УТВЕРЖДЕН

ЛКНВ.11100-01 92 02-ЛУ

ОПЕРАЦИОННАЯ СИСТЕМА АЛЬТ 8 СП

(ОС Альт 8 СП)

Руководство администратора

Виртуализация и контейнеризация

ЛКНВ.11100-01 92 02

Листов 534

2025

Литера О

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

АННОТАЦИЯ

Настоящий документ содержит сведения о средствах виртуализации и контейнеризации программного изделия (ПИ) «Операционная система Альт 8 СП» ЛКНВ.11100-01, сокращенное наименование – ОС Альт 8 СП, варианта исполнения **Сервер релиз 10** для процессоров архитектур **64 бит (x86_64)**, **AArch64 (ARMv8)**.

Далее в документе будет использоваться альтернативное наименование ПИ: ОС Альт СП.

Версия документа: 2.2.

Документ предназначен для администратора ОС Альт СП и содержит общие сведения о структуре, настройке и работе со средствами виртуализации и контейнеризации ОС Альт СП, компонентами виртуальной инфраструктуры.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

ОС Альт СП Сервер 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 операционной системы (ОС) как на одной персональной электронно-вычислительной машине (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 (ВМ);
- обеспечивает сетевую обработку данных, в том числе разграничение доступа к сетевым пакетам.

В ОС Альт СП выполняются следующие функциональные требования безопасности к виртуальной инфраструктуре:

- идентификация и аутентификация субъектов доступа и объектов доступа в виртуальной инфраструктуре, в том числе администраторов управления средствами виртуализации (ЗСВ.1);
- управление доступом субъектов доступа к объектам доступа в виртуальной инфраструктуре, в том числе внутри ВМ (ЗСВ.2);
- регистрация событий безопасности в виртуальной инфраструктуре (ЗСВ.3);
- управление (фильтрация, маршрутизация, контроль соединения, однонаправленная передача) потоками информации между компонентами виртуальной инфраструктуры, а также по периметру виртуальной инфраструктуры (ЗСВ.4);
- доверенная загрузка серверов виртуализации, виртуальной машины (ЗСВ.5);
- управление перемещением ВМ (контейнеров) и обрабатываемых на них данных (ЗСВ.6);

- контроль целостности виртуальной инфраструктуры и ее конфигураций (ЗСВ.7);
- резервное копирование данных и компонентов средств виртуализации (ЗСВ.8).

Дистрибутив ОС Альт СП предоставляет администратору возможность размещать системные службы (сервисы) в изолированных окружениях, ВМ.

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

ОС Альт СП 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.

ОС Альт СП Сервер предоставляет средства виртуализации и набор дополнительных служб, востребованных в инфраструктуре виртуализации любой сложности и архитектуры:

1) базовый гипервизор (libvirt, qemu-kvm);

2) контейнерная виртуализация (kubernetes, podman);

3) ПО для организации хранилища:

- распределенная сетевая файловая система CEPH;
- распределенная сетевая файловая система GlusterFS;
- сервер сетевой файловой системы NFS;
- поддержка iSCSI как в качестве клиента, так и создание сервера;

4) ПО для сети:

- 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, keepalived;
- веб-серверы Apache и Nginx;

5) ПО для мониторинга (zabbix-agent, prometheus-node_exporter);

6) ПО резервного копирования (rsync).

2. УПРАВЛЕНИЕ ВИРТУАЛИЗАЦИЕЙ НА ОСНОВЕ LIBVIRT

2.1. Установка и настройка libvirt

Виртуализацию как таковую можно по подходу разбить на несколько типов:

- полная виртуализация;
- паравиртуализация;
- виртуализация окружения.

Различные типы виртуализации реализуются различными системами виртуализации. В ОС Альт СП не поддерживается паравиртуализация. Полная виртуализация представлена системой виртуализации KVM (Kernel-based Virtual Machine).

Различные системы виртуализации представляют различные интерфейсы для управления ВМ и контейнерами. Однако набор действий, производимых с ВМ и контейнерами, не меняется от системы виртуализации к системе виртуализации, меняется лишь способ указания системе виртуализации произвести те или иные действия. Эта особенность позволила создать некоторый общий API и набор утилит для управления ВМ, поддерживающий различные системы виртуализации.

libvirt – это набор инструментов, предоставляющий единый API к множеству различных технологий виртуализации.

Кроме управления ВМ/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, например, на компьютере с ОС Альт СП Рабочая станция, можно воспользоваться virt-manager (группа пакетов «Управление локальными и удаленными виртуальными машинами»).

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.

Для развертывания libvirt в уже установленной системе, достаточно установить пакеты:

```
# apt-get update
# apt-get install libvirt-kvm virt-install
```

Запуск службы:

```
# systemctl start libvirtd
# systemctl enable libvirtd
```

Для непривилегированного доступа (не `root`) к управлению libvirt, нужно добавить пользователя в группу `vmusers`:

```
# usermod -a -G vmusers user
```

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.

2.2. 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-install – создание ВМ с помощью опций командной строки;
- virt-xml – редактирование XML-файлов описаний ВМ.

2.2.1. Утилита Virsh

virsh – утилита для командной строки, предназначенная для управления ВМ и гипервизорами KVM.

virsh использует libvirt API и служит альтернативой графическому менеджеру ВМ (virt-manager).

С помощью virsh можно сохранять состояние ВМ, переносить ВМ между гипервизорами и управлять виртуальными сетями.

В таблице 1 и таблице 2 приведены основные параметры утилиты командной строки virsh.

Т а б л и ц а 1 – Команды управления ВМ

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
`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

Для получения списка доступных команд или параметров, выполните команду:

```
$ virsh help
```

Т а б л и ц а 2 – Параметры управления ресурсами ВМ и гипервизора

Команда	Описание
`setmem`	Определяет размер выделенной ВМ памяти
`setmaxmem`	Ограничивает максимально доступный гипервизору объем памяти
`setvcpus`	Изменяет число предоставленных ВМ виртуальных процессоров
`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 работающей 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 определения XML, что и `attach-device`)
`detach-disk`	Отключить дисковое устройство от ВМ
`detach-interface`	Отключить сетевой интерфейс от ВМ

2.2.2. Утилита virt-install

virt-install – это инструмент для создания ВМ в командной строке.

Далее подробно рассматриваются возможности создания ВМ при помощи этой утилиты. В таблице 3 приведено описание только наиболее часто используемые опции virt-install. Описание всех доступных опций можно получить, выполнив команду:

```
$ man virt-install
```

Процесс создания ВМ с использованием virt-install и ее опции описаны также далее в п. 2.4.2.

Т а б л и ц а 3 – Параметры команд virt-install

Опции	Описание
-n NAME, --name=NAME	Имя новой ВМ. Это имя должно быть уникально внутри одного гипервизора.
--memory MEMORY	Определяет размер выделенной ВМ памяти (в Мбайт).
--vcpus VCPUS	Определяет количество виртуальных центральных процессорных устройств (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.
--boot BOOT	Параметры загрузки ВМ. Например: --boot hd,cdrom,menu=on --boot init=/sbin/init (для контейнеров)
--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 Гбайт 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

Окончание таблицы 3

Опции	Описание

`--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	
`--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.

2.2.3. Утилита qemu-img

qemu-img – инструмент для манипулирования 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/alt10.1.qcow2
image: /var/lib/libvirt/images/alt10.1.qcow2
file format: qcow2
virtual size: 12 GiB (12884901888 bytes)
disk size: 12 GiB
cluster_size: 65536
Format specific information:
     compat: 1.1
     lazy refcounts: true
     refcount bits: 16
     corrupt: 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
```

2.2.4. Менеджер ВМ virt-manager

Менеджер ВМ 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 ВМ;
- доступ к текстовой и графической консоли;
- автономная и живая миграция.

Для запуска менеджера ВМ, в меню приложений необходимо выбрать «Системные» → «Менеджер виртуальных машин» («Manage virtual machines»).

П р и м е ч а н и е . Должен быть установлен пакет virt-manager.

В главном окне менеджера (рис. 1) показаны все запущенные ВМ и выделенные им ресурсы. Поля можно отфильтровать. Двойной щелчок на имени ВМ открывает ее консоль. Выбор ВМ и двойной щелчок на кнопке «Подробности» («Details») откроет окно сведений об этой машине.

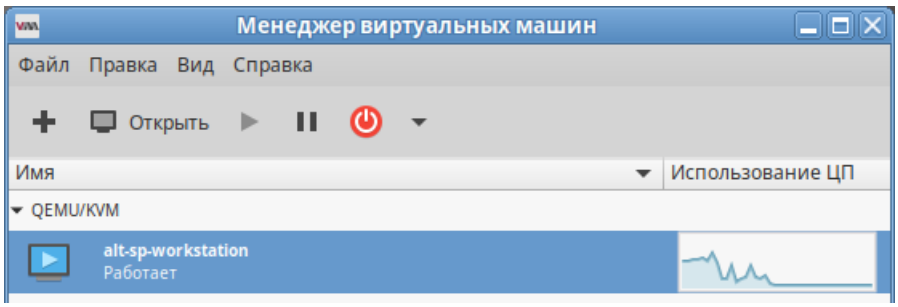

Рис. 1 – Главное окно менеджера ВМ

2.3. Подключение к гипервизору

2.3.1. Управление доступом к libvirt через SSH

В дополнение к аутентификации SSH также необходимо определить управление доступом для службы libvirt в хост-системе (рис. 2).

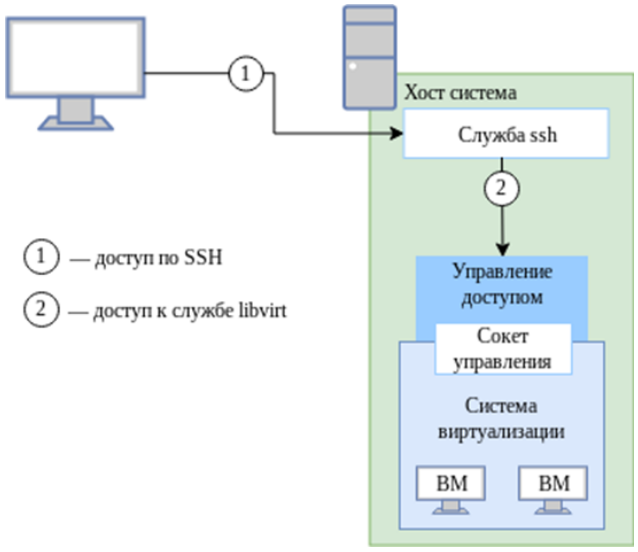

Рис. 2 – Доступ к libvirt с удаленного узла

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 rsa
$ ssh-copy-id user@192.168.88.185
```

где `192.168.88.185`– IP-адрес сервера с libvirt.

