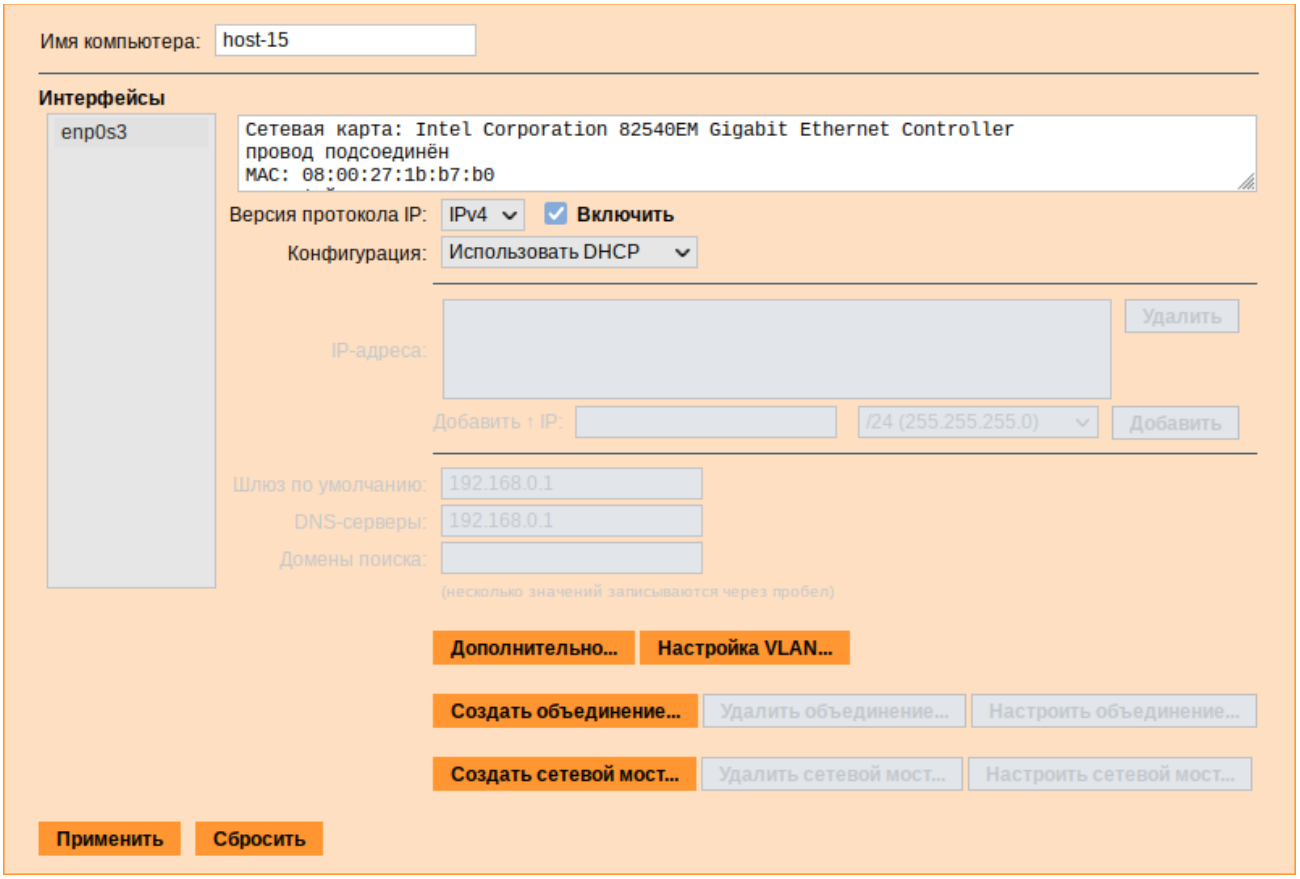


Рис. 250

Если в компьютере имеется несколько сетевых карт, то возможна ситуация, когда при очередной загрузке ядро присвоит имена интерфейсов (enp0s3, enp0s8) в другом порядке. В результате интерфейсы получат не свои настройки. Чтобы этого не происходило, можно привязать интерфейс к имени по его аппаратному адресу (MAC) или по местоположению на системной шине.

Дополнительно для каждого интерфейса можно настроить сетевую подсистему, а также указать должен ли запускаться данный интерфейс при загрузке системы (Рис. 251).

Выбор сетевой подсистемы

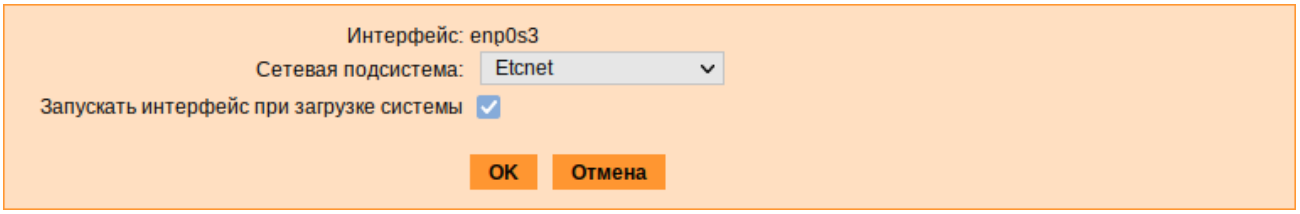


Рис. 251

В списке «Сетевая подсистема» можно выбрать следующие режимы:

– «Etcnet» – в этом режиме настройки берутся исключительно из файлов находящихся в каталоге настраиваемого интерфейса `/etc/net/ifaces/<интерфейс>`. Настройки сети

могут изменяться либо в ЦУС в данном модуле, либо напрямую через редактирование файлов `/etc/net/ifaces/<интерфейс>`;

- «NetworkManager (etcnet)» – в этом режиме NetworkManager сам инициирует сеть, используя в качестве параметров – настройки из файлов Etcnet. Настройки сети могут изменяться либо в ЦУС в данном модуле, либо напрямую через редактирование файлов `/etc/net/ifaces/<интерфейс>`. В этом режиме можно просмотреть настройки сети, например полученный по DHCP IP-адрес, через графический интерфейс NetworkManager;
- «NetworkManager (native)» – в данном режиме управление настройками интерфейса передаётся NetworkManager и не зависит от файлов Etcnet. Управлять настройками можно через графический интерфейс NetworkManager. Файлы с настройками находятся в каталоге `/etc/NetworkManager/system-connections`. Этот режим особенно актуален для задач настройки сети на клиенте, когда IP-адрес необходимо получать динамически с помощью DHCP, а DNS-сервер указать явно. Через ЦУС так настроить невозможно, так как при включении DHCP отключаются настройки, которые можно задавать вручную;
- «Не контролируется» – в этом режиме интерфейс находится в состоянии DOWN (выключен).

7.3 Доступ к службам сервера из сети Интернет

7.3.1 Внешние сети

ОС предоставляет возможность организовать доступ к своим службам извне. Для обеспечения такой возможности необходимо разрешить входящие соединения на внешних интерфейсах. По умолчанию такие соединения блокируются.

Для разрешения внешних и внутренних входящих соединений предусмотрен раздел ЦУС «Брандмауэр». В списке «Разрешить входящие соединения на внешних интерфейсах» модуля «Внешние сети» (пакет alterator-net-iptables) перечислены наиболее часто используемые службы, отметив которые, можно сделать их доступными для соединений на внешних сетевых интерфейсах (Рис. 252). Если необходимо предоставить доступ к службе, отсутствующей в списке, то нужно задать используемые этой службой порты в соответствующих полях.

Модуль «Внешние сети»

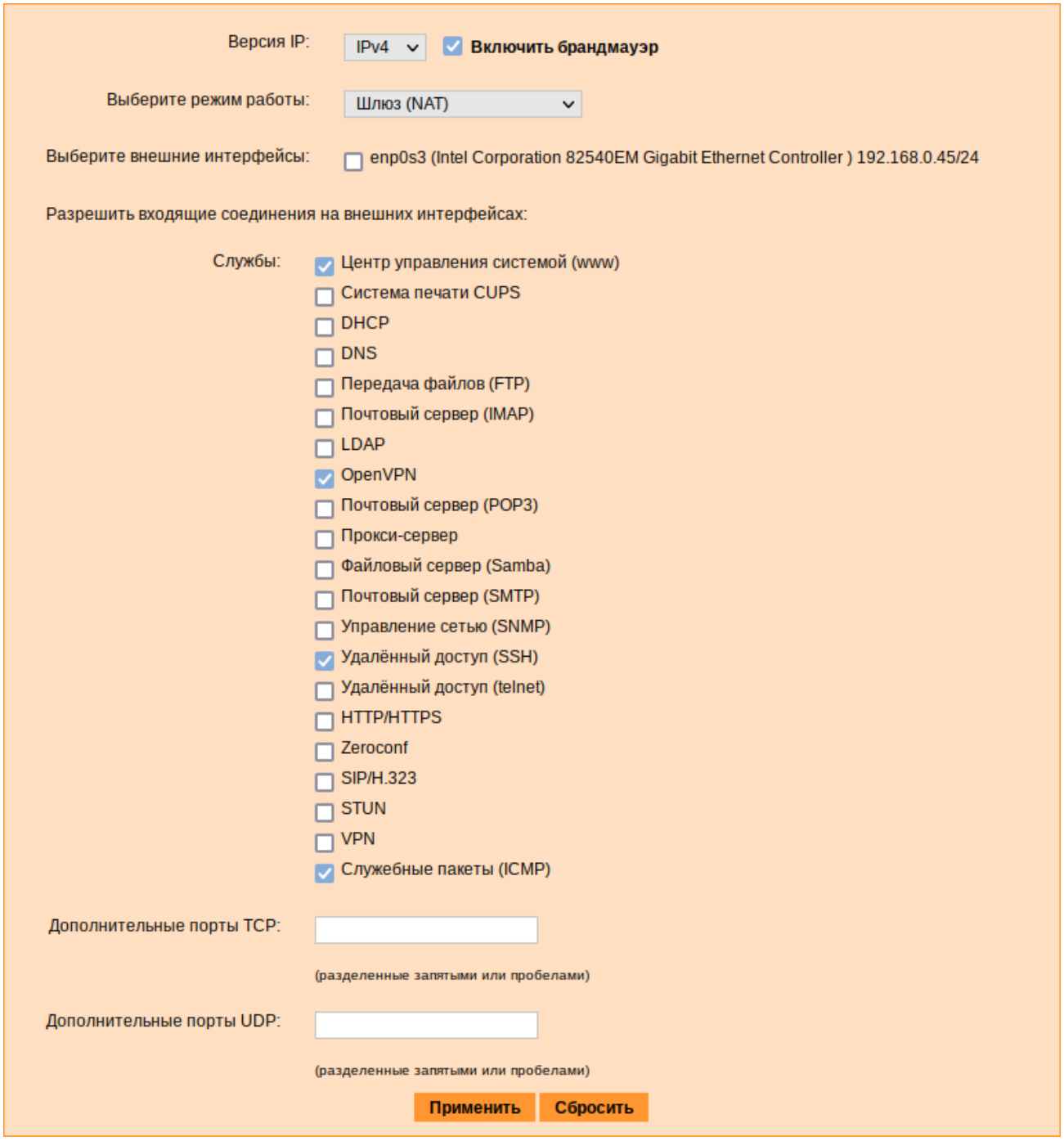


Рис. 252

Можно выбрать один из трех режимов работы:

– роутер – перенаправление пакетов между сетевыми интерфейсами происходит без трансляции сетевых адресов;

– шлюз (NAT) – в этом режиме будет настроена трансляция сетевых адресов (NAT) при перенаправлении пакетов на внешние интерфейсы. Использование этого режима имеет смысл, если на компьютере настроен, по крайней мере, один внешний и один внутренний интерфейс;

– Хост (Рабочая станция) – в этом режиме можно для всех интерфейсов открыть или закрыть порт. Внешними автоматически выбираются все интерфейсы, кроме lo и специальных исключений (virbr*, docker*).

В любом режиме включено только перенаправление пакетов с внутренних интерфейсов. Перенаправление пакетов с внешних интерфейсов всегда выключено. Все внутренние интерфейсы открыты для любых входящих соединений.

7.3.2 Список блокируемых хостов

Модуль «Список блокируемых хостов» (пакет alterator-net-iptables) позволяет настроить блокировку любого сетевого трафика с указанных в списке узлов (входящий, исходящий и пересылаемый).

Блокирование трафика с указанных в списке узлов начинается после установки флажка «Использовать чёрный список» (Рис. 253).

Модуль «Список блокируемых хостов»

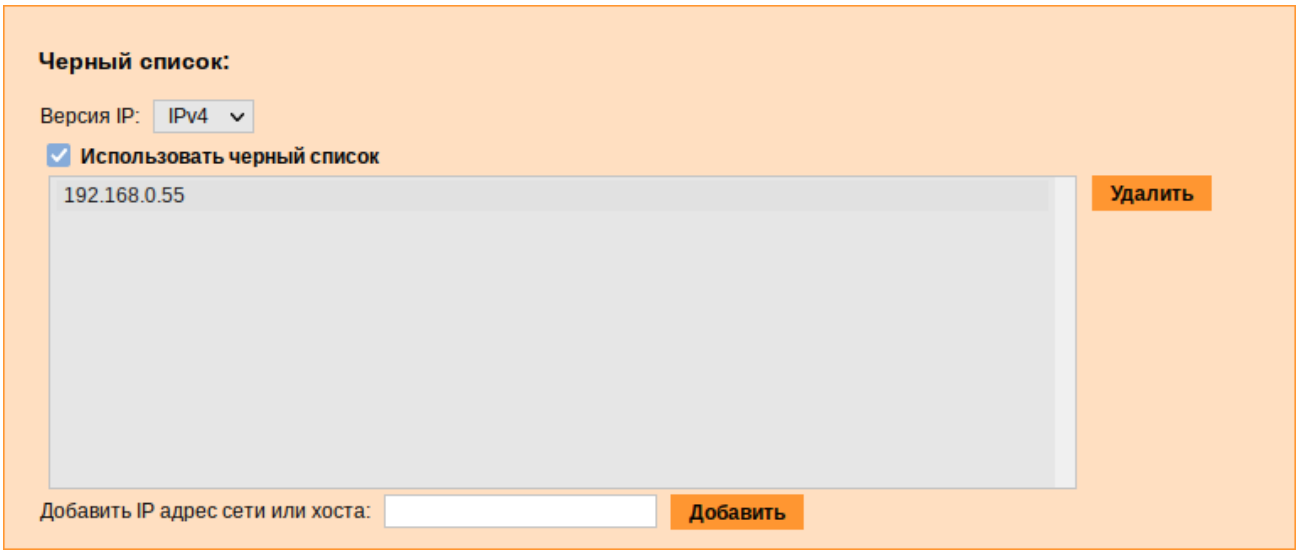


Рис. 253

Для добавления блокируемого узла необходимо ввести IP-адрес в поле «Добавить IP адрес сети или хоста» и нажать кнопку «Добавить».