В результате появится 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`.

2.3.2. 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.

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                Статус
----------------------------------------------------

 -          alt10.1            выключен
```

Подключение к удаленному гипервизору QEMU через протокол SSH:

```
$ virsh -c qemu+ssh://user@192.168.88.185/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`;
- `192.168.88.185` – IP-адрес или имя хоста ВМ.

2.3.3. Настройка соединения с удаленным гипервизором в virt-manager

virt-manager позволяет управлять несколькими удаленными хостами ВМ.

Для создания нового подключения необходимо в меню менеджера ВМ выбрать «Файл» → «Добавить соединение…».

В открывшемся окне необходимо выбрать сессию гипервизора, отметить пункт «Подключиться к удаленному узлу с помощью SSH» и ввести имя пользователя и адрес сервера (рис. 3).

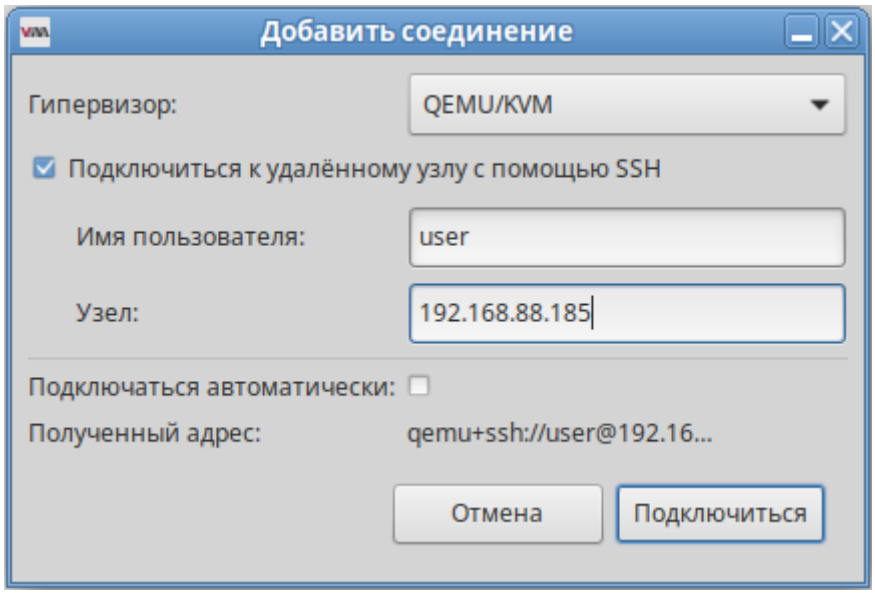

Рис. 3 – Окно соединений менеджера ВМ

Примечание. На управляющей системе можно запустить virt-manager, выполнив следующую команду:

```
virt-manager -c qemu+ssh://user@192.168.88.185/system
```

где:

- `user` – имя пользователя на удаленном хосте, который входит в группу `vmusers`;
- `192.168.88.185` – IP-адрес или имя хоста ВМ.

2.4. Создание ВМ

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 программы virt-install или из пользовательского интерфейса программы virt-manager.

2.4.1. Создание ВМ на основе файла конфигурации

ВМ могут быть созданы из файлов конфигурации. Можно сделать копию существующего XML-файла ранее созданной ВМ, или использовать опцию `dumpxml`.

Вывод файла конфигурации ВМ:

```
# virsh dumpxml <domain-id, domain-name или domain-uuid>
```

Эта команда выводит XML-файл конфигурации ВМ в стандартный вывод (`stdout`). Можно сохранить данные, отправив вывод в файл.

Пример передачи вывода в файл `guest.xml`:

```
# virsh dumpxml alt10.1 > guest.xml
```

Можно отредактировать этот файл конфигурации, чтобы настроить дополнительные устройства или развернуть дополнительные ВМ.

Команда создания ВМ из XML файла:

```
# virsh create guest.xml
```

2.4.2. Создание ВМ с помощью virt-install

Утилита virt-install поддерживает как графическую установку ОС 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 ВМ/контейнера без установки ОС.

Утилита virt-install поддерживает большое число опции, позволяющих создать полностью независимую ВМ, готовую к работе, что хорошо подходит для автоматизации установки ВМ.

Минимальные требуемые опции: `--name`, `--memory`, хранилище (`--disk`, `--filesystem` или `--nodisks`) и опции установки. Далее описаны только наиболее часто используемые опции.

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-variant=alt10.1 \
--cdrom /var/lib/libvirt/images/alt-server-v-x86_64.iso \
--graphics vnc \
--disk pool=default,size=20,bus=virtio,format=qcow2 \
--memory 2048 \
--vcpus=2 \
--network network=default \
--hvm \
--virt-type=kvm
```

где:

- `--name alt-server` – название ВМ;
- `--os-variant=alt.p10` – версия ОС;
- `--cdrom /var/lib/libvirt/images/alt-server-v-x86_64.iso` – путь к ISO-образу установочного диска ОС;
- `--graphics vnc` – графическая консоль;
- `--disk pool=default, size=20, bus=virtio, format=qcow2` – хранилище ВМ будет создано в пространстве объемом 20 Гбайт, которое автоматически выделяется из пула хранилищ `default`. Образ диска для этой ВМ будет создан в формате `qcow2`;
- `--memory 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 эти опции обеспечивают определенную оптимизацию в процессе создания и установки ОС; если эти опции не заданы в явном виде, то вышеуказанные значения применяются по умолчанию.

Можно использовать подобную команду для создания ВМ, использующую другую ОС. С этой целью нужно задать надлежащее имя для ВМ и соответствующим образом изменить аргументы опций `--cdrom` и `--os-variant`.

Список доступных вариантов ОС можно получить, выполнив команду:

```
$ osinfo-query os
```

2.4.2.1. Общие опции

Общие опции утилиты virt-install:

1) `-h`, `--help` – показать помощь и выйти;
2) `-q`, `--quiet` – печатать только сообщения о критических ошибках;
3) `-d`, `--debug` – выводить отладочную информацию в терминал. Отладочная информация выводится в файл `$HOME/.virtinst/virt-install.log` даже если эта опция не указана;
4) `--connect=URI` – подключить к заданному гипервизору. Если данная опция не задана, то libvirt попытается определить наиболее подходящий гипервизор. Доступные значения:
   - `qemu:///system` – для создания KVM и QEMU гостей, запускаемых системным экземпляром libvirtd. Этот вариант используется по умолчанию для пользователей virt-manager;
   - `qemu:///session` – для созданий KVM и QEMU гостей, запускаемых от имени обычного пользователя;
   - `xen:///` – для создания Xen;
5) `-n ИМЯ`, `--name=ИМЯ` – задает имя новой ВМ. Это имя должно быть уникально внутри одного гипервизора. Для того чтобы переопределить

существующего гостя следует сначала воспользоваться virsh для остановки и удаления его (`virsh shutdown` и `virsh undefine`);

6) `--memory ПАМЯТЬ` – задает объем оперативной памяти гостя в Мбайтах. Если у гипервизора недостаточно памяти для того чтобы назначить ее гостю, то он автоматически возьмет недостающую часть у хост-системы;
7) `--arch=архитектура` – задает «не родную» архитектуру процессора для ВМ. Если данная опция не указана, то будет использована та же архитектура процессора что и у процессора хост-системы;
8) `--vcpus=ВИРТCPU[,maxvcpus=МАКСИМУМ][,sockets=#][,cores=#][,threads=#]` – задает число виртуальных процессоров для гостя. Если задано значение для `maxvcpus`, то гость будет иметь возможность подключать до `МАКСИМУМ` виртуальных процессоров, но запускаться он будет с `ВИРТCPU`. Также для виртуального процессора можно задать число сокетов, ядер и нитей. Если какие-то из этих значений не указаны, то они будут автоматически вычислены.

2.4.2.2. Опции метода установки

Опции метода установки утилиты virt-install:

1) `-c CDROM`, `--cdrom=CDROM` – для гостей с полной виртуализацией задает файл или устройство, которое будет использоваться как устройство CD-ROM. Может указываться на файл ISO-образа или на устройство чтения CD/DVD-дисков. Также может быть URL до ISO-образа. Формат URL такой же, как и в опции `--location`;
2) `-l РАСПОЛОЖЕНИЕ`, `--location=РАСПОЛОЖЕНИЕ` – указывает расположение дистрибутива для установки. Для некоторых дистрибутивов virt-install может распознать ядро и initrd, и получить их перед запуском установки.

   `РАСПОЛОЖЕНИЕ` может указываться в следующих формах:

   - `ДИРЕКТОРИЯ` – путь до локальной директории, содержащей установочный образ дистрибутива;

- `nfs:хост:/путь` или `nfs://хост/путь` – NFS-путь, по которому доступен установочный образ дистрибутива;
- `http://хост/путь` – HTTP-путь, по которому доступен установочный образ дистрибутива;
- `ftp://хост/путь` – FTP-путь, по которому доступен установочный образ дистрибутива;

3) `--import` – пропустить установку ОС, и создать ВМ с существующим диском. В качестве загрузочного устройства будет использоваться первое указанное, с опцией `--disk` или `--filesystem`;
4) `--init=ПУТЬ_К_INIT` – задает путь для бинарного файла, который будет использоваться в гостевой системе в качестве процесса init. Если корневая файловая система задана через `--filesystem`, то по умолчанию будет использоваться `/sbin/init`, в противном случае – `/bin/sh`;
5) `--livecd` – указывает, что установочный диск является LiveCD и что следует настроить ВМ на постоянную загрузку с CDROM. Может быть полезно в сочетании с опцией `--nodisks`;
6) `-x ДОПОЛНИТЕЛЬНО`, `--extra-args=ДОПОЛНИТЕЛЬНО` – дополнительные аргументы ядра, передаваемые в процессе установки (если указана опция `location`);
7) `--os-variant=ВАРИАНТ_ОС` – дополнительные оптимизации ВМ для конкретного варианта ОС. Данная опция не является обязательной. По умолчанию virt-install пытается автоматически определить `ВАРИАНТ_ОС` на основании установочного носителя. Автоопределение можно отключить, используя значение none. Если задано значение `list`, то virt-install напечатает список доступный вариантов ОС.

2.4.2.3. Опции хранилища

Опции хранилища утилиты virt-install:

`--disk=ОПЦИИ_ДИСКА` – задает носитель для использования в качестве хранилища в ВМ. Общий формат следующий:

```
--disk опция1=значение1,опция2=значение2,...
```

Для задания носителя может использоваться сокращенный формат:

`--disk /some/storage/path,опция1=значение1`

Или же может быть использован один из следующих аргументов:

- `path` – путь к какому-либо носителю (существующему или не существующему). Существующий носитель может быть либо файлом, либо блочным устройством. При установке на удаленной хост-системе существующий носитель должен быть общим как том хранилища libvirt. Указание несуществующего пути подразумевает попытку создать новое хранилище и требует задания значения `size`. Если директория в `path` это пул хранилища libvirt на хост-системе, то новое хранилище будет создано как том хранилища libvirt. Для удаленных систем директория должна указывать на пул хранилища;
- `pool` – имя существующего пула хранилищ, в котором будет создано новое хранилище. Также требует указания значения `size`;
- `vol` – имя существующего тома в хранилище libvirt. Указывается как `poolname/volname`;
- `size` – размер в Гбайт для создаваемых хранилищ;
- `--nodisks` – указывает, что у ВМ не должно быть логических дисков для хранения информации. Обычно данная опция используется для запуска LiveCD образов или при установке на сетевое хранилище типа iSCSI или NFS.

2.4.2.4. Опции сети

Опции сети утилиты `virt-install`:

`-w СЕТЬ, --network=СЕТЬ,опция1=значение1,опция2=значение2` – подключить ВМ к сети.

Значение `СЕТЬ` может задаваться в трех формах:

- `bridge=МОСТ` – подключить к устройству типа мост с именем `МОСТ` в хост-системе. Используйте эту опцию, если в хост-системе заданы постоянные сетевые настройки и гостевая система требует прямого

взаимодействия с локальной сетью. Также `bridge` следует использовать, если требуется живая миграция ВМ;

- `network=ИМЯ` – подключить к виртуальной сети хост-системы под названием `ИМЯ`. Виртуальные сети можно просматривать, создавать и удалять при помощи утилиты командной строки virsh. По умолчанию при установке libvirt создается сеть с именем `default`. Используйте виртуальную сеть в случае, если в хост-системе могут меняться сетевые настройки (например, при помощи NetworkManager) или же используется WiFi. Для пакетов из ВМ будет применяться трансляция адресов в подключенную в данный момент локальную сеть;
- `user` – подключить к локальной сети при помощи SLiRP. Используйте только при запуске ВМ QEMU от непривилегированного пользователя. Предоставляет очень ограниченный вариант трансляции адресов.

Если данная опция не указана, то будет создан один сетевой интерфейс. Если на физическом интерфейсе в хост-системе создан мостовой интерфейс, то будет использован именно он. В противном случае будет использовано подключение к виртуальной сети `default`. Эту опцию можно указать несколько раз для создания нескольких интерфейсов.

Другие доступные опции:

- `model` – задать сетевое устройство (как оно будет отображаться в ВМ). В качестве значений следует использовать поддерживаемые гипервизором, например, `e1000`, `rtl8139`, `virtio` и другие;
- `mac` – задать MAC-адрес для ВМ. Если данный параметр не указан (или указано значение `RANDOM`), то будет сгенерирован случайный MAC-адрес. Для ВМ QEMU и KVM MAC-адрес должен начинаться с 52:54:00;
- `--nonetworks` – используется для создания ВМ без сетевых интерфейсов.

2.4.2.5. Опции графики

Если не заданы графические опции, то, в зависимости от того задана переменная окружения `DISPLAY` или нет, будет использоваться по умолчанию опция `--graphics vnc` или `--graphics none` соответственно.

`--graphics ТИП,опция1=аргумент1,опция2=аргумент2,...` – задает настройки для виртуального монитора. Данная опция не влияет на оборудование ВМ, она лишь задает способ доступа к монитору ВМ.

Поддерживаются следующие опции:

1) `type` – тип дисплея. Может быть одним из:
   - `vnc` – предоставить доступ к дисплею ВМ при помощи VNC (при этом используется адрес хост-системы). Если не задать параметр `port`, то будет использован первый свободный порт выше `5900`. Актуальные значение параметров VNC, назначенные ВМ, можно получить при помощи команды `virsh vncdisplay`;
   - `sdl` – открыть дисплей ВМ в SDL-окне на хост-системе. Если это окно закрыть, то ВМ будет также закрыта;
   - `spice` – экспортировать дисплей ВМ по протоколу Spice. Spice поддерживает проброс аудио и USB-устройств на ВМ, а также имеет высокую графическую производительность;
   - `none` – графическая консоль не будет подключена в ВМ. Для ВМ с полной виртуализацией (Xen и QEMU/KVM) потребуется на первом последовательном порту гостевой системы (этого можно добиться при помощи опции `--extra-args`). Для подключения к последовательному устройству можно использовать `virsh console ИМЯ`;
2) `port` – задать постоянный порт для дисплея ВМ. Используется совместно с `vnc` и `spice`.

2.4.2.6. Опции виртуализации

Опции виртуализации утилиты virt-install:

- `-v`, `--hvm` – если на хост-системе доступны полная виртуализация и паравиртуализация, то будет использована полная виртуализация. Эта опция недоступна при подключении к XEN без аппаратной поддержки виртуализации. Эта опция предполагается по умолчанию при подключении к гипервизору QEMU;

- `--container` – гостевая машина должна быть типа контейнер. Данную опцию следует использовать, только если хост-система также поддерживает другие типы виртуализации. Она предполагается по умолчанию для OpenVZ, но может быть явно указана для полноты;
- `--virt-type` – задает тип используемого гипервизора, например: kvm, qemu, xen или kqemu. Доступные типы виртуализации можно увидеть в тегах `domain`.

2.4.2.7. Опции устройств

Опции устройств утилиты virt-install:

- `--serial=ОПЦИИ` – задает подключение последовательного устройства к ВМ.

Общий формат следующий:

- `--serialtype,опция1=значение1,опция2=значение2,...`
- `--serialpty` – псевдо-TTY. Назначенное pty-устройство можно будет узнать из XML-описания ВМ;
- `--serialfile,path=ИМЯ_ФАЙЛА` – записывать вывод в файл `ИМЯ_ФАЙЛА`.

2.4.2.8. Примеры установки ОС в гостевые системы

Установка Fedora 13 в гостевую систему на базе KVM с диском и сетью работающими по virtio, с созданием файла хранилища размеров 8 Гбайт, с установкой с CD/DVD диска, находящегося в приводе хост-системы, а также с автоматическим запуском VNC-клиента:

```
# virt-install \
 --connect qemu:///system \
 --virt-type kvm \
 --name demo \
 --memory 500 \
 --disk path=/var/lib/libvirt/images/demo.img,size=8 \
 --graphics vnc \
 --cdrom /dev/cdrom \
 --os-variant fedora13
```

Установка Fedora 9 в гостевую систему на базе QEMU, с LVM разделом, с подключением к виртуальной сети, загрузкой по сети и с использованием VNC для доступа к дисплею:

```
# virt-install \
 --connect qemu:///system \
 --name demo \
```

```
 --memory 500 \
 --disk path=/dev/HostVG/DemoVM \
 --network network=default \
 --virt-type qemu
 --graphics vnc \
 --os-variant fedora9
```

Установка FedoraCore 6 в гостевую систему на базе QEMU с архитектурой процессора, отличной от архитектуры хост-системы, использованием SDL для доступа к дисплею ВМ, а также с использованием удаленных ядра и initrd:

```
# virt-install \
 --connect qemu:///system \
 --name demo \
 --memory 500 \
 --disk path=/dev/hdc \
 --network bridge=eth1 \
 --arch ppc64 \
 --graphics sdl \
 --location http://download.fedora.redhat.com/pub/fedora/linux/core/6/x86_64/os/
```

Запуск Live CD в ВМ без дисков:

```
# virt-install \
 --hvm \
 --name demo \
 --memory 500 \
 --nodisks \
 --livecd \
 --graphics vnc \
 --cdrom /var/lib/libvirt/images/altlive.iso
```

Создать ВМ, используя существующий том хранилища:

```
# virt-install \
 --name demo \
 --memory 512 \
 --disk /home/user/VMs/mydisk.img \
 --import
```

Тестировать отдельное ядро и initrd при запуске существующего тома хранилища с привязкой последовательного порта ВМ к `ttyS0` на хост-системе:

```
# virt-install \
--name mykernel \
--memory 512 \
--disk /home/user/VMs/mydisk.img \
--boot \
kernel=/tmp/mykernel,initrd=/tmp/myinitrd,kernel_args="console=ttyS0" \
--serial pty
```

2.4.3. Создание ВМ с помощью virt-manager

Создание новой ВМ:

- нажать на кнопку «Создать виртуальную машину» в главном окне virt-manager,

либо

- выбрать в меню «Файл» → «Создать виртуальную машину».

На первом шаге создания ВМ необходимо выбрать метод установки ОС (рис. 4) и нажать на кнопку «Вперед».

В следующем окне для установки гостевой ОС требуется указать ISO-образ установочного диска ОС или CD/DVD-диск с дистрибутивом (рис. 5).

Данное окно будет выглядеть по-разному в зависимости от выбора, сделанного на предыдущем этапе.

Здесь также можно указать версию устанавливаемой ОС.

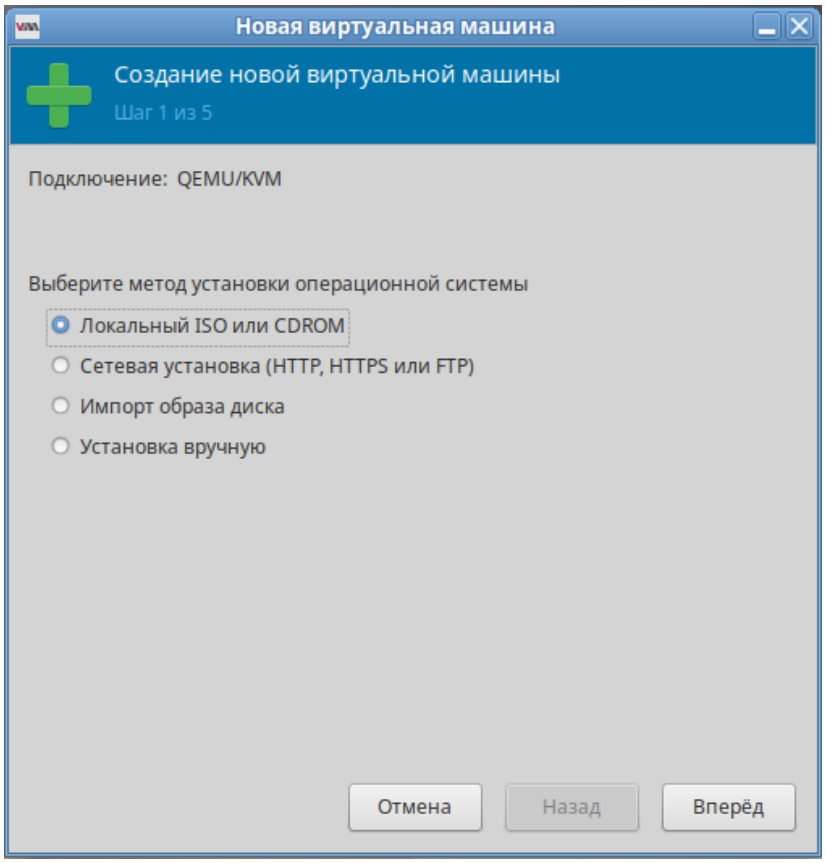

Рис. 4 – Создание ВМ. Выбор метода установки

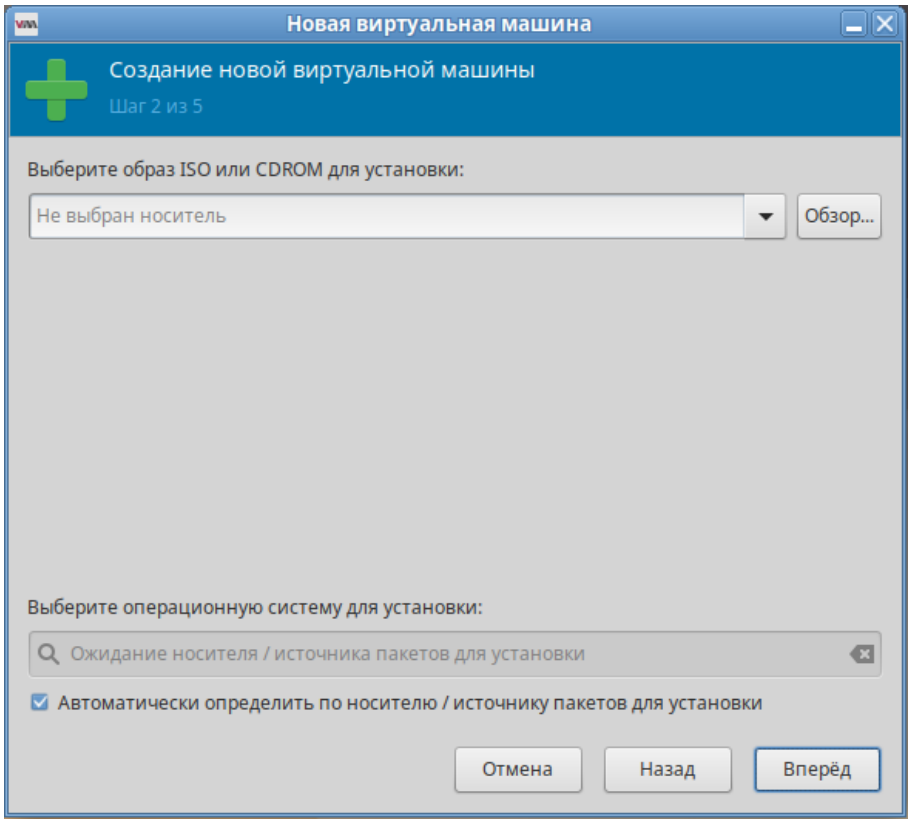

Рис. 5 – Создание ВМ. Выбор ISO образа

На третьем шаге необходимо указать размер памяти и количество процессоров для ВМ (рис. 6). Эти значения влияют на производительность хоста и ВМ.

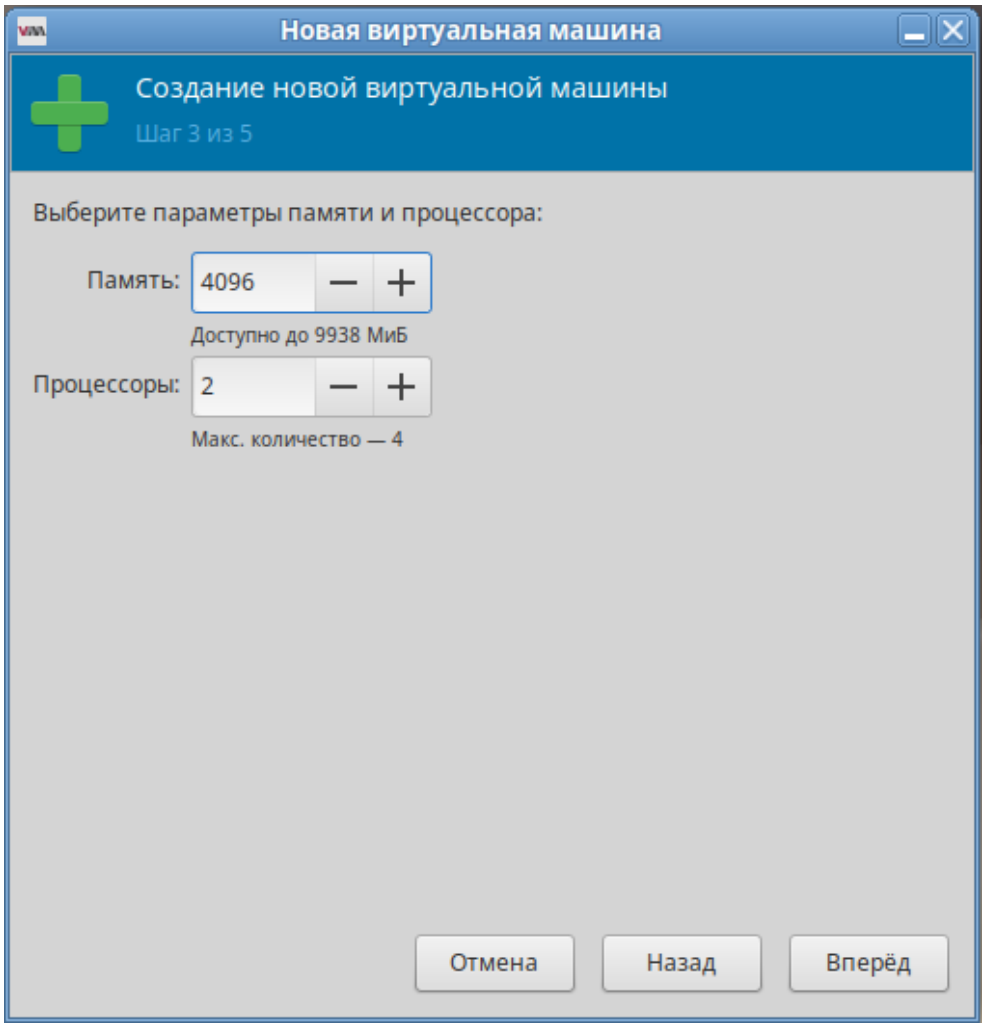

Рис. 6 – Создание ВМ. Настройка оперативного запоминающего устройства (ОЗУ) и ЦПУ для ВМ

На следующем этапе настраивается пространство хранения данных (рис. 7).

На последнем этапе (рис. 8) можно задать название ВМ, выбрать сеть и нажать на кнопку «Готово».

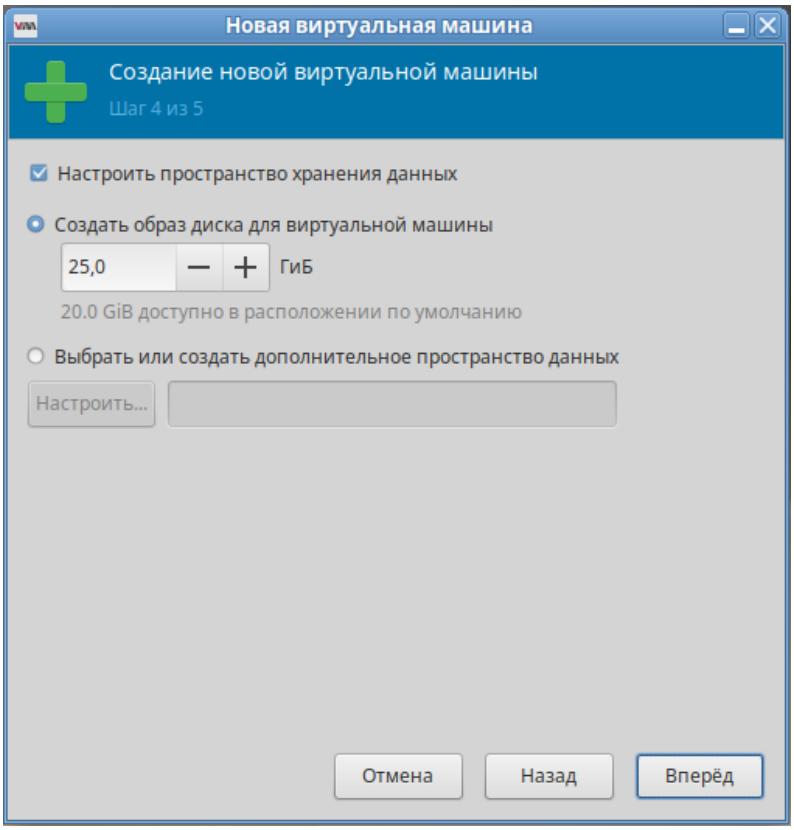

Рис. 7 – Создание ВМ. Настройка пространства хранения данных

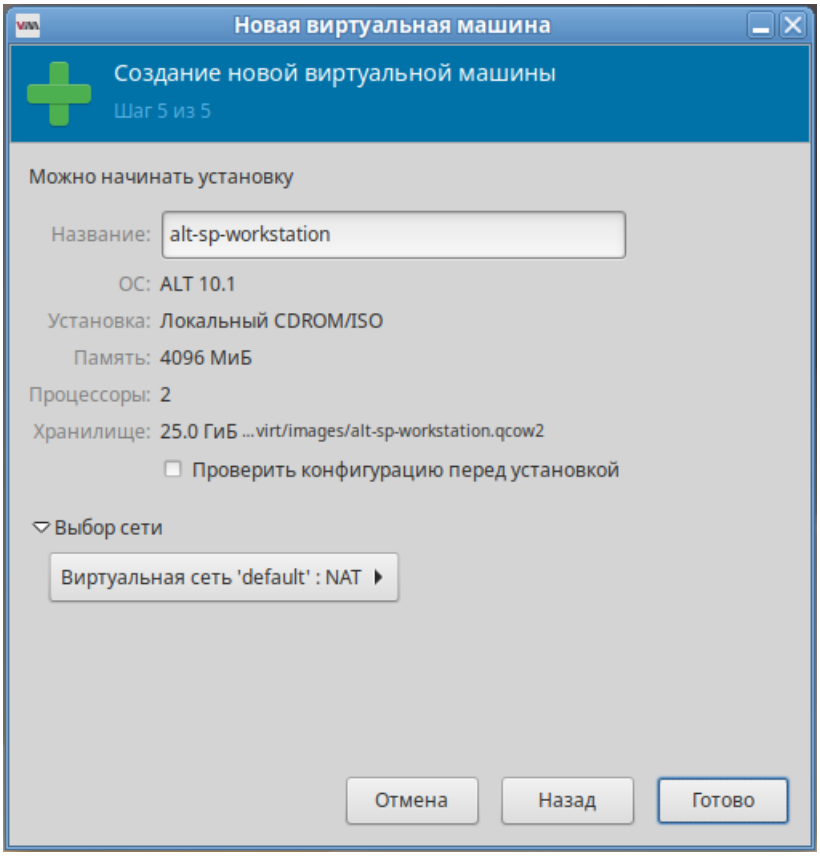

Рис. 8 – Создание ВМ. Выбор сети

В результате созданная ВМ будет запущена и после завершения исходной загрузки начнется стандартный процесс установки ОС.

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 клавиш <Ctrl>+<Alt>+<F11>, необходимо последовательно нажать клавиши <Ctrl>+<Ctrl>+<Ctrl>+<Alt>+<F11> или воспользоваться меню «Отправить комбинацию клавиш».

2.5. Управление ВМ

2.5.1. Управление конфигурацией ВМ

2.5.1.1. 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`.

2.5.1.2. Получение информации о ВМ

Команда для получения информации о ВМ:

```
virsh dominfo <domain-id, domain-name or domain-uuid>
```

Пример вывода `virsh dominfo`:

```
$ virsh dominfo alt10.1
ID:       -
Имя:      alt10.1
UUID:           e645d4c4-4044-42cc-af17-91ef146dcd9d
Тип ОС:         hvm
Статус:         выключен
```

```
CPU:           1
Макс.память: 512000 KiB
Занято памяти: 512000 KiB
Постоянство: yes
Автозапуск: выкл.
Управляемое сохранение: no
Модель безопасности: none
DOI безопасности: 0
```

Команда получения информации об узле:

```
virsh nodeinfo
```

Пример вывода `virsh nodeinfo`:

```
$ virsh nodeinfo
Модель процессора: x86_64
CPU:                   1
Частота процессора: 1995 MHz
Сокеты:             1
Ядер на сокет:      1
Потоков на ядро: 1
Ячейки NUMA:        1
Объем памяти: 1003296 KiB
```

Вывод содержит информацию об узле и машинах, поддерживающих виртуализацию.

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            Статус
----------------------------------------
8    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 ОС 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-id, domain-name or domain-uuid>
```

Пример вывода:

```
# virsh vcpuinfo alt-server
VCPU:     0
CPU:      0
Статус:   работает
Время CPU: 115,3s
Соответствие CPU: y
```

Команда сопоставления виртуальных процессоров физическим:

```
virsh vcpupin <domain-id, domain-name or domain-uuid> vcpu, cpulist
```

Здесь `vcpu` – номер виртуального процессора, а `cpulist` – сопоставляемые ему физические процессоры.

Команда изменения числа процессоров для домена (заданное число не может превышать значение, определенное при создании ВМ):

```
virsh setvcpus <domain-id, domain-name or domain-uuid> count [-- maximum] [--config] [--live] [--current] [--guest]
```

где:

- `[--count] <число>` – число виртуальных процессоров;

- `--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-id or domain-name> size [--config] [--live] [-- current]
```

где:

- `[--size] <число>` – целое значение нового размера памяти (по умолчанию в Кбайт);
- `--config` – с сохранением после перезагрузки;
- `--live` – применить к работающему домену;
- `--current` – применить к текущему домену.

Объем памяти, определяемый заданным числом, должен быть указан в Кбайт. Объем не может превышать значение, определенное при создании ВМ, но в то же время не должен быть меньше 64 Мбайт. 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 для изменения максимального ограничения памяти:

```
virsh setmaxmem <domain-id or domain-name> size [--config] [--live] [--current]
```

где:

- `[--size] <число>` – целое значение максимально допустимого размера памяти (по умолчанию в Кбайт);
- `--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 --connect qemu:///system setmaxmem --size 624000 alt10.1
# virsh --connect qemu:///system setmem --size 52240 alt10.1
# virsh --connect qemu:///system setvcpus --config alt10.1 3 --maximum
```

Команда для получения информации о блочных устройствах работающей ВМ:

```
virsh domblkstat GuestName <block-device>
```

Команда для получения информации о сетевых интерфейсах работающей ВМ:

```
virsh domifstat GuestName <interface-device>
```

2.5.1.3. Конфигурирование ВМ в менеджере ВМ

С помощью менеджера ВМ можно получить доступ к подробной информации обо всех ВМ, для этого следует:

1) в главном окне менеджера выбрать ВМ;
2) нажать на кнопку «Открыть» (рис. 9);
3) в открывшемся окне нажать на кнопку «Показать виртуальное оборудование» (рис. 10);
4) появится окно просмотра сведений ВМ.

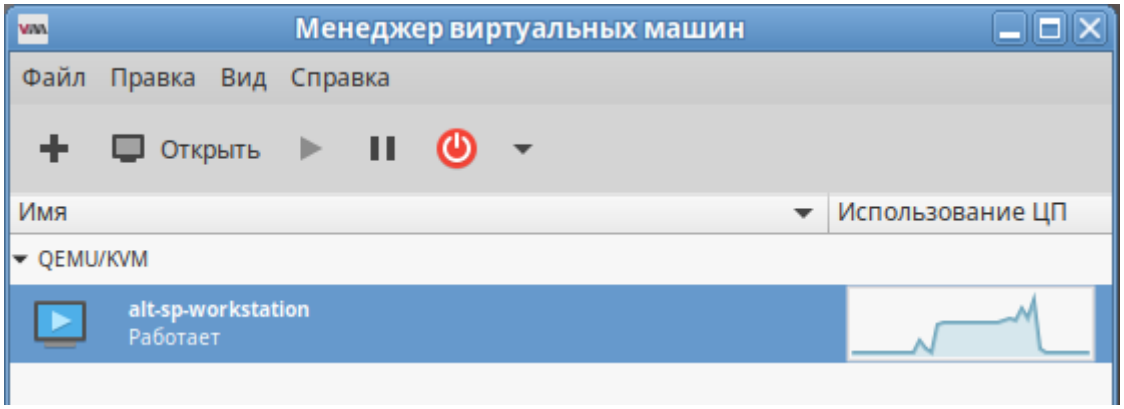

Рис. 9 – Окно менеджера ВМ

Для изменения требуемого параметра необходимо перейти на нужную вкладку (например, рис. 11, рис. 12), внести изменения и подтвердить операцию – нажать на кнопку «Применить».