Для удаления узла необходимо выбрать его из списка и нажать кнопку «Удалить».

7.4 Обслуживание сервера

Регулярный мониторинг состояния системы, своевременное резервное копирование, обновление установленного ПО, являются важной частью комплекса работ по обслуживанию сервера.

7.4.1 Мониторинг состояния системы

Для обеспечения бесперебойной работы сервера крайне важно производить постоянный мониторинг его состояния. Все события, происходящие с сервером, записываются в журналы, анализ которых помогает избежать сбоев в работе сервера и предоставляет возможность разобраться в причинах некорректной работы сервера.

Для просмотра журналов предназначен модуль ЦУС «Системные журналы» (пакет alterator-logs) из раздела «Система». Интерфейс позволяет просмотреть различные типы журналов с возможностью перехода к более старым или более новым записям.

Различные журналы могут быть выбраны из списка «Журналы» (Рис. 254).

Доступны следующие виды журналов:

– «Брандмауэр» – отображаются события безопасности, связанные с работой межсетевого экрана ОС;
– «Системные сообщения» – сообщения от системных служб (сообщения с типом DAEMON).

Модуль «Системные журналы»

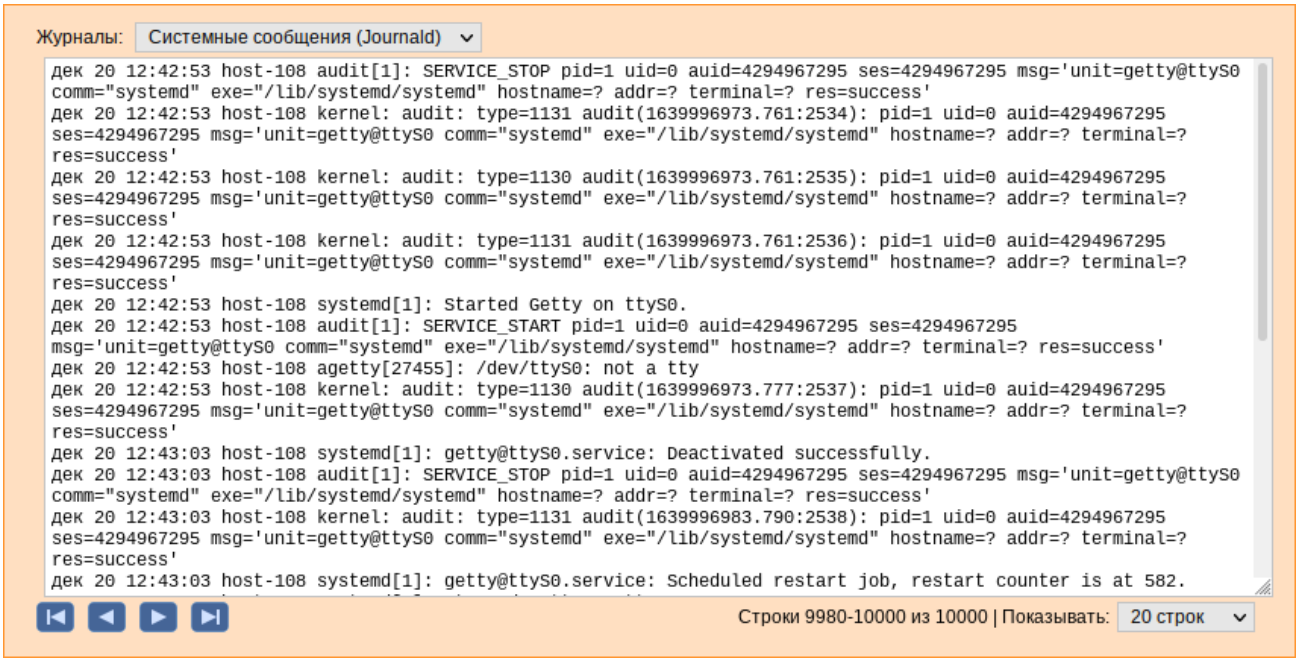


Рис. 254

Каждый журнал может содержать довольно большое количество сообщений. Уменьшить либо увеличить количество выводимых строк можно, выбрав нужное значение в списке «Показывать».

7.4.2 Системные службы

Для изменения состояния служб можно использовать модуль ЦУС «Системные службы» (пакет alterator-services) из раздела «Система». Интерфейс позволяет изменять текущее состояние службы и, если необходимо, применить опцию запуска службы при загрузке системы (Рис. 255).

Модуль «Системные службы»

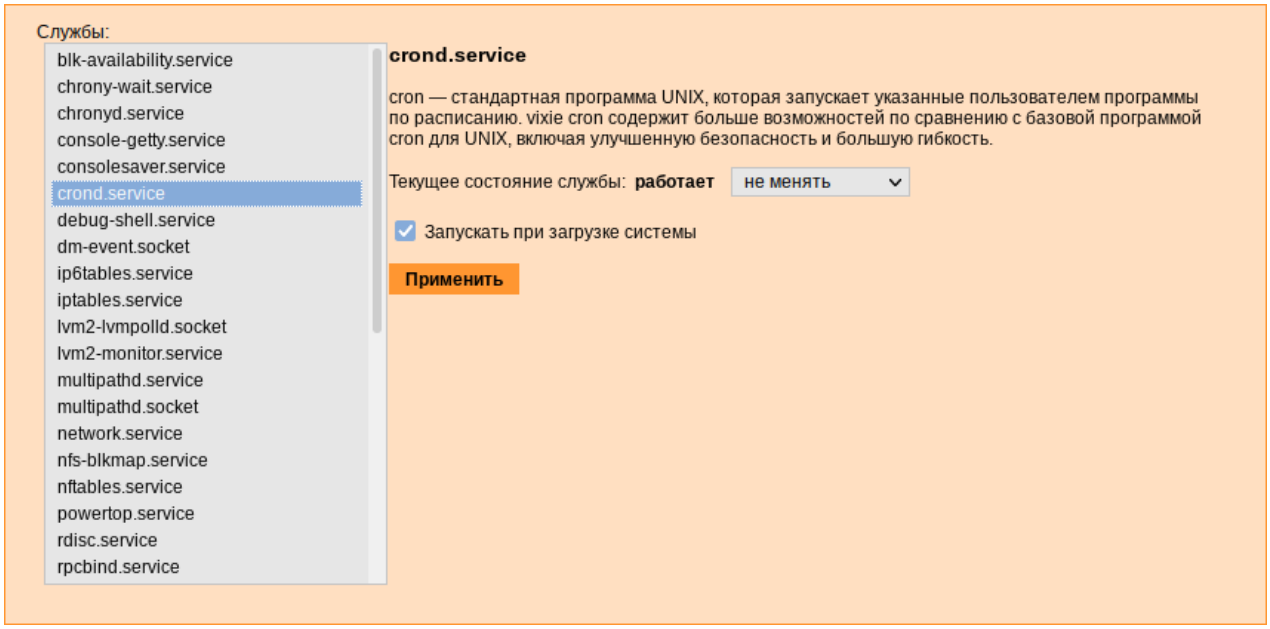


Рис. 255

После выбора названия службы из списка отображается описание данной службы, а также текущее состояние: «Работает»/«Остановлена»/«Неизвестно».

7.4.3 Обновление системы

После установки системы крайне важно следить за обновлениями ПО. Обновления для ОС «Альт Сервер Виртуализации» могут содержать как исправления, связанные с безопасностью, так и новый функционал или просто улучшение и ускорение алгоритмов. В любом случае настоятельно рекомендуется регулярно обновлять систему для повышения надёжности работы сервера.

Для автоматизации процесса установки обновлений предусмотрен модуль ЦУС «Обновление системы» (пакет alterator-updates) из раздела «Система». Здесь можно включить автоматическое обновление через Интернет с одного из предлагаемых серверов или задать собственные настройки (Рис. 256).

Источник обновлений указывается явно (при выбранном режиме «Обновлять систему автоматически из сети Интернет») или вычисляется автоматически (при выбранном режиме «Обновление системы управляемое сервером» и наличии в локальной сети настроенного сервера обновлений).

Процесс обновления системы будет запускаться автоматически согласно заданному расписанию.

Модуль «Обновление системы»

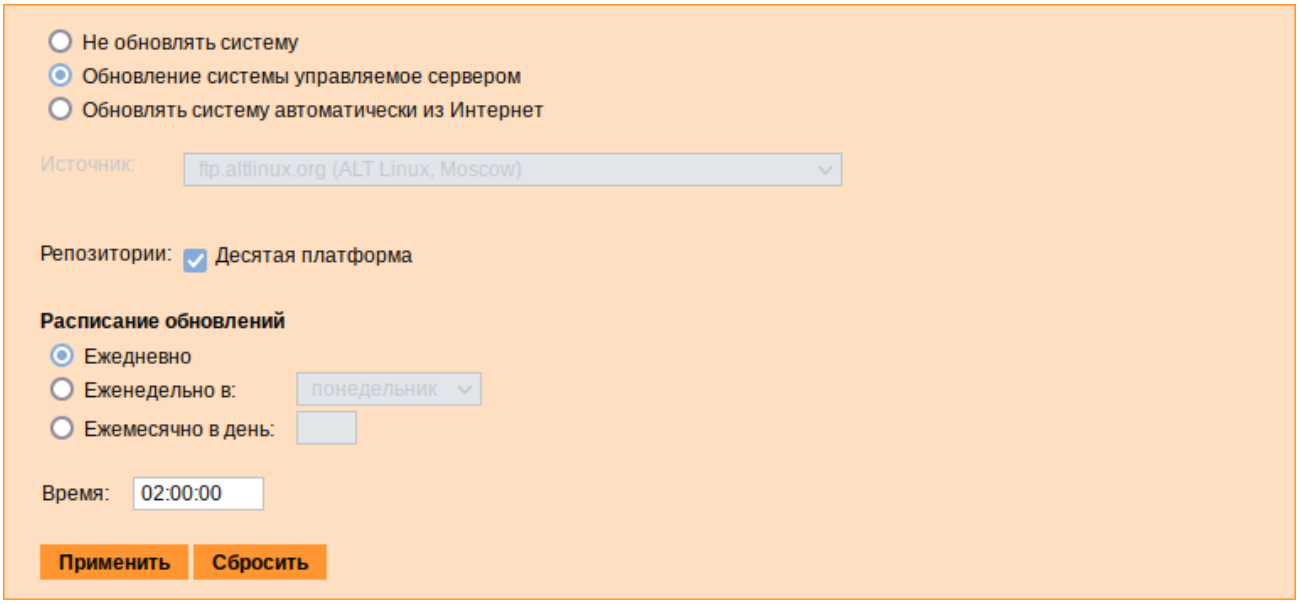


Рис. 256

7.4.4 Консоль

Модуль «Консоль» (пакет alterator-console) предназначен для запуска произвольных команд (Рис. 257).

Модуль «Консоль»

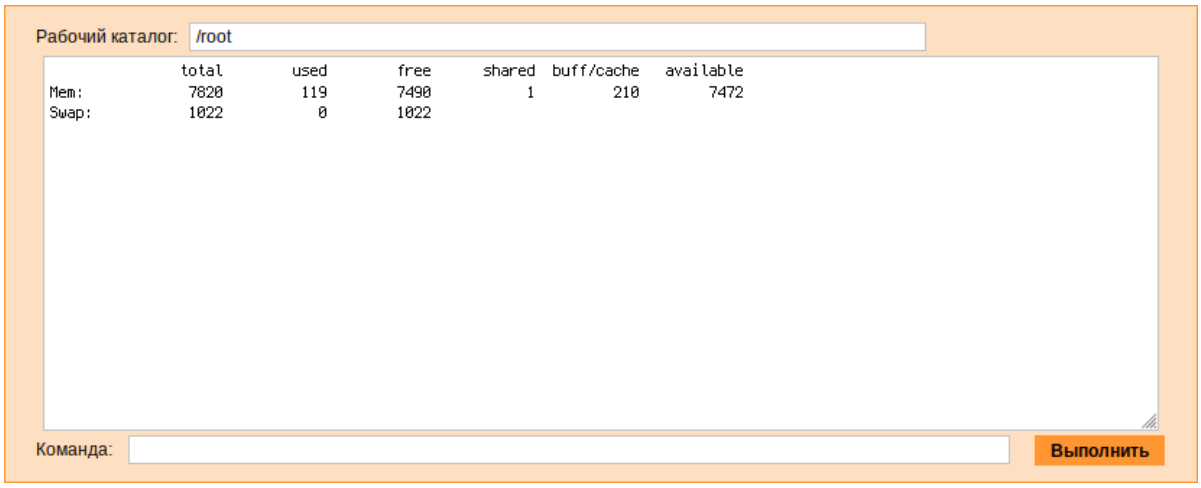


Рис. 257

Для запуска команды необходимо ввести команду в поле «Команда» и нажать кнопку «Выполнить». В рабочем поле будет выведен результат выполнения команды. Команда будет выполнена в указанном рабочем каталоге (по умолчанию `/root`).

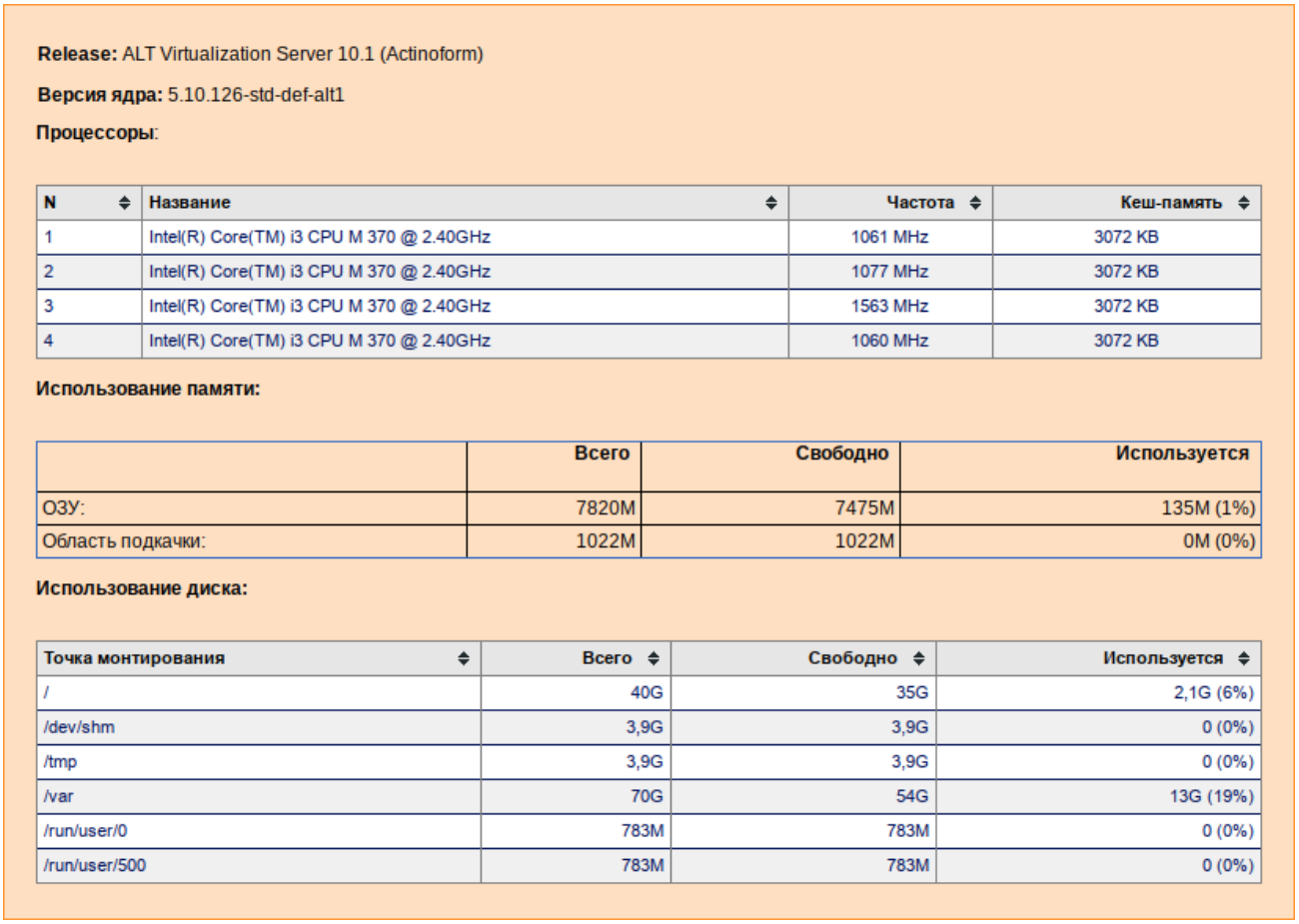
7.4.5 Информация о системе

Модуль «Информация о системе» (пакет alterator-sysinfo) предназначен для отображения информации о системе (Рис. 258):

– версии загруженного ядра;

– информации о процессорах;
– использование памяти;
– использование дискового пространства.

Модуль «Информация о системе»



Release: ALT Virtualization Server 10.1 (Actinoform)

Версия ядра: 5.10.126-std-def-alt1

Процессоры:

N	Название	Частота	Кеш-память
1	Intel(R) Core(TM) i3 CPU M 370 @ 2.40GHz	1061 MHz	3072 KB
2	Intel(R) Core(TM) i3 CPU M 370 @ 2.40GHz	1077 MHz	3072 KB
3	Intel(R) Core(TM) i3 CPU M 370 @ 2.40GHz	1563 MHz	3072 KB
4	Intel(R) Core(TM) i3 CPU M 370 @ 2.40GHz	1060 MHz	3072 KB

Использование памяти:

	Всего	Свободно	Используется
ОЗУ:	7820M	7475M	135M (1%)
Область подкачки:	1022M	1022M	0M (0%)

Использование диска:

Точка монтирования	Всего	Свободно	Используется
/	40G	35G	2,1G (6%)
/dev/shm	3,9G	3,9G	0 (0%)
/tmp	3,9G	3,9G	0 (0%)
/var	70G	54G	13G (19%)
/run/user/0	783M	783M	0 (0%)
/run/user/500	783M	783M	0 (0%)

Рис. 258

7.4.6 Веб-интерфейс

Модуль «Веб-интерфейс» предназначен для управления настройками веб-сервера, обеспечивающего работоспособность ЦУС. В поле «Порт» (Рис. 259) указывается номер TCP-порта, на котором сервер принимает соединения (порт по умолчанию 8080), в поле «Адрес» указывается IP-адрес сетевого интерфейса, на котором будет доступен ЦУС, в списке «Протоколирование» можно выбрать степень подробности протоколирования.

Модуль «Веб-интерфейс»

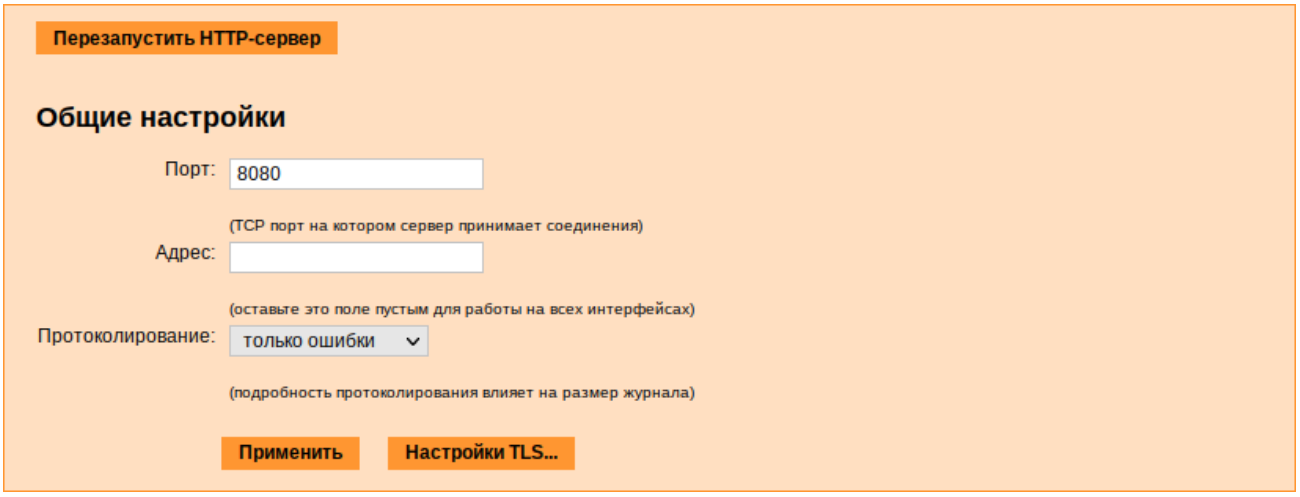


Рис. 259

7.4.7 Локальные учётные записи

Модуль «Локальные учётные записи» (пакет alterator-users) из раздела «Пользователи» предназначен для администрирования системных пользователей (Рис. 260).

Веб-интерфейс модуля alterator-users

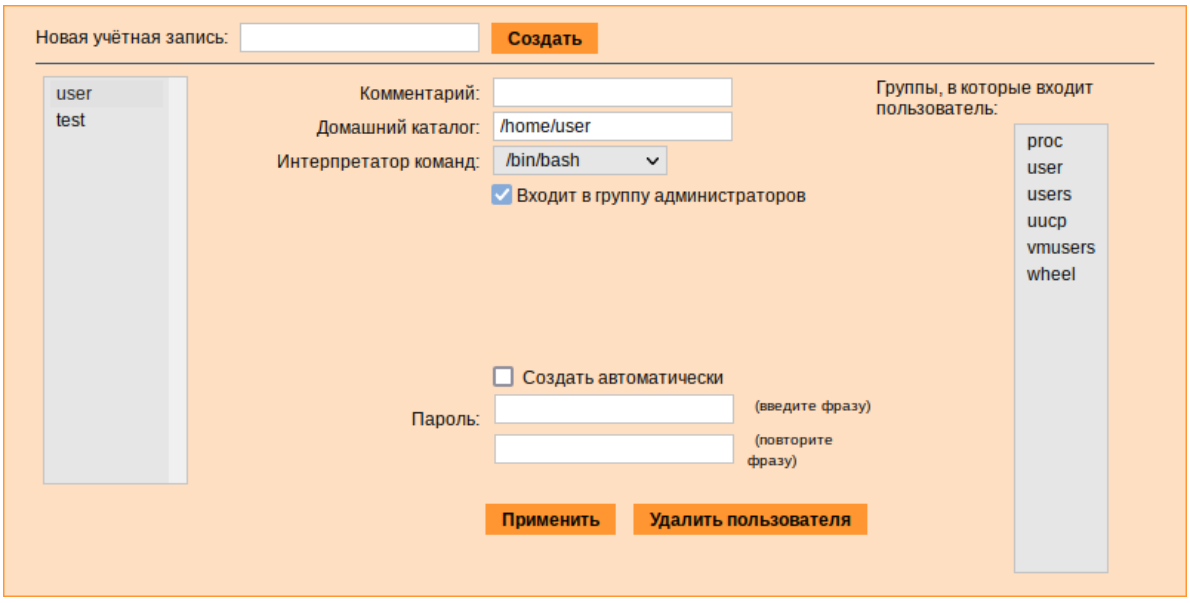


Рис. 260

Для создания новой учётной записи необходимо ввести имя новой учётной записи и нажать кнопку «Создать», после чего имя отобразится в списке слева.

Для дополнительных настроек необходимо выделить добавленное имя, либо, если необходимо изменить существующую учётную запись, выбрать её из списка.

7.4.8 Администратор системы

В модуле «Администратор системы» (пакет alterator-root) из раздела «Пользователи» можно изменить пароль суперпользователя (root), заданный при начальной настройке системы (Рис. 261).

Модуль «Администратор системы»

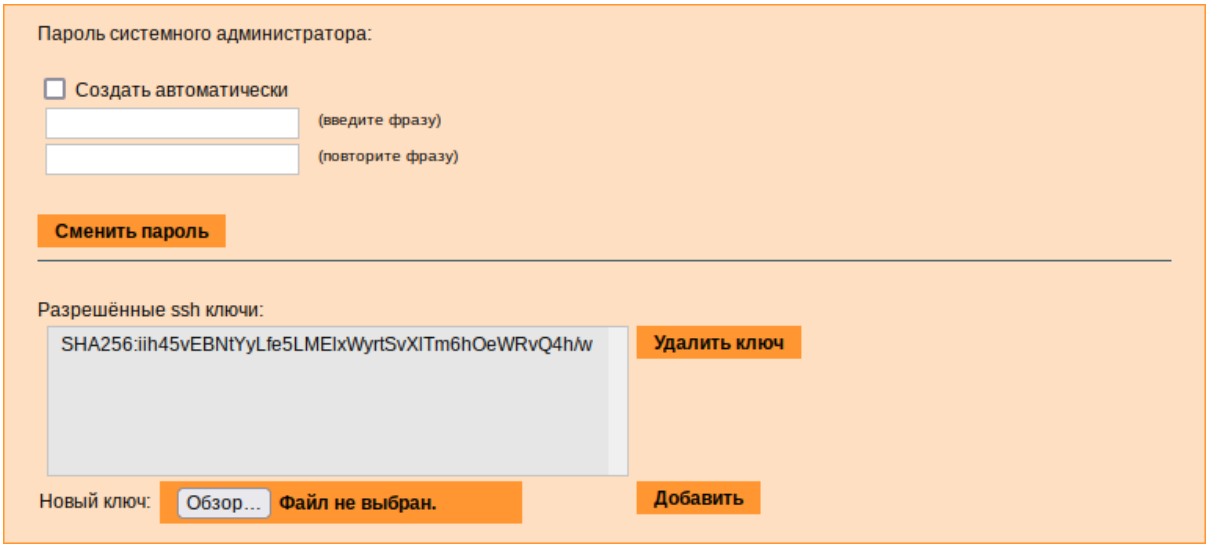


Рис. 261

В данном модуле можно добавить публичную часть ключа RSA или DSA для доступа к серверу по протоколу SSH.

7.4.9 Дата и время

В модуле «Дата и время» (пакет alterator-datetime) из раздела «Система» можно изменить дату и время на сервере, сменить часовой пояс, а также настроить автоматическую синхронизацию часов на самом сервере по протоколу NTP и предоставление точного времени по этому протоколу для рабочих станций локальной сети (Рис. 262).

Модуль «Дата и время»

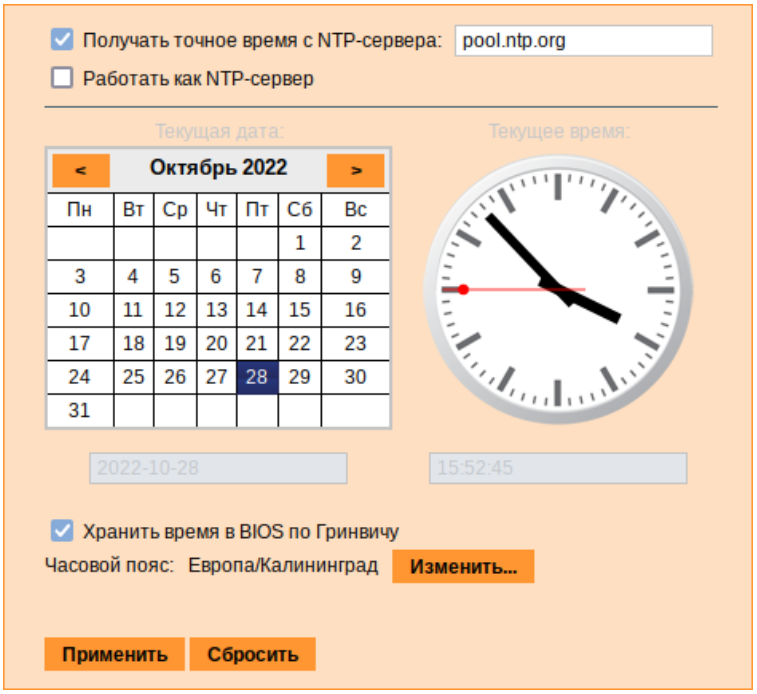


Рис. 262

Системное время зависит от следующих факторов:

– часы в BIOS – часы, встроенные в компьютер; они работают, даже если он выключен;
– системное время – часы в ядре операционной системы. Во время работы системы все процессы пользуются именно этими часами;
– часовые пояса – регионы Земли, в каждом из которых принято единое местное время.