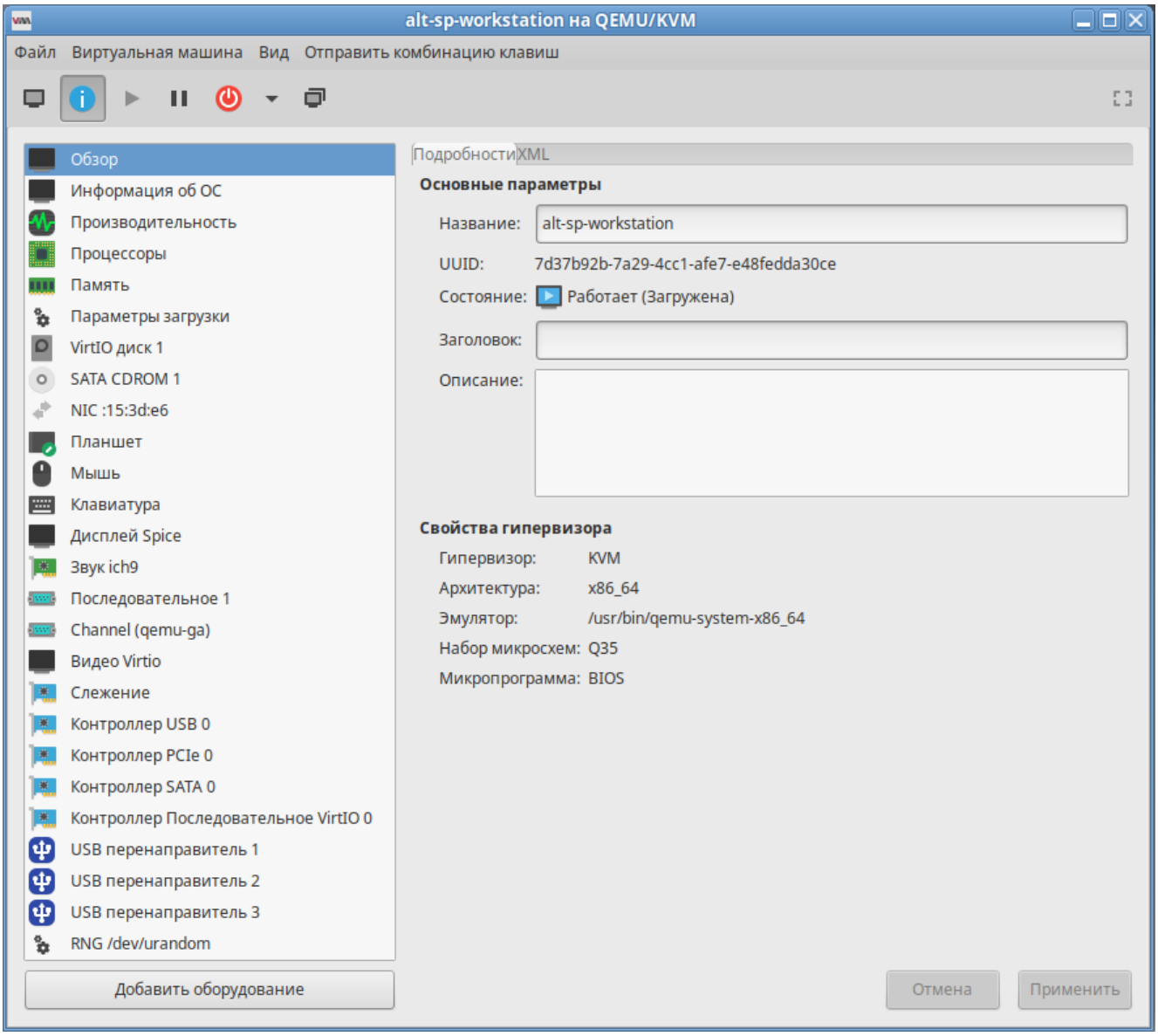

Рис. 10 – Окно параметров ВМ

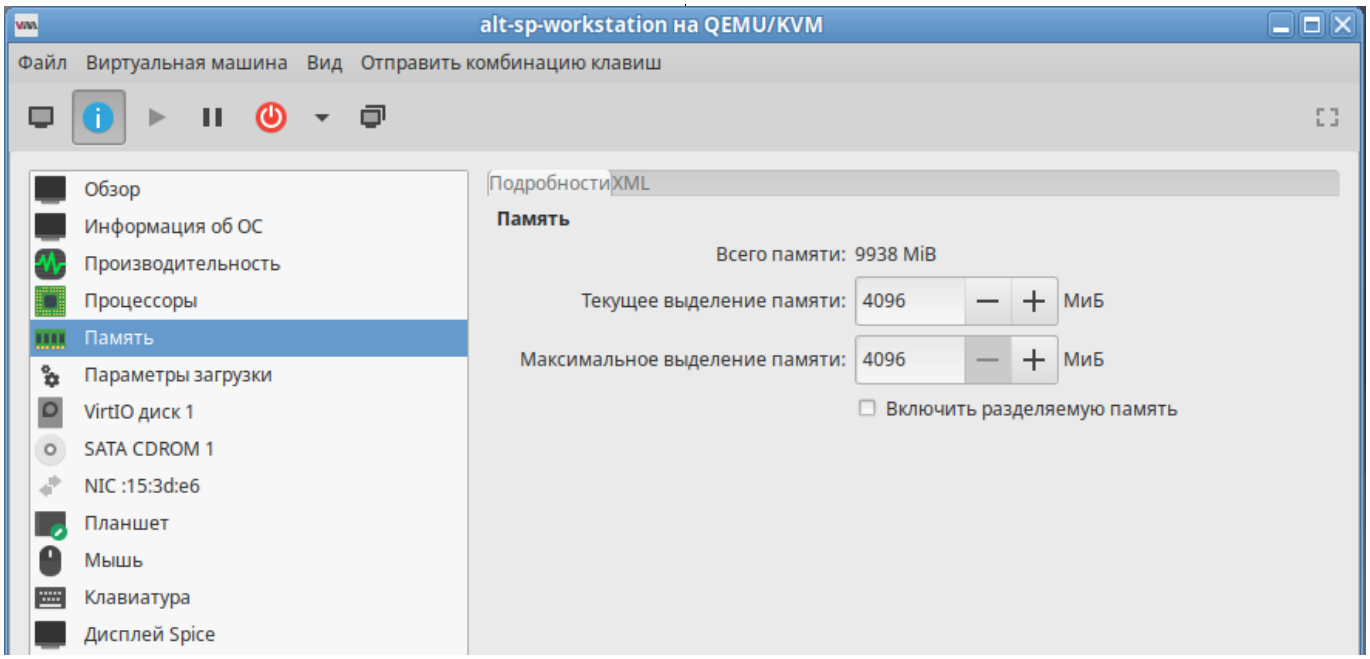

Рис. 11 – Вкладка «Память»

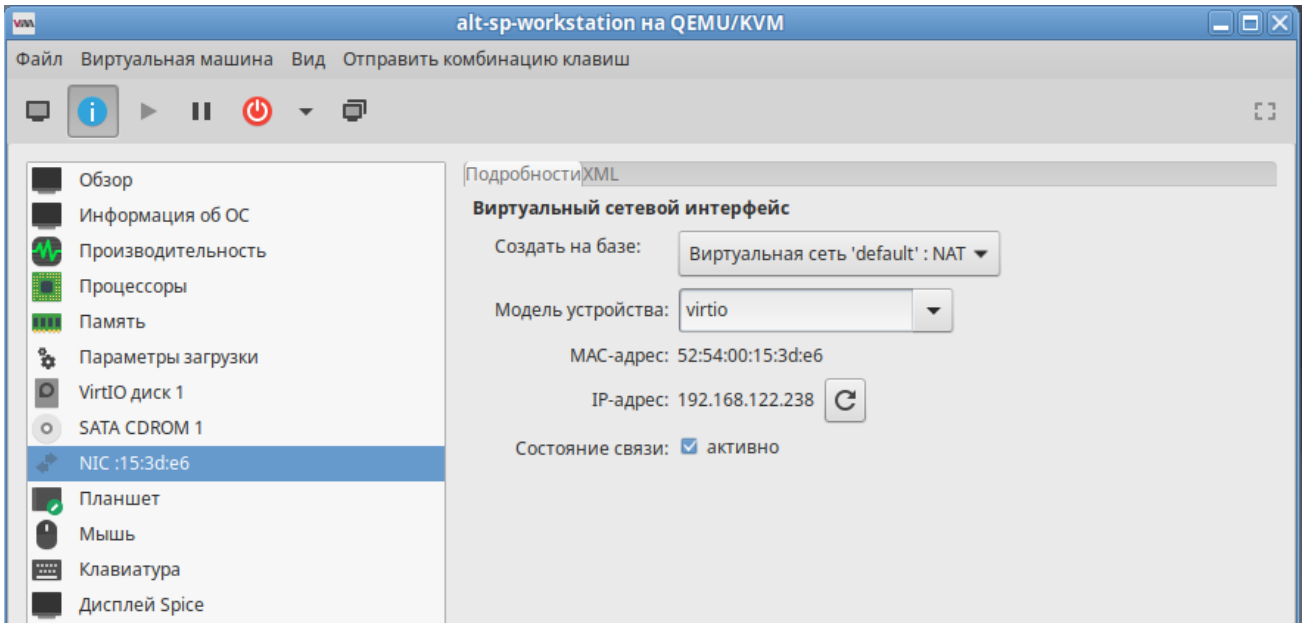

Рис. 12 – Вкладка «Сеть»

2.5.1.4. Мониторинг состояния

С помощью менеджера ВМ 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. 13).

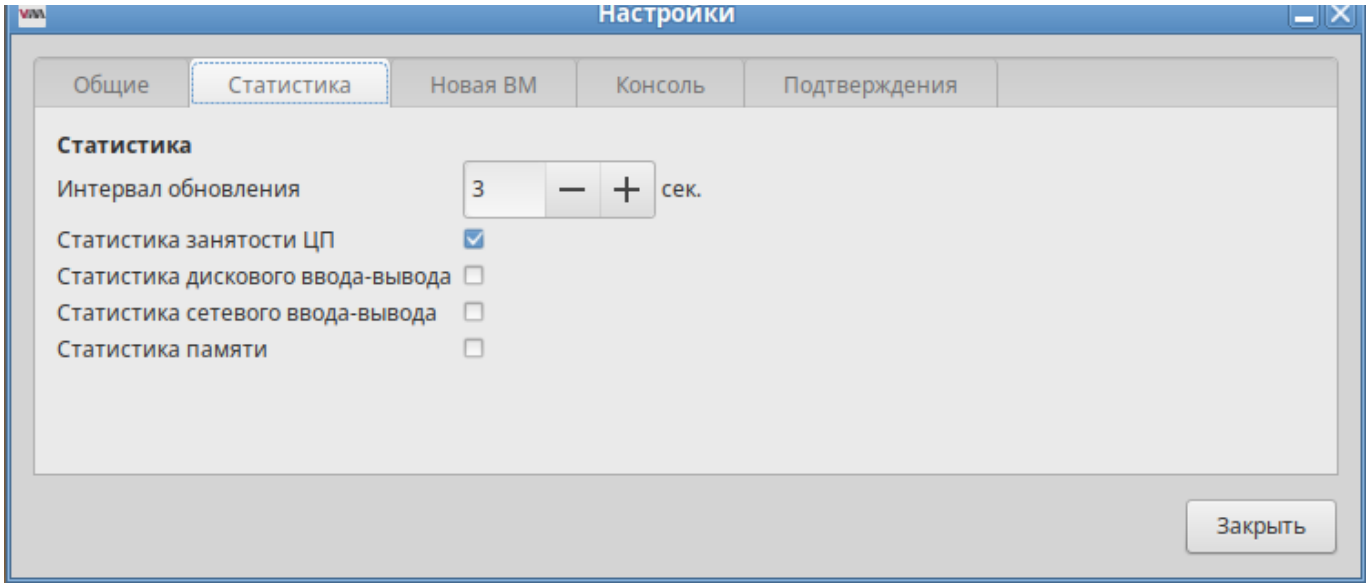

Рис. 13 – Вкладка «Статистика»

Во вкладке «Консоль» (рис. 14) можно выбрать, как открывать консоль, и указать устройство ввода.

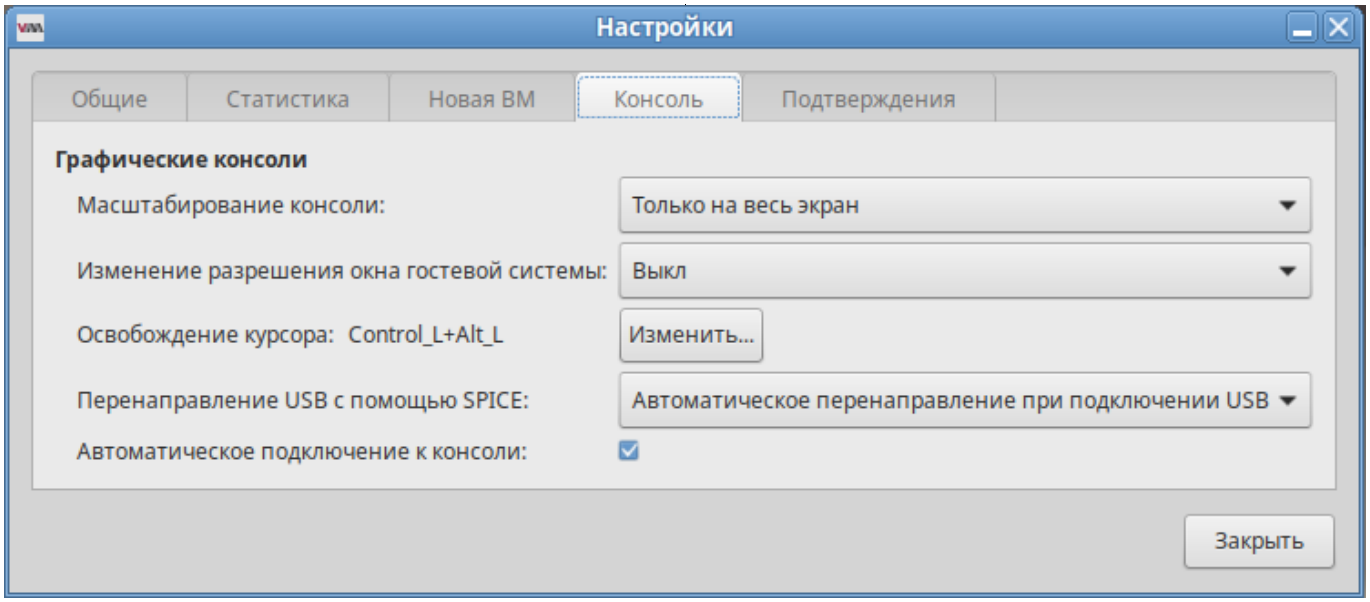

Рис. 14 – Вкладка «Консоль»

2.5.2. 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 из 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.