При запуске системы происходит активация системных часов и их синхронизация с аппаратными, кроме того, в определённых случаях учитывается значение часового пояса. При завершении работы системы происходит обратный процесс.

Если настроена синхронизация времени с NTP-сервером, то сервер сможет сам работать как сервер точного времени. Для этого достаточно отметить соответствующий пункт «Работать как NTP-сервер».

7.4.10 Ограничение использования диска

Модуль «Использование диска» (пакет alterator-quota) из раздела «Пользователи» позволяет ограничить использование дискового пространства пользователями, заведёнными на сервере в модуле «Пользователи».

Модуль позволяет задать ограничения (квоты) для пользователя при использовании определённого раздела диска. Ограничить можно как суммарное количество килобайт, занятых файлами пользователя, так и количество этих файлов (Рис. 263).

Модуль «Использование диска»

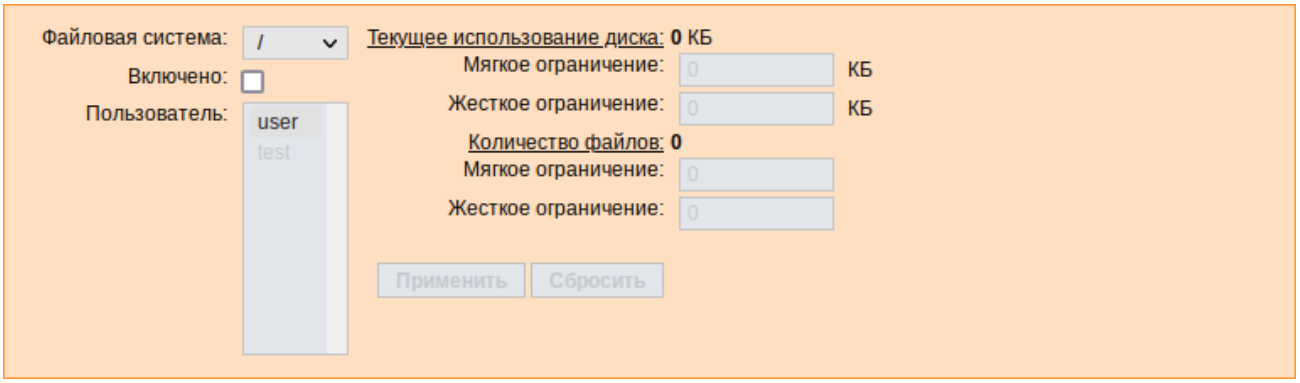


Рис. 263

Для управления квотами файловая система должна быть подключена с параметрами usrquota, grpquota. Для этого следует выбрать нужный раздел в списке «Файловая система» и установить отметку в поле «Включено» (Рис. 264).

Для того чтобы задать ограничения для пользователя, необходимо выбрать пользователя в списке «Пользователь», установить ограничения и нажать кнопку «Применить».

Модуль «Использование диска»

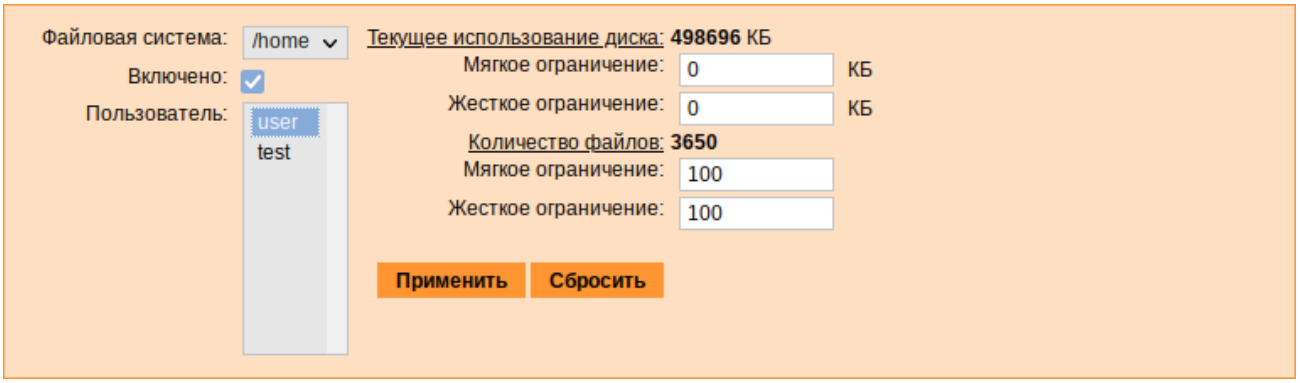


Рис. 264

При задании ограничений различают жёсткие и мягкие ограничения:

– мягкое ограничение: нижняя граница ограничения, которая может быть временно превышена. Временное ограничение – одна неделя;
– жёсткое ограничение: использование диска, которое не может быть превышено ни при каких условиях.

Значение 0 при задании ограничений означает отсутствие ограничений.

7.4.11 Выключение и перезагрузка компьютера

Иногда, в целях обслуживания или по организационным причинам необходимо корректно выключить или перезагрузить сервер. Для этого можно воспользоваться модулем ЦУС «Выключение компьютера» в разделе «Система».

Модуль ЦУС «Выключение компьютера» позволяет:

– выключить компьютер;
– перезагрузить компьютер;
– приостановить работу компьютера;
– погрузить компьютер в сон.

Возможна настройка ежедневного применения данных действий в заданное время.

Так как выключение и перезагрузка – критичные для функционирования компьютера операции, то по умолчанию настройка выставлена в значение «Продолжить работу» (Рис. 265). Для выключения, перезагрузки или перехода в энергосберегающие режимы нужно отметить соответствующий пункт и нажать «Применить».

Для ежедневного автоматического выключения компьютера, перезагрузки, а также перехода в энергосберегающие режимы необходимо отметить соответствующий пункт и задать желаемое время. Например, для выключения компьютера следует отметить пункт «Выключать компьютер каждый день в», задать время выключения в поле ввода слева от этого флажка и нажать кнопку «Применить».

Модуль «Выключение компьютера»

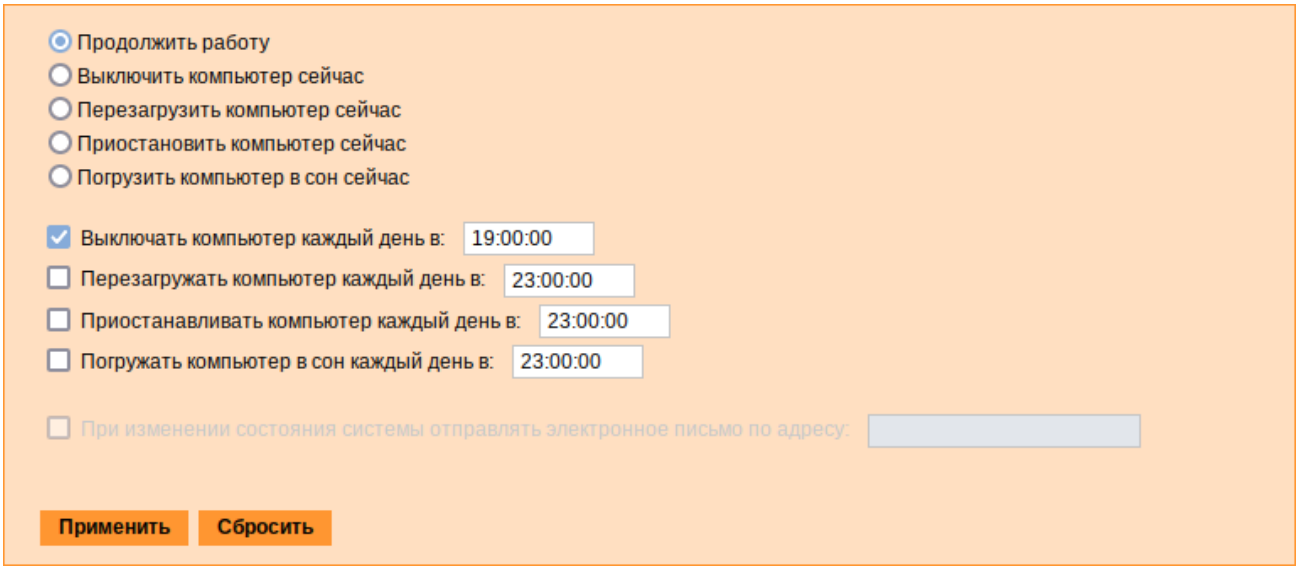


Рис. 265

Примечание. Для возможности настройки оповещений на e-mail, должен быть установлен пакет state-change-notify-postfix:

```
# apt-get install state-change-notify-postfix
```

Для настройки оповещений необходимо отметить пункт «При изменении состояния системы отправлять электронное письмо по адресу», ввести e-mail адрес и нажать кнопку «Применить» (Рис. 266).

Модуль «Выключение компьютера». Настройка оповещений

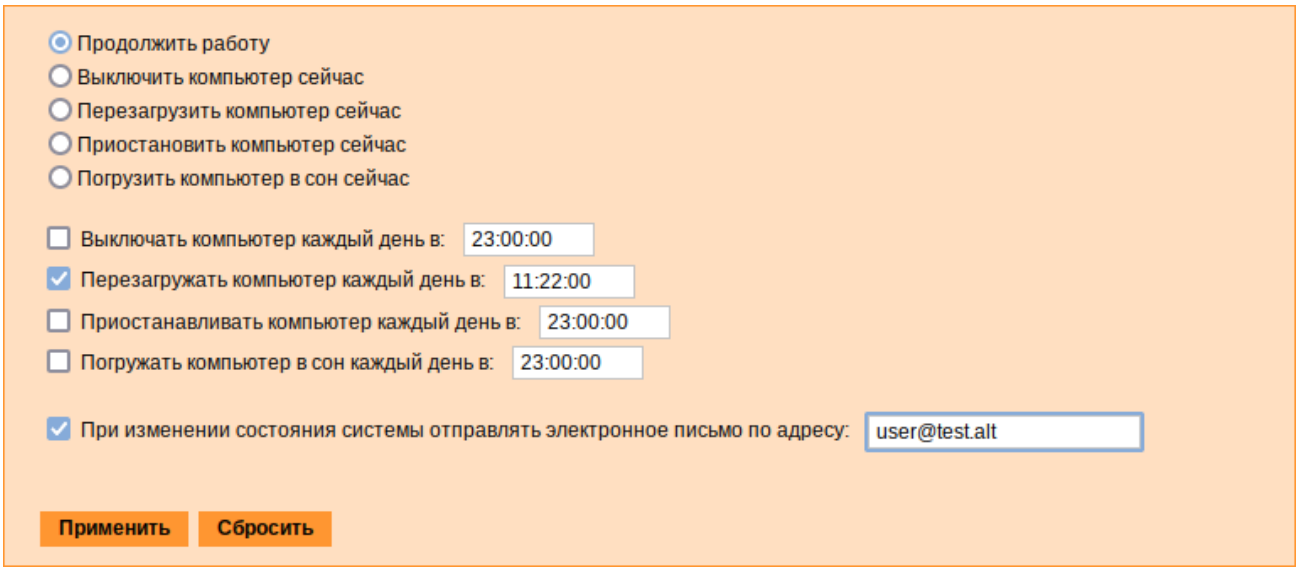


Рис. 266

По указанному адресу, при изменении состоянии системы будут приходить электронные письма. Например, при включении компьютера, содержание письма будет следующее:

```
Tue Jun 16 11:46:59 EET 2022: The server.test.alt is about to start.
```

При выключении:

```
Tue Jun 16 12:27:02 EET 2022: The server.test.alt is about to
shutdown.
```

Кнопка «Сбросить» возвращает сделанный выбор к безопасному значению по умолчанию: «Продолжить работу», перечитывает расписания и выставляет отметки для ежедневного автоматического действия в соответствии с прочитанным.

7.5 Прочие возможности ЦУС

Возможности ЦУС ОС «Альт Сервер Виртуализации» не ограничиваются только теми, что были описаны выше.

Установленные пакеты, которые относятся к ЦУС, можно посмотреть, выполнив команду:

```
rpm -qa | grep alterator*
```

Прочие пакеты для ЦУС можно найти, выполнив команду:

```
apt-cache search alterator*
```

Модули можно дополнительно загружать и удалять как обычные программы:

```
# apt-get install alterator-net-openvpn
# apt-get remove alterator-net-openvpn
```

7.6 Права доступа к модулям ЦУС

Администратор системы (root) имеет доступ ко всем модулям, установленным в системе, и может назначать права доступа для пользователей к определенным модулям.

Для разрешения доступа пользователю к конкретному модулю, администратору в веб-интерфейсе ЦУС необходимо выбрать нужный модуль и нажать ссылку «Параметры доступа к модулю», расположенную в нижней части окна модуля (Рис. 267).

Ссылка «Параметры доступа к модулю»

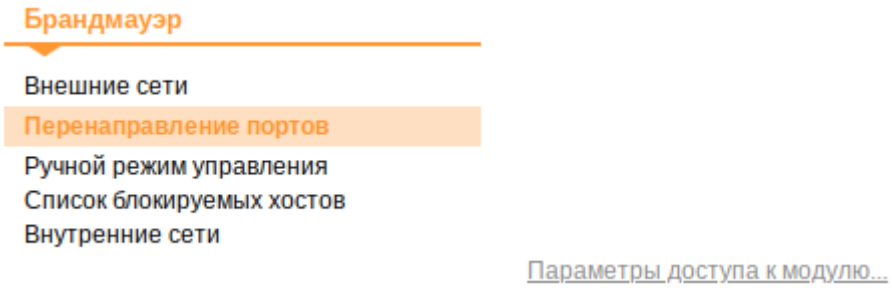


Рис. 267

В открывшемся окне, в списке «Новый пользователь» необходимо выбрать пользователя, который получит доступ к данному модулю, и нажать кнопку «Добавить» (Рис. 268). Для сохранения настроек необходимо перезапустить HTTP-сервер, для этого достаточно нажать кнопку «Перезапустить HTTP-сервер».

Параметры доступа к модулю

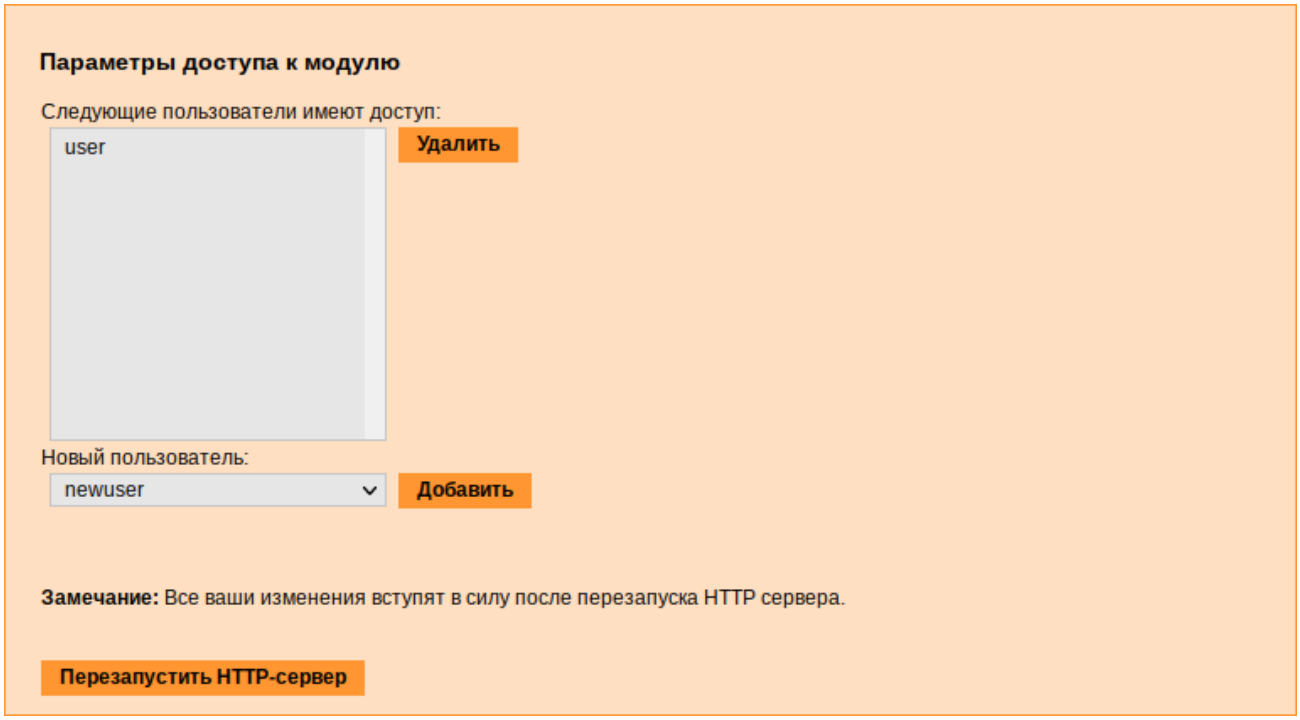


Рис. 268

Для удаления доступа пользователя к определенному модулю, администратору, в окне этого модуля необходимо нажать ссылку «Параметры доступа к модулю», в открывшемся окне в списке пользователей которым разрешен доступ, должен выбрать пользователя, нажать кнопку «Удалить» (Рис. 268) и перезапустить HTTP-сервер.

Системный пользователь, пройдя процедуру аутентификации, может просматривать и вызывать модули, к которым он имеет доступ.

8 УСТАНОВКА ДОПОЛНИТЕЛЬНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

После установки ОС «Альт Сервер Виртуализации», при первом запуске, доступен тот или иной набор программного обеспечения. Количество предустановленных программ зависит от выбора, сделанного при установке системы. Имеется возможность доустановить программы, которых не хватает в системе, из разных источников.

Дополнительное программное обеспечение может находиться на установочном диске и/или в специальных банках программ (репозиториях), расположенных в сети Интернет и/или в локальной сети. Программы, размещённые в указанных источниках, имеют вид подготовленных для установки пакетов.

Для установки, удаления и обновления программ и поддержания целостности системы в ОС семейства Linux используются менеджеры пакетов типа «rpm». Для автоматизации этого процесса и применяется Усовершенствованная система управления программными пакетами APT (Advanced Packaging Tool).

Автоматизация достигается созданием одного или нескольких внешних репозиториев, в которых хранятся пакеты программ и относительно которых производится сверка пакетов, установленных в системе. Репозитории могут содержать как официальную версию дистрибутива, обновляемую его разработчиками по мере выхода новых версий программ, так и локальные наработки, например, пакеты, разработанные внутри компании.

Таким образом, в распоряжении APT находятся две базы данных: одна описывает установленные в системе пакеты, вторая – внешний репозиторий. APT отслеживает целостность установленной системы и, в случае обнаружения противоречий в зависимостях пакетов, руководствуется сведениями о внешнем репозитории для разрешения конфликтов и поиска корректного пути их устранения.

Система APT состоит из нескольких утилит. Чаще всего используется утилита управления пакетами apt-get, которая автоматически определяет зависимости между пакетами и строго следит за их соблюдением при выполнении любой из следующих операций: установка, удаление или обновление пакетов.

8.1 Источники программ (репозитории)

Репозитории, с которыми работает APT, отличаются от обычного набора пакетов наличием мета информации – индексов пакетов, содержащихся в репозитории, и сведений о них. Поэтому, чтобы получить всю информацию о репозитории, APT достаточно получить его индексы.

APT может работать с любым количеством репозиториев одновременно, формируя единую

информационную базу обо всех содержащихся в них пакетах. При установке пакетов APT обращает внимание только на название пакета, его версию и зависимости, а расположение в том или ином репозитории не имеет значения. Если потребуется, APT в рамках одной операции установки группы пакетов может пользоваться несколькими репозиториями.

Подключая одновременно несколько репозиториев, нужно следить за тем, чтобы они были совместимы друг с другом по пакетной базе – отражали один определенный этап разработки. Совместимыми являются основной репозиторий дистрибутива и репозиторий обновлений по безопасности к данному дистрибутиву. В то же время смешение среди источников APT репозиториев, относящихся к разным дистрибутивам, или смешение стабильного репозитория с нестабильной веткой разработки (Sisyphus) чревато различными неожиданными трудностями при обновлении пакетов.

APT позволяет взаимодействовать с репозиторием с помощью различных протоколов доступа. Наиболее популярные – HTTP и FTP, однако существуют и некоторые дополнительные методы.

Для того чтобы APT мог использовать тот или иной репозиторий, информацию о нем необходимо поместить в файл `/etc/apt/sources.list`, либо в любой файл `.list` (например, `mysources.list`) в каталоге `/etc/apt/sources.list.d/`. Описания репозиториев заносятся в эти файлы в следующем виде:

```
rpm [подпись] метод: путь база название
rpm-src [подпись] метод: путь база название
```

где:

- rpm или rpm-src – тип репозитория (скомпилированные программы или исходные тексты);
- [подпись] – необязательная строка-указатель на электронную подпись разработчиков. Наличие этого поля подразумевает, что каждый пакет из данного репозитория должен быть подписан соответствующей электронной подписью. Подписи описываются в файле `/etc/apt/vendor.list`;
- метод – способ доступа к репозиторию: ftp, http, file, cdrom, copy;
- путь – путь к репозиторию в терминах выбранного метода;
- база – относительный путь к базе данных репозитория;
- название – название репозитория.

При выборе пакетов для установки APT руководствуется всеми доступными репозиториями вне зависимости от способа доступа к ним. Таким образом, если в репозитории, доступном по сети Интернет, обнаружена более новая версия программы, чем на CD (DVD)-носителе информации, APT начнет загружать данный пакет по сети.

8.1.1 Добавление репозиториев

Непосредственно после установки дистрибутива «Альт Сервер Виртуализации» в `/etc/apt/sources.list`, а также в файлах `/etc/apt/sources.list.d/*.list` обычно указывается несколько репозиториев:

- репозиторий с установочного диска дистрибутива;
- интернет-репозиторий, совместимый с установленным дистрибутивом.

8.1.1.1 Утилита apt-repo для работы с репозиториями

Для добавления репозиториев можно воспользоваться утилитой `apt-repo`.

Примечание. Для выполнения большинства команд необходимы права администратора.

Просмотреть список активных репозиториев можно, выполнив команду:

```
$ apt-repo list
```

Команда добавления репозитория в список активных репозиториев:

```
apt-repo add <репозиторий>
```

Команда удаления или выключения репозитория:

```
apt-repo rm <репозиторий>
```

Команда удаления всех репозиториев:

```
apt-repo clean
```

Обновление информации о репозиториях:

```
apt-repo update
```

Вывод справки:

```
man apt-repo
```

или

```
apt-repo –help
```

Типичный пример использования: удалить все источники и добавить стандартный репозиторий p10 (архитектура выбирается автоматически):

```
# apt-repo rm all
# apt-repo add p10
```

Или то же самое одной командой:

```
# apt-repo set p10
```

8.1.1.2 Добавление репозитория на CD/DVD-носителе

Для добавления в `sources.list` репозитория на CD/DVD-носителе информации в APT предусмотрена специальная утилита – `apt-cdrom`. Чтобы добавить запись о репозитории на носителе, достаточно вставить его в привод для чтения (записи) CD (DVD)-носителей информации и выполнить следующую команду:

```
# apt-cdrom add
```

После этого в `sources.list` появится запись о подключенном диске примерно такого вида:

```
rpm cdrom:[ALT Server v x86_64]/ ALTLinux main
```

Примечание. В случае если записи для cdrom в файле `/etc/fstab` нет, потребуется примонтировать носитель информации вручную (каталог `/media/ALTLinux` должен существовать):

```
# mount /dev/cdrom /media/ALTLinux
```

Затем использовать команду добавления носителя с дополнительным ключом:

```
# apt-cdrom add -m
```

8.1.1.3 Добавление репозиториев вручную

Для редактирования списка репозиториев можно отредактировать в любом текстовом редакторе файлы из папки `/etc/apt/sources.list.d/`. Для изменения этих файлов необходимы права администратора. В файле `alt.list` может содержаться такая информация:

```
rpm [alt] http://ftp.altlinux.org/pub/distributions/ALTLinux
p10/x86_64 classic
rpm [alt] http://ftp.altlinux.org/pub/distributions/ALTLinux
p10/x86_64-i586 classic
rpm [alt] http://ftp.altlinux.org/pub/distributions/ALTLinux
p10/noarch classic
```

По сути, каждая строчка соответствует некому репозиторию. Для выключения репозитория достаточно закомментировать соответствующую строку (дописать символ решётки перед строкой). Для добавления нового репозитория необходимо дописать его вниз этого или любого другого файла.

8.1.2 Обновление информации о репозиториях

Практически любое действие с системой apt начинается с обновления данных от активированных источников. Список источников необходимо обновлять при поиске новой версии пакета, установке пакетов или обновлении установленных пакетов новыми версиями.

Обновление данных осуществляется командой:

```
# apt-get update
```

После выполнения этой команды, apt обновит свой кэш новой информацией.

8.2 Поиск пакетов

Утилита `apt-cache` предназначена для поиска программных пакетов, в репозитории, и позволяет искать не только по имени пакета, но и по его описанию.

Команда `apt-cache search <подстрока>` позволяет найти все пакеты, в именах или описании которых присутствует указанная подстрока. Пример поиска может выглядеть следующим образом:

```
$ apt-cache search ^telegraf
ceph-mgr-telegraf - Telegraf module for Ceph Manager Daemon
telegraf - The plugin-driven server agent for collecting and reporting metrics
```

Символ «^» в поисковом выражении, указывает на то, что необходимо найти совпадения только в начале строки (в данном случае – в начале имени пакета).

Для того чтобы подробнее узнать о каждом из найденных пакетов и прочитать его описание, можно воспользоваться командой `apt-cache show`, которая покажет информацию о пакете из репозитория:

```
$ apt-cache show telegraf
Package: telegraf
Section: Development/Other
Installed Size: 132855876
Maintainer: Alexey Shabalin (ALT Team) <shaba@altlinux.org>
Version: 1.19.2-alt1:p10+281572.100.1.1@1627678454
Pre-Depends: /bin/sh, /usr/sbin/groupadd, /usr/sbin/useradd, /usr/sbin/usermod, /usr/sbin/post_service, /usr/sbin/preun_service, rpmlib(PayloadIsXz)
Depends: /bin/kill, /bin/sh, /etc/logrotate.d, /etc/rc.d/init.d, /etc/rc.d/init.d(SourceIfNotEmpty), /etc/rc.d/init.d(msg_reloading), /etc/rc.d/init.d(msg_usage), /etc/rc.d/init.d(start_daemon), /etc/rc.d/init.d(status), /etc/rc.d/init.d(stop_daemon), /etc/rc.d/init.d/functions
Provides: telegraf (= 1.19.2-alt1:p10+281572.100.1.1)
Architecture: x86_64
Size: 22968033
MD5Sum: d9d6ecaba627d86436ddfdffc243f2cd
Filename: telegraf-1.19.2-alt1.x86_64.rpm
Description: The plugin-driven server agent for collecting and reporting metrics
 Telegraf is an agent written in Go for collecting, processing, aggregating, and writing metrics.
```

```
 Design goals are to have a minimal memory footprint with a plugin
system so that developers
 in the community can easily add support for collecting metrics from
well known services
 (like Hadoop, Postgres, or Redis) and third party APIs (like
Mailchimp, AWS CloudWatch,
 or Google Analytics).
```

При поиске с помощью `apt-cache` можно использовать русскую подстроку. В этом случае будут найдены пакеты, имеющие описание на русском языке.

8.3 Установка или обновление пакета

Установка пакета с помощью APT выполняется командой:

```
# apt-get install <имя_пакета>
```

Примечание. Перед установкой и обновлением пакетов необходимо выполнить команду обновления индексов пакетов:

```
# apt-get update
```

Если пакет уже установлен и в подключенном репозитории нет обновлений для данного пакета, система сообщит об уже установленном пакете последней версии. Если в репозитории присутствует более новая версия или новое обновление – программа начнет процесс установки.

`apt-get` позволяет устанавливать в систему пакеты, требующие для работы другие, пока еще не установленные. В этом случае он определяет, какие пакеты необходимо установить, и устанавливает их, пользуясь всеми доступными репозиториями.