2.5.2.1. Управление виртуальными сетями в командной строке

Команда просмотра списка виртуальных сетей:

```
# virsh net-list
Имя       Статус         Автозапуск          Persistent
----------------------------------------------------------------------------

default   активен             no                  yes
```

Просмотр информации для заданной виртуальной сети:

```
# virsh net-dumpxml <Имя сети>
```

Пример вывода этой команды (в формате XML):

```
# virsh net-dumpxml vnet1
<network connections='1'>
     <name>default</name>
     <uuid>54fdc7a0-b143-4307-a2f4-a9f9d997cb1b</uuid>
```

```
    <forward mode='nat'>
        <nat>
            <port start='1024' end='65535'/>
        </nat>
    </forward>
    <bridge name='virbr0' stp='on' delay='0'/>
    <mac address='52:54:00:3e:12:c7'/>
    <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>
```

Другие 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-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-start имя_неактивной_сети` – запуск неактивной сети;
- `virsh net-undefine имя_неактивной_сети` – удаление определения неактивной сети.

```
# virsh net-list --all
Имя       Статус         Автозапуск     Persistent
----------------------------------------------------------------------
default   не активен          no         yes
# virsh net-start default
# virsh net-list --all
Имя       Статус       Автозапуск      Persistent
----------------------------------------------------------------------
default   активен      no              yes
```

2.5.2.2. Управление виртуальными сетями в менеджере ВМ

В менеджере ВМ 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. 15);
2) в открывшемся окне перейти на вкладку «Виртуальные сети» (рис. 16);
3) доступные виртуальные сети будут перечислены в левой части окна. Чтобы редактировать настройки сети, необходимо выбрать сеть из списка доступных.

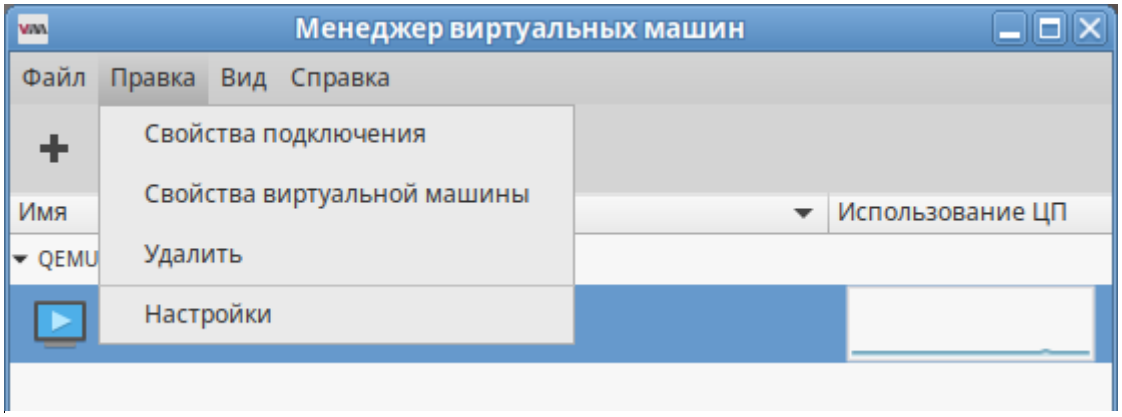

Рис. 15 – Меню «Правка»

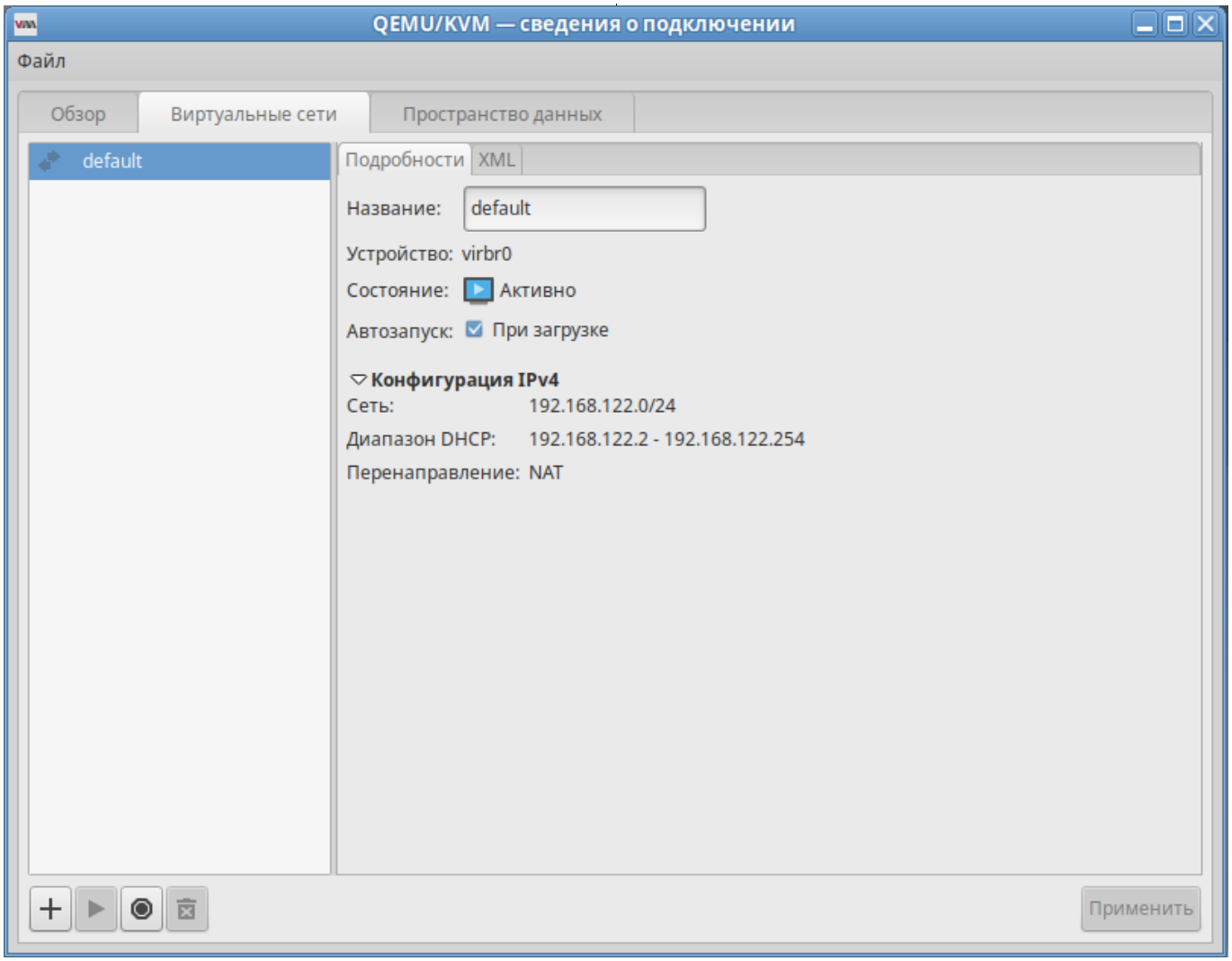

Рис. 16 – Окно параметров виртуальной сети

Для добавления новой виртуальной сети следует нажать на кнопку «Добавить сеть»  (рис. 16), расположенную в нижнем левом углу диалогового окна «Свойства соединения». В открывшемся окне (рис. 17) 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 на кнопку «Готово».