Установка пакета `telegraf` командой `apt-get install telegraf` приведет к следующему диалогу с APT (если пакет еще не установлен):

```
# apt-get install telegraf
Чтение списков пакетов... Завершено
Построение дерева зависимостей... Завершено
Следующие НОВЫЕ пакеты будут установлены:
  telegraf
0 будет обновлено, 1 новых установлено, 0 пакетов будет удалено и 0 не
будет обновлено.
Необходимо получить 23,0MB архивов.
После распаковки потребуется дополнительно 133MB дискового
пространства.
```

```
Получено: 1 http://ftp.altlinux.org p10/branch/x86_64/classic telegraf
1.19.2-alt1:p10+281572.100.1.1@1627678454 [23,0MB]
Получено 23,0MB за 7s (2942kB/s).
Совершаем изменения...
Подготовка...
##################################################################
##### [100%]
Обновление / установка...
1:                                                  telegraf-1.19.2-alt1
##################################################################
##### [100%]
Завершено.
```

Команда `apt-get install <имя_пакета>` используется и для обновления уже установленного пакета или группы пакетов. В этом случае `apt-get` дополнительно проверяет, не обновилась ли версия пакета в репозитории по сравнению с установленным в системе.

Например, если пакет gimp установлен и в репозитории нет обновлённой версии этого пакета, то вывод команды `apt-get install telegraf` будет таким:

```
# apt-get install telegraf
Чтение списков пакетов... Завершено
Построение дерева зависимостей... Завершено
Последняя версия telegraf уже установлена.
0 будет обновлено, 0 новых установлено, 0 пакетов будет удалено и 2262
не будет обновлено.
```

При помощи APT можно установить и отдельный бинарный rpm-пакет, не входящий ни в один из репозиториев. Для этого достаточно выполнить команду `apt-get install путь_к_файлу.rpm`. При этом APT проведет стандартную процедуру проверки зависимостей и конфликтов с уже установленными пакетами.

В результате операций с пакетами без использования APT может нарушиться целостность ОС «Альт Сервер Виртуализации», и `apt-get` в таком случае откажется выполнять операции установки, удаления или обновления.

Для восстановления целостности ОС «Альт Сервер Виртуализации» необходимо повторить операцию, задав опцию `-f`, заставляющую `apt-get` исправить нарушенные зависимости, удалить или заменить конфликтующие пакеты. Любые действия в этом режиме обязательно требуют подтверждения со стороны пользователя.

При установке пакетов происходит запись в системный журнал вида:

```
apt-get: имя-пакета installed
```

8.4 Удаление установленного пакета

Для удаления пакета используется команда `apt-get remove <имя_пакета>`. Удаление пакета с сохранением его файлов настройки производится при помощи следующей команды:

```
# apt-get remove <значимая_часть_имени_пакета>
```

В случае, если при этом необходимо полностью очистить систему от всех компонент удаляемого пакета, то применяется команда:

```
# apt-get remove --purge <значимая_часть_имени_пакета>
```

Для того чтобы не нарушать целостность системы, будут удалены и все пакеты, зависящие от удаляемого.

В случае удаления с помощью `apt-get` базового компонента системы появится запрос на подтверждение операции:

```
# apt-get remove filesystem
Обработка файловых зависимостей... Завершено
Чтение списков пакетов... Завершено
Построение дерева зависимостей... Завершено
Следующие пакеты будут УДАЛЕНЫ:
basesystem filesystem ppp sudo
Внимание: следующие базовые пакеты будут удалены:
В обычных условиях этого не должно было произойти, надеемся, вы точно
представляете, чего требуете!
basesystem filesystem (по причине basesystem)
0 пакетов будет обновлено, 0 будет добавлено новых, 4 будет
удалено(заменено) и 0 не будет обновлено.
Необходимо получить 0B архивов. После распаковки 588kБ будет
освобождено.
Вы делаете нечто потенциально опасное!
Введите фразу 'Yes, do as I say!' чтобы продолжить.
```

Каждую ситуацию, в которой APT выдает такое сообщение, необходимо рассматривать отдельно. Однако, вероятность того, что после выполнения этой команды система окажется неработоспособной, очень велика.

При удалении пакетов происходит запись в системный журнал вида:

```
apt-get: имя-пакета removed
```

8.5 Обновление всех установленных пакетов

Полное обновление всех установленных в системе пакетов производится при помощи команд:

```
# apt-get update && apt-get dist-upgrade
```

Первая команда (`apt-get update`) обновит индексы пакетов. Вторая команда (`apt-get dist-upgrade`) позволяет обновить только те установленные пакеты, для которых в репозиториях, перечисленных в `/etc/apt/sources.list`, имеются новые версии.

В случае обновления всего дистрибутива APT проведёт сравнение системы с репозиторием и удалит устаревшие пакеты, установит новые версии присутствующих в системе пакетов, отследит ситуации с переименованиями пакетов или изменения зависимостей между старыми и новыми версиями программ. Всё, что потребуется поставить (или удалить) дополнительно к уже имеющемуся в системе, будет указано в отчете apt-get, которым APT предварит само обновление.

Примечание. Команда `apt-get dist-upgrade` обновит систему, но ядро ОС не будет обновлено.

8.6 Обновление ядра

Для обновления ядра ОС необходимо выполнить команду:

```
# update-kernel
```

Примечание. Если индексы сегодня еще не обновлялись перед выполнением команды update-kernel необходимо выполнить команду `apt-get update`.

Команда `update-kernel` обновляет и модули ядра, если в репозитории обновилось что-то из модулей без обновления ядра.

Новое ядро загрузится только после перезагрузки системы.

9 КОРПОРАТИВНАЯ ИНФРАСТРУКТУРА

9.1 Zabbix

Zabbix – система мониторинга и отслеживания статусов разнообразных сервисов компьютерной сети, серверов и сетевого оборудования.

Для управления системой мониторинга и чтения данных используется веб-интерфейс.

Перед установкой должен быть установлен и запущен сервер PostgreSQL, с созданным пользователем zabbix и созданной базой zabbix.

9.1.1 Установка клиента Zabbix

Установить необходимый пакет:

```
# apt-get install zabbix-agent
```

Добавить Zabbix agent в автозапуск и запустить его:

```
# systemctl enable --now zabbix_agentd
```

Адрес сервера, которому разрешено обращаться к агенту задается в конфигурационном файле `/etc/zabbix/zabbix_agentd.conf` параметрами:

```
Server=127.0.0.1
ServerActive=127.0.0.1
```

10 ОБЩИЕ ПРИНЦИПЫ РАБОТЫ ОС

Работа с операционной средой заключается во вводе определенных команд (запросов) к операционной среде и получению на них ответов в виде текстового отображения.

Основой операционной среды является операционная система.

Операционная система (ОС) – совокупность программных средств, организующих согласованную работу операционной среды с аппаратными устройствами компьютера (процессор, память, устройства ввода-вывода и т. д.).

Диалог с ОС осуществляется посредством командных интерпретаторов и системных библиотек.

Каждая системная библиотека представляет собой набор программ, динамически вызываемых операционной системой.

Командные интерпретаторы – особый род специализированных программ, позволяющих осуществлять диалог с ОС посредством команд.

Для удобства пользователей при работе с командными интерпретаторами используются интерактивные рабочие среды (далее – ИРС), предоставляющие пользователю удобный интерфейс для работы с ОС.

В самом центре ОС изделия находится управляющая программа, называемая ядром. В ОС изделия используется новейшая модификация «устойчивого» ядра Linux – версия 5.4.

Ядро взаимодействует с компьютером и периферией (дисками, принтерами и т. д.), распределяет ресурсы и выполняет фоновое планирование заданий.

Другими словами, ядро ОС изолирует вас от сложностей аппаратуры компьютера, командный интерпретатор от ядра, а ИРС от командного интерпретатора.

ОС «Альт Сервер Виртуализации» является многопользовательской интегрированной системой. Это значит, что она разработана в расчете на одновременную работу нескольких пользователей.

Пользователь может либо сам работать в системе, выполняя некоторую последовательность команд, либо от его имени могут выполняться прикладные процессы.

Пользователь взаимодействует с системой через командный интерпретатор, который представляет собой, как было сказано выше, прикладную программу, которая принимает от пользователя команды или набор команд и транслирует их в системные вызовы к ядру системы. Интерпретатор позволяет пользователю просматривать файлы, передвигаться по дереву файловой системы, запускать прикладные процессы. Все командные интерпретаторы UNIX имеют развитый командный язык и позволяют писать достаточно сложные программы, упрощающие процесс администрирования системы и работы с ней.

10.1 Процессы функционирования ОС

Все программы, которые выполняются в текущий момент времени, называются процессами. Процессы можно разделить на два основных класса: системные процессы и пользовательские процессы. Системные процессы – программы, решающие внутренние задачи ОС, например, организацию виртуальной памяти на диске или предоставляющие пользователям те или иные сервисы (процессы-службы).

Пользовательские процессы – процессы, запускаемые пользователем из командного интерпретатора для решения задач пользователя или управления системными процессами. Linux изначально разрабатывался как многозадачная система. Он использует технологии, опробованные и отработанные другими реализациями UNIX, которые существовали ранее.

Фоновый режим работы процесса – режим, когда программа может работать без взаимодействия с пользователем. В случае необходимости интерактивной работы с пользователем (в общем случае) процесс будет «остановлен» ядром, и работа его продолжится только после переведения его в «нормальный» режим работы.

10.2 Файловая система ОС

В ОС использована файловая система Linux, которая в отличие от файловых систем DOS и Windows(™) является единым деревом. Корень этого дерева – каталог, называемый root (рут), и обозначаемый «/». Части дерева файловой системы могут физически располагаться в разных разделах разных дисков или вообще на других компьютерах, – для пользователя это прозрачно. Процесс присоединения файловой системы раздела к дереву называется монтированием, удаление – размонтированием. Например, файловая система CD-ROM в изделии монтируется по умолчанию в каталог /media/cdrom (путь в изделии обозначается с использованием «/», а не «\», как в DOS/Windows). Текущий каталог обозначается «./».

Файловая система изделия содержит каталоги первого уровня:

- /bin (командные оболочки (shell), основные утилиты);
- /boot (содержит ядро системы);
- /dev (псевдофайлы устройств, позволяющие работать с ними напрямую);
- /etc (файлы конфигурации);
- /home (личные каталоги пользователей);
- /lib (системные библиотеки, модули ядра);
- /lib64 (64-битные системные библиотеки);
- /media (каталоги для монтирования файловых систем сменных устройств);

- /mnt (каталоги для монтирования файловых систем сменных устройств и внешних файловых систем);
- /proc (файловая система на виртуальном устройстве, ее файлы содержат информацию о текущем состоянии системы);
- /root (личный каталог администратора системы);
- /sbin (системные утилиты);
- /sys (файловая система, содержащая информацию о текущем состоянии системы);
- /usr (программы и библиотеки, доступные пользователю);
- /var (рабочие файлы программ, очереди, журналы);
- /tmp (временные файлы).

10.3 Организация файловой структуры

Система домашних каталогов пользователей помогает организовывать безопасную работу пользователей в многопользовательской системе. Вне своего домашнего каталога пользователь обладает минимальными правами (обычно чтение и выполнение файлов) и не может нанести ущерб системе, например, удалив или изменив файл.

Кроме файлов, созданных пользователем, в его домашнем каталоге обычно содержатся персональные конфигурационные файлы некоторых программ.

Маршрут (путь) – это последовательность имён каталогов, представляющий собой путь в файловой системе к данному файлу, где каждое следующее имя отделяется от предыдущего наклонной чертой (слэшем). Если название маршрута начинается со слэша, то путь в искомый файл начинается от корневого каталога всего дерева системы. В обратном случае, если название маршрута начинается непосредственно с имени файла, то путь к искомому файлу должен начаться от текущего каталога (рабочего каталога).

Имя файла может содержать любые символы за исключением косой черты (/). Однако следует избегать применения в именах файлов большинства знаков препинания и непечатаемых символов. При выборе имен файлов рекомендуется ограничиться следующими символами:

- строчные и ПРОПИСНЫЕ буквы. Следует обратить внимание на то, что регистр всегда имеет значение;
- цифры;
- символ подчеркивания (_);
- точка (.).

Для удобства работы можно использовать точку (.) для отделения имени файла от расширения файла. Данная возможность может быть необходима пользователям или некоторым программам, но не имеет значение для shell.

10.3.1 Иерархическая организация файловой системы

Каталог /:

/boot – место, где хранятся файлы необходимые для загрузки ядра системы;

/lib – здесь располагаются файлы динамических библиотек, необходимых для работы большей части приложений и подгружаемые модули ядра;

/lib64 – здесь располагаются файлы 64-битных динамических библиотек, необходимых для работы большей части приложений;

/bin – минимальный набор программ необходимых для работы в системе;

/sbin – набор программ для административной работы с системой (программы необходимые только суперпользователю);

/home – здесь располагаются домашние каталоги пользователей;

/etc – в данном каталоге обычно хранятся общесистемные конфигурационные файлы для большинства программ в системе;

/etc/rc?.d,/etc/init.d,/etc/rc.boot,/etc/rc.d – директории, где расположены командные файлы системы инициализации SysVinit;

/etc/passwd – база данных пользователей, в которой содержится информация об имени пользователя, его настоящем имени, личном каталоге, закодированный пароль и другие данные;

/etc/shadow – теневая база данных пользователей. При этом информация из файла /etc/passwd перемещается в /etc/shadow, который недоступен по чтению всем, кроме пользователя root. В случае использования альтернативной схемы управления теневыми паролями (TCB) все теневые пароли для каждого пользователя располагаются в директории /etc/tcb/<имя пользователя>/shadow;

/dev – в этом каталоге находятся файлы устройств. Файлы в /dev создаются сервисом udev;

/usr – обычно файловая система /usr достаточно большая по объему, так как все программы установлены именно здесь. Вся информация в каталоге /usr помещается туда во время установки системы. Отдельно устанавливаемые пакеты программ и другие файлы размещаются в каталоге /usr/local. Некоторые подкаталоги системы /usr рассмотрены ниже;

/usr/bin – практически все команды, хотя некоторые находятся в /bin или в /usr/local/bin;

/usr/sbin – команды, используемые при администрировании системы и не предназначенные для размещения в файловой системе root;

/usr/local – здесь рекомендуется размещать файлы, установленные без использования пакетных менеджеров, внутренняя организация каталогов практически такая же, как и корневого каталога;

/usr/man – каталог, где хранятся файлы справочного руководства man;

/usr/share – каталог для размещения общедоступных файлов большей части приложений.

Каталог /var:

/var/log – место, где хранятся файлы аудита работы системы и приложений;

/var/spool – каталог для хранения файлов находящих в очереди на обработку для того или иного процесса (очередь на печать, отправку почты и т. д.);

/tmp – временный каталог необходимый некоторым приложениям;

/proc – файловая система /proc является виртуальной и в действительности она не существует на диске. Ядро создает её в памяти компьютера. Система /proc предоставляет информацию о системе.

10.3.2 Имена дисков и разделов

Все физические устройства вашего компьютера отображаются в каталог /dev файловой системы изделия (об этом – ниже). Диски (в том числе IDE/SATA/SCSI жёсткие диски, USB-диски) имеют имена:

/dev/sda – первый диск;

/dev/sdb – второй диск;

и т. д.

Диски обозначаются /dev/sdX, где X – a,b,c,d,e,... в порядке обнаружения системой.

Раздел диска обозначается числом после его имени. Например, /dev/sdb4 – четвертый раздел второго диска.

10.4 Разделы, необходимые для работы ОС

Для работы ОС необходимо создать на жестком диске (дисках) по крайней мере, два раздела: корневой (то есть тот, который будет содержать каталог /) и раздел подкачки (swap). Размер последнего, как правило, составляет от однократной до двукратной величины оперативной памяти компьютера. Если свободного места на диске много, то можно создать отдельные разделы для каталогов /usr, /home, /var.

10.5 Управление системными сервисами и командами

10.5.1 Сервисы

Сервисы – это программы, которые запускаются и останавливаются через инициализированные скрипты, расположенные в каталоге /etc/init.d. Многие из этих сервисов запускаются на этапе старта ОС «Альт Сервер Виртуализации». В ОС существует шесть системных уровней выполнения, каждый из которых определяет список служб (сервисов), запускаемых на данном уровне. Уровни 0 и 6 соответствуют выключению и перезагрузке системы.

При загрузке системы процесс init запускает все сервисы, указанные в каталоге /etc/rc (0-6).d/ для уровня по умолчанию. Поменять его можно в конфигурационном файле /etc/inittab. Следующая строка соответствует второму уровню выполнения:

```
id:2:initdefault:
```

Для тестирования изменений, внесенных в файл inittab, применяется команда telinit. При указании аргумента -q процесс init повторно читает inittab.

Для перехода ОС «Альт Сервер Виртуализации» на нужный уровень выполнения можно воспользоваться командой init, например:

```
init 3
```

Данная команда переведет систему на третий уровень выполнения, запустив все сервисы, указанные в каталоге /etc/rc3.d/.

10.5.2 Команды

Далее приведены основные команды, использующиеся в ОС «Альт Сервер Виртуализации»:

- ar – создание и работа с библиотечными архивами;
- at – формирование или удаление отложенного задания;
- awk – язык обработки строковых шаблонов;
- batch – планирование команд в очереди загрузки;
- bc – строковый калькулятор;
- chfn – управление информацией учетной записи (имя, описание);
- chsh – управление выбором командного интерпретатора (по умолчанию – для учетной записи);
- cut – разбивка файла на секции, задаваемые контекстными разделителями;
- df – вывод отчета об использовании дискового пространства;
- dmesg – вывод содержимого системного буфера сообщений;
- du – вычисление количества использованного пространства элементов ФС;
- echo – вывод содержимого аргументов на стандартный вывод;
- egrep – поиск в файлах содержимого согласно регулярным выражениям;
- fgrep – поиск в файлах содержимого согласно фиксированным шаблонам;
- file – определение типа файла;
- find – поиск файла по различным признакам в иерархии каталогов;
- gettext – получение строки интернационализации из каталогов перевода;
- grep – вывод строки, содержащей шаблон поиска;
- groupadd – создание новой учетной записи группы;

- groupdel – удаление учетной записи группы;
- groupmod – изменение учетной записи группы;
- groups – вывод списка групп;
- gunzip – распаковка файла;
- gzip – упаковка файла;
- hostname – вывод и задание имени хоста;
- install – копирование файла с установкой атрибутов;
- ipcrm – удаление ресурса IPC;
- ipcs – вывод характеристик ресурса IPC;
- kill – прекращение выполнения процесса;
- killall – удаление процессов по имени;
- lpr – система печати;
- ls – вывод содержимого каталога;
- lsb_release – вывод информации о дистрибутиве;
- m4 – запуск макропроцессора;
- md5sum – генерация и проверка MD5-сообщения;
- mknod – создание файла специального типа;
- mktemp – генерация уникального имени файла;
- more – постраничный вывод содержимого файла;
- mount – монтирование ФС;
- msgfmt – создание объектного файла сообщений из файла сообщений;
- newgrp – смена идентификатора группы;
- nice – изменение приоритета процесса перед его запуском;
- nohup – работа процесса после выхода из системы;
- od – вывод содержимого файла в восьмеричном и других видах;
- passwd – смена пароля учетной записи;
- patch – применение файла описания изменений к оригинальному файлу;
- pidof – вывод идентификатора процесса по его имени;
- ps – вывод информации о процессах;
- renice – изменение уровня приоритета процесса;
- sed – строковый редактор;
- sendmail – транспорт системы электронных сообщений;
- sh – командный интерпретатор;

- shutdown – команда останова системы;
- su – изменение идентификатора запускаемого процесса;
- sync – сброс системных буферов на носители;
- tar – файловый архиватор;
- umount – размонтирование ФС;
- useradd – создание новой учетной записи или обновление существующей;
- userdel – удаление учетной записи и соответствующих файлов окружения;
- usermod – модификация информации об учетной записи;
- w – список пользователей, кто в настоящий момент работает в системе и с какими файлами;
- who – вывод списка пользователей системы.

Узнать об опциях команд можно с помощью команды man.

11 РАБОТА С НАИБОЛЕЕ ЧАСТО ИСПОЛЬЗУЕМЫМИ КОМПОНЕНТАМИ

11.1 Командные оболочки (интерпретаторы)

Для управления ОС используется командные интерпретаторы (shell).

Зайдя в систему, можно увидеть приглашение – строку, содержащую символ «$» (далее, этот символ будет обозначать командную строку). Программа ожидает ввода команд. Роль командного интерпретатора – передавать команды пользователя операционной системе. При помощи командных интерпретаторов можно писать небольшие программы – сценарии (скрипты). В Linux доступны следующие командные оболочки:

bash – самая распространённая оболочка под linux. Она ведет историю команд и предоставляет возможность их редактирования.

pdksh – клон korn shell, хорошо известной оболочки в UNIX(™) системах.

Оболочкой по умолчанию является «Bash» (Bourne Again Shell) Проверить, какая оболочка используется, можно выполнив команду:

```
$ echo $SHELL
```

У каждой оболочки свой синтаксис. Все примеры в дальнейшем построены с использованием оболочки Bash.

11.1.1 Командная оболочка Bash

В Bash имеется несколько приемов для работы со строкой команд. Например, используя клавиатуру, можно:

<Ctrl> + <A> – перейти на начало строки;

<Ctrl> + <U> – вырезать/удалить все символы слева от курсора до начала строки в буфер обмена;

<Ctrl> + <C> – остановить текущую задачу.

Для ввода нескольких команд одной строкой можно использовать разделитель «;». По истории команд можно перемещаться с помощью клавиш <↑> и <↓>. Чтобы найти конкретную команду в списке набранных, не пролистывая всю историю, необходимо набрать <Ctrl> + <R> и начать вводить символы ранее введенной команды.

Для просмотра истории команд можно воспользоваться командой history. Команды, присутствующие в истории, отображаются в списке пронумерованными. Чтобы запустить конкретную команду необходимо набрать:

```
!номер команды
```

Если ввести:

```
!!
```

запустится последняя, из набранных команд.

В Bash имеется возможность самостоятельного завершения имен команд из общего списка команд, что облегчает работу при вводе команд, в случае, если имена программ и команд слишком длинны. При нажатии клавиши <Tab> Bash завершает имя команды, программы или каталога, если не существует нескольких альтернативных вариантов. Например, чтобы использовать программу декомпрессии gunzip, можно набрать следующую команду:

```
$ gu
```

Затем нажать <Tab>. Так как в данном случае существует несколько возможных вариантов завершения команды, то необходимо повторно нажать клавишу <Tab>, чтобы получить список имен, начинающихся с gu.

В предложенном примере можно получить следующий список:

```
$ gu
guile gunzip gupnp-binding-tool
```

Если набрать: n (gunzip – это единственное имя, третьей буквой которого является «n»), а затем нажать клавишу <Tab>, то оболочка самостоятельно дополнит имя. Чтобы запустить команду нужно нажать <Enter>.

Программы, вызываемые из командной строки, Bash ищет в каталогах, определяемых в системной переменной PATH. По умолчанию в этот перечень каталогов не входит текущий каталог, обозначаемый ./ (точка слеш) (если только не выбран один из двух самых слабых уровней защиты). Поэтому, для запуска программы из текущего каталога, необходимо использовать команду (в примере запускается команда prog):

```
./prog
```

11.1.2 Базовые команды оболочки Bash

Все команды, приведенные ниже, могут быть запущены в режиме консоли. Для получения более подробной информации следует использовать команду `man`. Пример:

```
$ man ls
```

11.1.2.1 Учетные записи пользователей

Команда su

Команда su позволяет получить права администратора. При вводе команды su, будет запрошен пароль суперпользователя (root). И в случае ввода корректного пароля, оператор получит привилегии суперпользователя. Чтобы вернуться к правам оператора, необходимо ввести команду:

```
# exit
```

Команда id

Команда id выводит информацию о пользователе и группах, в которых он состоит для заданного пользователя или о текущем пользователе (если ничего не указано).

Синтаксис:

```
id [параметры] [ПОЛЬЗОВАТЕЛЬ]
```

Команда passwd

Команда passwd меняет (или устанавливает) пароль, связанный с входным_именем пользователя.

Обычный пользователь может менять только пароль, связанный с его собственным входным_именем.

Команда запрашивает у обычных пользователей старый пароль (если он был), а затем дважды запрашивает новый. Новый пароль должен соответствовать техническим требованиям к паролям, заданным администратором системы.

11.1.2.2 Основные операции с файлами и каталогами

Команда ls

Команда ls (list) выдает список файлов каталога.

Синтаксис:

```
ls [-CFRacdilqrtu1] [[-H] | [-L]] [-fgmnopsx] [файл...]
```

Основные опции:

-a – просмотр всех файлов, включая скрытые;

-l – отображение более подробной информации;

-R – выводить рекурсивно информацию о подкаталогах.

Команда cd

Команда cd предназначена для смены каталога. Команда работает как с абсолютными, так и с относительными путями. Если каталог не указан, используется значение переменной окружения HOME (домашний каталог пользователя). Если каталог задан полным маршрутным именем, он становится текущим. По отношению к новому каталогу нужно иметь право на выполнение, которое в данном случае трактуется как разрешение на поиск.

Синтаксис:

```
cd [-L|-P] [каталог]
```

Если в качестве аргумента задано -, то это эквивалентно $OLDPWD. Если переход был осуществлен по переменной окружения CDPATH или в качестве аргумента был задан - и смена каталога была успешной, то абсолютный путь нового рабочего каталога будет выведен на стандартный вывод.

Пример. Находясь в домашнем каталоге перейти в его подкаталог docs/ (относительный путь):

```
$ cd docs/
```

Сделать текущим каталог /usr/bin (абсолютный путь):

```
$ cd /usr/bin/
```

Сделать текущим родительский каталог:

```
$ cd ..
```

Вернуться в предыдущий каталог:

```
$ cd -
```

Сделать текущим домашний каталог:

```
$ cd
```

Команда pwd

Команда pwd выводит абсолютный путь текущего (рабочего) каталога.

Синтаксис:

```
pwd [-L|-P]
```

Опции:

-P – не выводить символические ссылки;

-L – выводить символические ссылки.

Команда rm

Команда rm используется для удаления файлов.

Синтаксис:

```
rm [-fiRr] имя_файла
```

Основные опции:

-f – не запрашивать подтверждения;

-i – запрашивать подтверждение;

-r, -R – рекурсивно удалять содержимое указанных каталогов.

Пример. Удалить все файлы html в каталоге ~/html:

```
$ rm -i ~/html/*.html
```

Команда mkdir

Команда mkdir позволяет создать каталог.

Синтаксис:

```
mkdir [-p] [-m права] [каталог...]
```

Команда rmdir

Команда rmdir удаляет записи, соответствующие указанным пустым каталогам.

Синтаксис:

```
rmdir [-p] [каталог...]
```

Команда rmdir часто заменяется командой rm -rf, которая позволяет удалять каталоги, даже если они не пусты.

Команда cp

Команда cp предназначена для копирования файлов.

Синтаксис:

```
cp [-fip] [исх_файл] [цел_файл]
cp [-fip] [исх_файл...] [каталог]
cp [-R] [[-H] | [-L] | [-P]] [-fip] [исх_файл...] [каталог]
```

Основные опции:

-p – сохранять по возможности времена изменения и доступа к файлу, владельца и группу, права доступа;

-i – запрашивать подтверждение перед копированием в существующие файлы;

-r, -R – рекурсивно копировать содержимое каталогов.

Команда mv

Команда mv предназначена для перемещения файлов.

Синтаксис:

```
mv [-fi] [исх_файл...] [цел_файл]
mv [-fi] [исх_файл...] [каталог]
```

В первой синтаксической форме, характеризующейся тем, что последний операнд не является ни каталогом, ни символической ссылкой на каталог, mv перемещает исх_файл в цел_файл.

Во второй синтаксической форме mv перемещает исходные файлы в указанный каталог под именами, совпадающими с краткими именами исходных файлов.

Основные опции:

-f – не запрашивать подтверждения перезаписи существующих файлов;

-i – запрашивать подтверждение перезаписи существующих файлов.

Команда cat

Команда cat последовательно выводит содержимое файлов.

Синтаксис:

```
cat [параметры] [файл...]
```

Основные опции:

-n, --number – нумеровать все строки при выводе;

-E, --show-ends – показывать $ в конце каждой строки.

Если файл не указан, читается стандартный ввод. Если в списке файлов присутствует имя -, вместо этого файла читается стандартный ввод.

Команда less

Команда less позволяет постранично просматривать текст (для выхода необходимо нажать <q>).

Синтаксис:

```
less имя_файла
```

Команда grep

Команда grep имеет много опций и предоставляет возможности поиска символьной строки в файле.

Синтаксис:

```
grep шаблон_поиска файл
```

Команда chmod

Команда chmod изменяет права доступа к файлу.

Синтаксис:

```
chmod ОПЦИЯ]... РЕЖИМ[,РЕЖИМ]... [Файл...]
chmod ОПЦИЯ]... --reference=ИФАЙЛ ФАЙЛ...
```

Основные опции:

-R – рекурсивно изменять режим доступа к файлам, расположенным в указанных каталогах;

--reference=ИФАЙЛ – использовать режим файла ИФАЙЛ.

Команда chmod изменяет права доступа каждого указанного файла в соответствии с правами доступа, указанными в параметре режим, который может быть представлен как в символьном виде, так и в виде восьмеричного, представляющего битовую маску новых прав доступа.

Формат символьного режима следующий:

```
[ugoa...][[+-=][разрешения...]...]
```

Здесь разрешения – это ноль или более букв из набора «rwxXst» или одна из букв из набора «ugo».

Каждый аргумент – это список символьных команд изменения прав доступа, разделены запятыми. Каждая такая команда начинается с нуля или более букв «ugoa», комбинация которых указывает, чьи права доступа к файлу будут изменены: пользователя, владеющего файлом (u), пользователей, входящих в группу, к которой принадлежит файл (g), остальных пользователей (o) или всех пользователей (a). Если не задана ни одна буква, то автоматически будет использована буква «a», но биты, установленные в umask, не будут затронуты.

Оператор «+» добавляет выбранные права доступа к уже имеющимся у каждого файла, «-» удаляет эти права, «=» присваивает только эти права каждому указанному файлу.

Буквы «rwxXst» задают биты доступа для пользователей: «r» – чтение, «w» – запись, «x» – выполнение (или поиск для каталогов), «X» – выполнение/поиск, только если это каталог или же файл с уже установленным битом выполнения, «s» – задать ID пользователя и группы при выполнении, «t» – запрет удаления.

Примеры. Позволить всем выполнять файл f2:

```
$ chmod +x f2
```

Запретить удаление файла f3:

```
$ chmod+t f3
```

Команда chown

Команда chown изменяет владельца и/или группу для каждого заданного файла.

Синтаксис:

```
chown [КЛЮЧ]…[ВЛАДЕЛЕЦ][:[ГРУППА]] ФАЙЛ ...
```

Изменить владельца может только владелец файла или суперпользователь. Владелец не изменяется, если он не задан в аргументе. Группа также не изменяется, если не задана, но если после символьного ВЛАДЕЛЬЦА стоит символ «:», подразумевается изменение группы на основную группу текущего пользователя. Поля ВЛАДЕЛЕЦ и ГРУППА могут быть как числовыми, так и символьными.

Примеры. Поменять владельца /u на пользователя test:

```
$ chown test /u
```

Поменять владельца и группу /u:

```
$ chown test:staff /u
```

Поменять владельца /u и вложенных файлов на test:

```
$ chown -hR test /u
```

11.1.2.3 Поиск файлов

Команда find

Команда find предназначена для поиска всех файлов, начиная с корневой директории. Поиск может осуществляться по имени, типу или владельцу файла.

Синтаксис:

```
find [-H] [-L] [-P] [-Оуровень] [-D
help|tree|search|stat|rates|opt|exec] [путь…] [выражение]
```

Ключи для поиска:

-name – поиск по имени файла;

-type – поиск по типу f=файл, d=каталог, l=ссылка(lnk);

-user – поиск по владельцу (имя или UID).

Когда выполняется команда find, можно выполнять различные действия над найденными файлами. Основные действия:

-exec команда \; – выполнить команду. Запись команды должна заканчиваться экранированной точкой с запятой. Строка «{}» заменяется текущим маршрутным именем файла;

-execdir команда \; – то же самое что и exec, но команда вызывается из подкаталога, содержащего текущий файл;

-ok команда – эквивалентно -exec за исключением того, что перед выполнением команды запрашивается подтверждение (в виде сгенерированной командной строки со знаком вопроса в конце) и она выполняется только при ответе: y;

-print – вывод имени файла на экран.

Путем по умолчанию является текущий подкаталог. Выражение по умолчанию -print.

Примеры. Найти в текущем каталоге обычные файлы (не каталоги), имя которых начинается с символа «~»:

```
$ find . -type f -name "~*" -print
```

Найти в текущем каталоге файлы, измененные позже, чем файл file.bak:

```
$ find . -newer file.bak -type f -print
```

Удалить все файлы с именами a.out или *.o, доступ к которым не производился в течение недели:

```
$ find / \( -name a.out -o -name '*.o' \) \ -atime +7 -exec rm {} \;
```

Удалить из текущего каталога и его подкаталогов все файлы нулевого размера, запрашивая подтверждение:

```
$ find . -size 0c -ok rm {} \;
```

Команда whereis

Команда whereis сообщает путь к исполняемому файлу программы, ее исходным файлам (если есть) и соответствующим страницам справочного руководства.

Синтаксис:

```
whereis [options] <name>
```

Опции:

-b – вывод информации только об исполняемых файлах;

-m – вывод информации только о страницах справочного руководства;

-s – вывод информации только об исходных файлах.

11.1.2.4 Мониторинг и управление процессами

Команда ps

Команда ps отображает список текущих процессов.

Синтаксис:

```
ps [-aA] [-defl] [-G список] [-o формат...] [-p список] [-t список] [-U список] [-g список] [-n список] [-u список]
```

По умолчанию выводится информация о процессах с теми же действующим UID и управляющим терминалом, что и у подающего команду пользователя.

Основные опции:

-a – вывести информацию о процессах, ассоциированных с терминалами;

-f – вывести «полный» список;

-l – вывести «длинный» список;

-p список – вывести информацию о процессах с перечисленными в списке PID;

-u список – вывести информацию о процессах с перечисленными идентификаторами или именами пользователей.

Команда kill

Команда kill позволяет прекратить исполнение процесса или передать ему сигнал.

Синтаксис:

```
kill [-s] [сигнал] [идентификатор] [...]
kill [-l] [статус_завершения]
kill [-номер_сигнала] [идентификатор] [...]
```

Идентификатор – PID ведущего процесса задания или номер задания, предварённый знаком «%».

Основные опции:

-l – вывести список поддерживаемых сигналов;

-s сигнал, -сигнал – послать сигнал с указанным именем.

Если обычная команда kill не дает желательного эффекта, необходимо использовать команду kill с параметром -9:

```
$ kill -9 PID_номер
```

Команда df

Команда df показывает количество доступного дискового пространства в файловой системе, в которой содержится файл, переданный как аргумент. Если ни один файл не указан, показывается доступное место на всех смонтированных файловых системах. Размеры по умолчанию указаны в блоках по 1КБ по умолчанию.

Синтаксис:

```
df [опция]... [файл]...
```

Основные опции:

-total – подсчитать общий объем в конце;

-h, --human-readable – печатать размеры в удобочитаемом формате (например, 1K 234M 2G);

-h, --human-readable – печатать размеры в удобочитаемом формате (например, 1K 234M 2G).

Команда du

Команда du подсчитывает использование диска каждым файлом, для каталогов подсчет происходит рекурсивно.

Синтаксис:

```
du [опции][файл...]
```

Основные опции:

-a, --all – выводить общую сумму для каждого заданного файла, а не только для каталогов;

-c, --total – подсчитать общий объем в конце. Может быть использовано для выяснения суммарного использования дискового пространства для всего списка заданных файлов;

-d, --max-depth=N – выводить объем для каталога (или файлов, если указано --all) только если она на N или менее уровней ниже аргументов командной строки;

-S, --separate-dirs – выдавать отдельно размер каждого каталога, не включая размеры подкаталогов;

-s, --summarize – отобразить только сумму для каждого аргумента.

Команда which

Команда which – отображает полный путь к указанным командам или сценариям.

Синтаксис:

```
which [опции] [--] имя_программы [...]
```

Основные опции:

-a, --all – выводит все совпавшие исполняемые файлы по содержимому в переменной окружения PATH, а не только первый из них;

-c, --total – подсчитать общий объем в конце. Может быть использовано для выяснения суммарного использования дискового пространства для всего списка заданных файлов;

-d, --max-depth=N – выводить объем для каталога (или файлов, если указано --all) только если она на N или менее уровней ниже аргументов командной строки;

-S, --separate-dirs – выдавать отдельно размер каждого каталога, не включая размеры подкаталогов;

--skip-dot – пропускает все каталоги из переменной окружения PATH, которые начинаются с точки.

11.1.2.5 Использование многозадачности

ОС «Альт Сервер Виртуализации» – многозадачная система.

Для того чтобы запустить программу в фоновом режиме необходимо набрать «&» после имени программы. После этого оболочка дает возможность запускать другие приложения.

Так как некоторые программы интерактивны – их запуск в фоновом режиме бессмысленен. Подобные программы просто остановятся, если их запустить в фоновом режиме.

Можно также запускать нескольких независимых сеансов. Для этого в консоли необходимо набрать <Alt> и одну из клавиш, находящихся в интервале от <F1> до <F6>. На экране появится новое приглашение системы, и можно открыть новый сеанс.

Команда bg

Команда bg используется для того, чтобы перевести задание на задний план.

Синтаксис:

```
bg [идентификатор ...]
```

Идентификатор – PID ведущего процесса задания или номер задания, предварённый знаком «%».

Команда fg

Команда fg позволяет перевести задание на передний план.

Синтаксис:

```
fg [идентификатор ...]
```

Идентификатор – PID ведущего процесса задания или номер задания, предварённый знаком «%».

11.1.2.6 Сжатие и упаковка файлов

Команда tar

Сжатие и упаковка файлов выполняется с помощью команды tar, которая преобразует файл или группу файлов в архив без сжатия (tarfile).

Упаковка файлов в архив чаще всего выполняется следующей командой:

```
$ tar -cf [имя создаваемого файла архива] [упаковываемые файлы и/или директории]
```

Пример использования команды упаковки архива:

```
$ tar -cf moi_dokumenti.tar Docs project.tex
```

Распаковка содержимого архива в текущий каталог выполняется командой:

```
$ tar -xf [имя файла архива]
```

Для сжатия файлов используются специальные программы сжатия: gzip, bzip2 и 7z.

11.2 Стыкование команд в системе

11.2.1 Стандартный ввод и стандартный вывод

Многие команды системы имеют так называемые стандартный ввод (standard input) и стандартный вывод (standard output), часто сокращаемые до stdin и stdout. Ввод и вывод здесь – это входная и выходная информация для данной команды. Программная оболочка делает так, что стандартным вводом является клавиатура, а стандартным выводом – экран монитора.

Пример с использованием команды cat. По умолчанию команда cat читает данные из всех файлов, которые указаны в командной строке, и посылает эту информацию непосредственно в стандартный вывод (stdout). Следовательно, команда:

```
$ cat history-final masters-thesis
```

выведет на экран сначала содержимое файла history-final, а затем – файла masters-thesis.

Если имя файла не указано, программа cat читает входные данные из stdin и возвращает их в stdout. Пример:

```
$ cat
Hello there.
Hello there.
Bye.
Bye.
<Ctrl>-<D>
```

Каждую строку, вводимую с клавиатуры, программа cat немедленно возвращает на экран. При вводе информации со стандартного ввода конец текста сигнализируется вводом специальной комбинации клавиш, как правило, <Ctrl>-<D>. Сокращённое название сигнала конца текста – EOT (end of text).

11.2.2 Перенаправление ввода и вывода

При необходимости можно перенаправить стандартный вывод, используя символ > и стандартный ввод, используя символ <.

Фильтр (filter) – программа, которая читает данные из стандартного ввода, некоторым образом их обрабатывает и результат направляет на стандартный вывод. Когда применяется перенаправление, в качестве стандартного ввода и вывода могут выступать файлы. Как указывалось выше, по умолчанию, stdin и stdout относятся к клавиатуре и к экрану соответственно. Программа sort является простым фильтром – она сортирует входные данные и посылает результат на стандартный вывод. Совсем простым фильтром является программа cat – она ничего не делает с входными данными, а просто пересылает их на выход.

11.2.3 Использование состыкованных команд

Стыковку команд (pipelines) осуществляет командная оболочка, которая stdout первой команды направляет на stdin второй команды. Для стыковки используется символ |. Направить stdout команды ls на stdin команды sort:

```
$ ls | sort -r
notes
masters-thesis
history-final
english-list
```

Вывод списка файлов частями:

```
$ ls /usr/bin | more
```

Пример стыкования нескольких команд. Команда head – является фильтром следующего свойства: она выводит первые строки из входного потока (в примере на вход будет подан выход от

нескольких состыкованных команд). Если необходимо вывести на экран последнее по алфавиту имя файла в текущем каталоге, можно использовать следующую команду:

```
$ ls | sort -r | head -1 notes
```

где команда head -1 выводит на экран первую строку получаемого ей входного потока строк (в примере поток состоит из данных от команды ls), отсортированных в обратном алфавитном порядке.

11.2.4 Не деструктивное перенаправление вывода

Эффект от использования символа > для перенаправления вывода файла является деструктивным; то есть, команда

```
$ ls > file-list
```

уничтожит содержимое файла file-list, если этот файл ранее существовал, и создаст на его месте новый файл. Если вместо этого перенаправление будет сделано с помощью символов >>, то вывод будет приписан в конец указанного файла, при этом исходное содержимое файла не будет уничтожено.

Примечание. Перенаправление ввода и вывода и стыкование команд осуществляется командными оболочками, которые поддерживают использование символов >, >> и |. Сами команды не способны воспринимать и интерпретировать эти символы.

12 ОБЩИЕ ПРАВИЛА ЭКСПЛУАТАЦИИ

12.1 Включение компьютера

Для включения компьютера необходимо:

- включить стабилизатор напряжения, если компьютер подключен через стабилизатор напряжения;
- включить принтер, если он нужен;
- включить монитор компьютера, если он не подключен к системному блоку кабелем питания;
- включить компьютер (переключателем на корпусе компьютера либо клавишей с клавиатуры).

После этого на экране компьютера появятся сообщения о ходе работы программ проверки и начальной загрузки компьютера.

12.2 Выключение компьютера

Для выключения компьютера надо:

- закончить работающие программы;
- выбрать функцию завершения работы и выключения компьютера, после чего ОС самостоятельно выключит компьютер, имеющий системный блок формата ATX;
- выключить компьютер (переключателем на корпусе AT системного блока);
- выключить принтер;
- выключить монитор компьютера (если питание монитора не от системного блока);
- выключить стабилизатор, если компьютер подключен через стабилизатор напряжения.