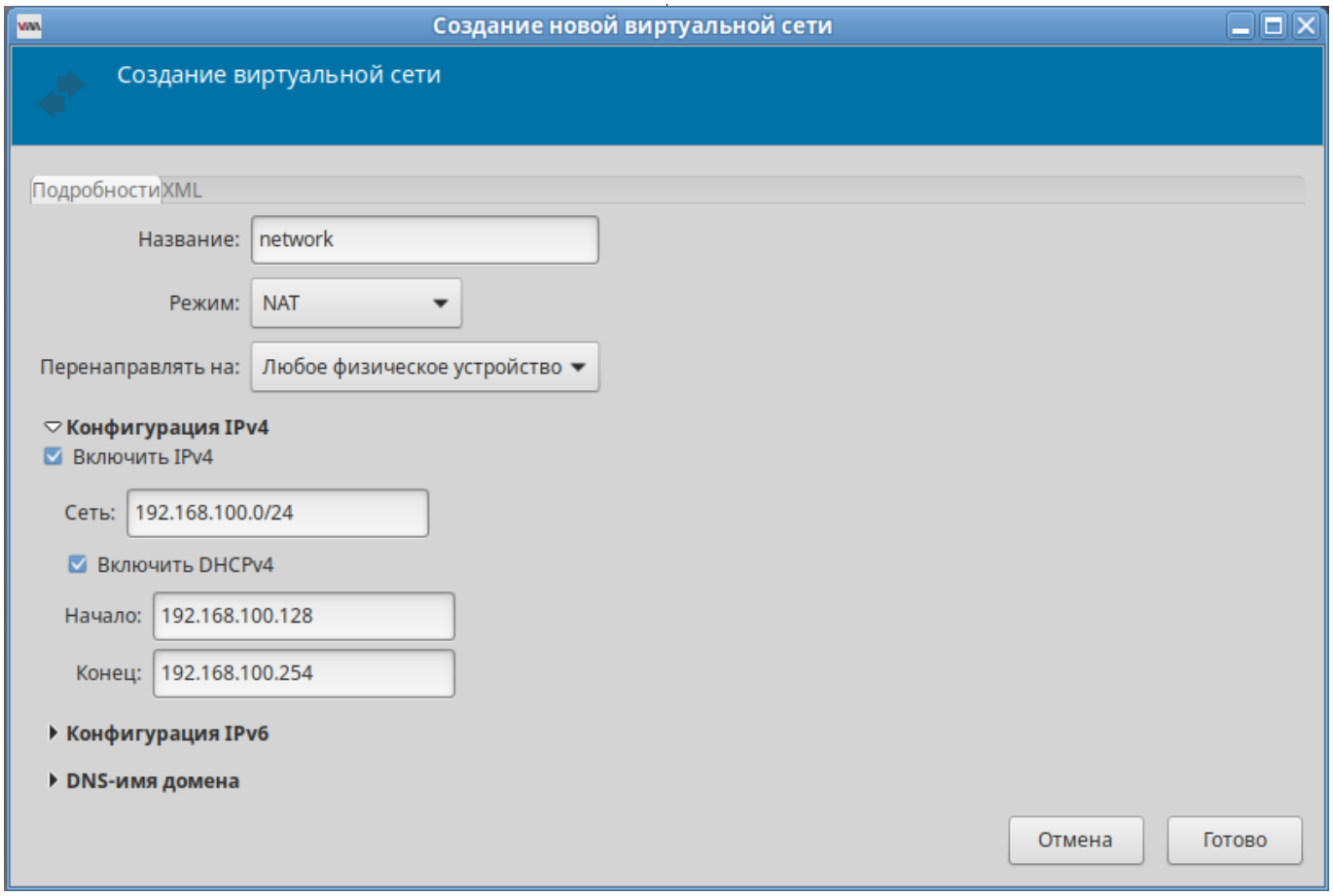

Рис. 17 – Создание новой виртуальной сети

2.5.3. Управление хранилищами

API-интерфейс libvirt обеспечивает удобную абстракцию для размещения образов ВМ и файловых систем, который 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 ВМ.

По умолчанию команды на базе libvirt используют в качестве исходного пула хранилищ для каталога файловой системы каталог `/var/lib/libvirt/images` на хосте виртуализации. Новый пул хранилищ можно с легкостью создать с помощью команды `virsh pool-create-as`.

Образ диска – это снимок данных диска ВМ, 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.

2.5.3.1. Управление хранилищами в командной строке

Новый пул хранилищ можно создать с помощью команды `virsh pool-create-as`. Например, следующая команда демонстрирует обязательные аргументы, которые необходимо указать при создании пула хранилищ на основе NFS (netfs):

```
# virsh pool-create-as NFS-POOL netfs \
--source-host 192.168.88.180 \
--source-path /export/storage \
--target /var/lib/libvirt/images/NFS-POOL
```

Первый аргумент (NFS-POOL) идентифицирует имя нового пула хранилищ, второй аргумент идентифицирует тип создаваемого пула хранилищ.

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.

Примечание. Для возможности монтирования NFS хранилища необходимо запустить службы `rpcbind` и `nfslock`:

```
# systemctl start rpcbind
# systemctl start nfslock
```

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
images     работает    yes       yes        125,43 GiB  16,87 GiB     108,56 GiB
NFS-POOL   работает    no        no         125,43 GiB  4,03 GiB      121,40 GiB
```

В выводе команды видно, что опция «Автозапуск» («Autostart») для пула хранилищ 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 (файл с расширением xml) имеет такое же имя, как у пула хранилищ, с которым он ассоциирован.

Чтобы создать файл XML-описания для сформированного в ручном режиме пула хранилищ, 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 стандартный поток вывода, поэтому необходимо перенаправить выводимую ей информацию в соответствующий файл.

Например, следующая команда создаст файл XML-описания для созданного ранее пула хранилищ NFS-POOL:

```
# virsh pool-dumpxml NFS-POOL > /etc/libvirt/storage/NFS-POOL.xml
```

Чтобы задать для пула хранилищ опцию «Автозапуск» («Autostart»), можно воспользоваться командой `virsh pool-autostart`:

```
# virsh pool-autostart NFS-POOL
ошибка: не удалось назначить автозапуск для пула NFS-POOL
ошибка: внутренняя ошибка: пул не включает файл конфигурации
```

Однако после перезагрузки системы, хранилище NFS-POOL становится постоянным и его можно добавить в автозапуск:

```
# reboot
# virsh pool-list --all --details
```

Имя	Состояние	Автозапуск	Постоянный	Размер	Распределение	Доступно
images	работает	yes	yes	125,43 GiB	22,46 GiB	102,97 GiB
NFS-POOL	не активен	no	yes	-	-	-

```
# virsh pool-autostart NFS-POOL
Добавлена метка автоматического запуска пула NFS-POOL
```

Маркировка пула хранилищ как «Autostart» 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).

2.5.3.2. Настройка пулов хранилищ в менеджере ВМ

Для настройки пулов хранилищ с помощью virt-manager необходимо:

1) в меню «Правка» выбрать «Свойства подключения» (рис. 18);

2) в открывшемся окне перейти на вкладку «Пространство данных» (рис. 19).

Для добавления пула следует нажать на кнопку «Добавить пул»  , расположенную в нижнем левом углу диалогового окна «Свойства соединения» (см. рис. 19). В открывшемся окне (рис. 20) следует выбрать тип пула, на втором шаге (рис. 21) задаются параметры пула.

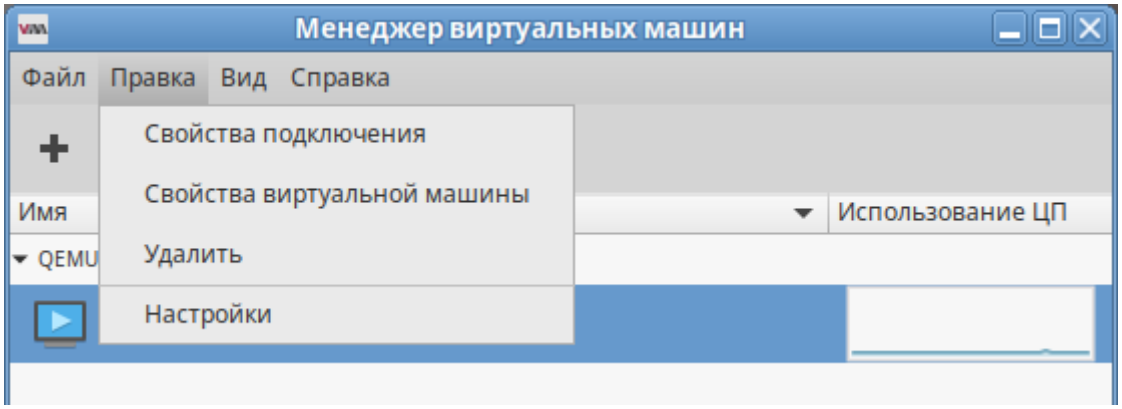

Рис. 18 – Меню «Правка»

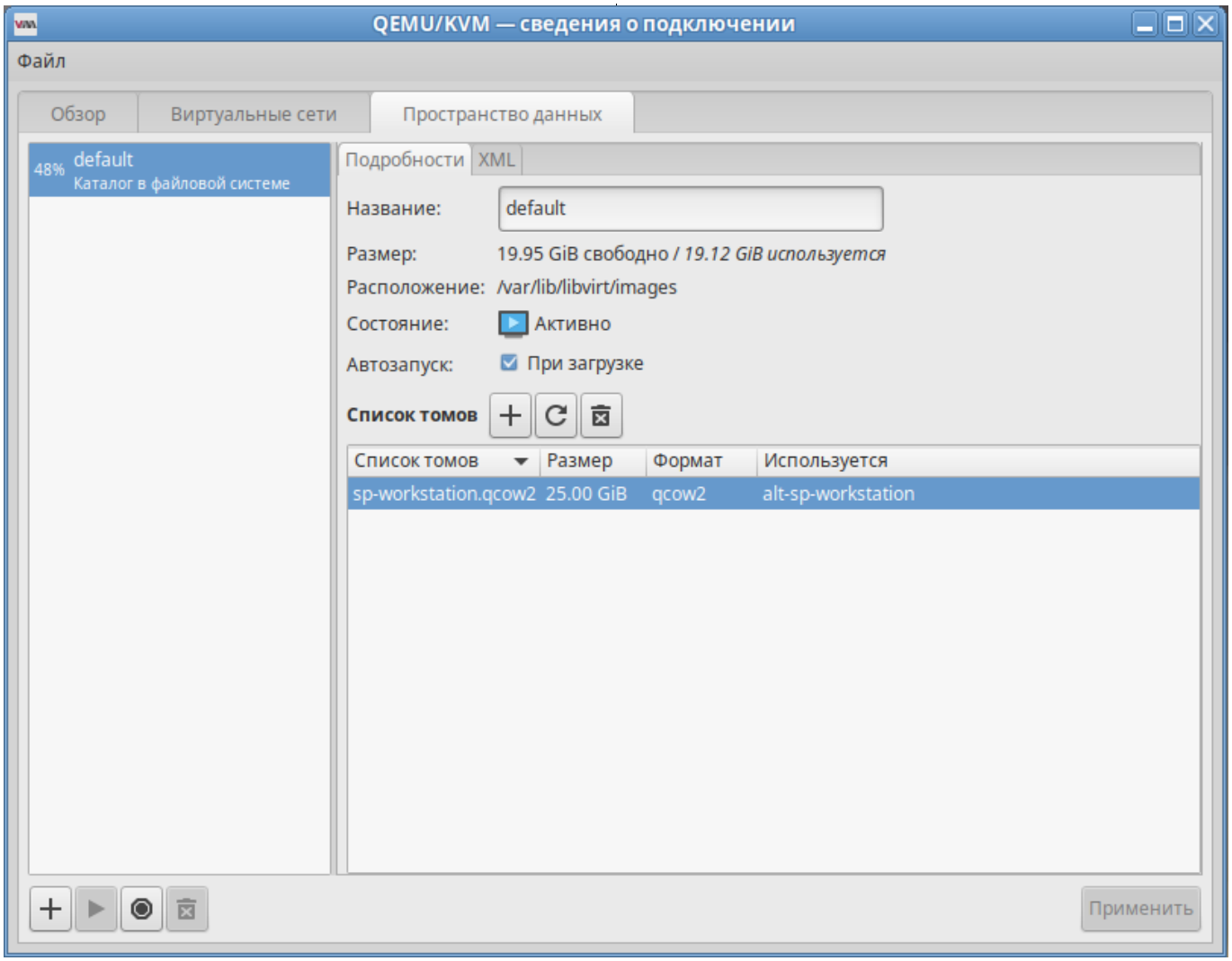

Рис. 19 – Вкладка «Пространство данных»

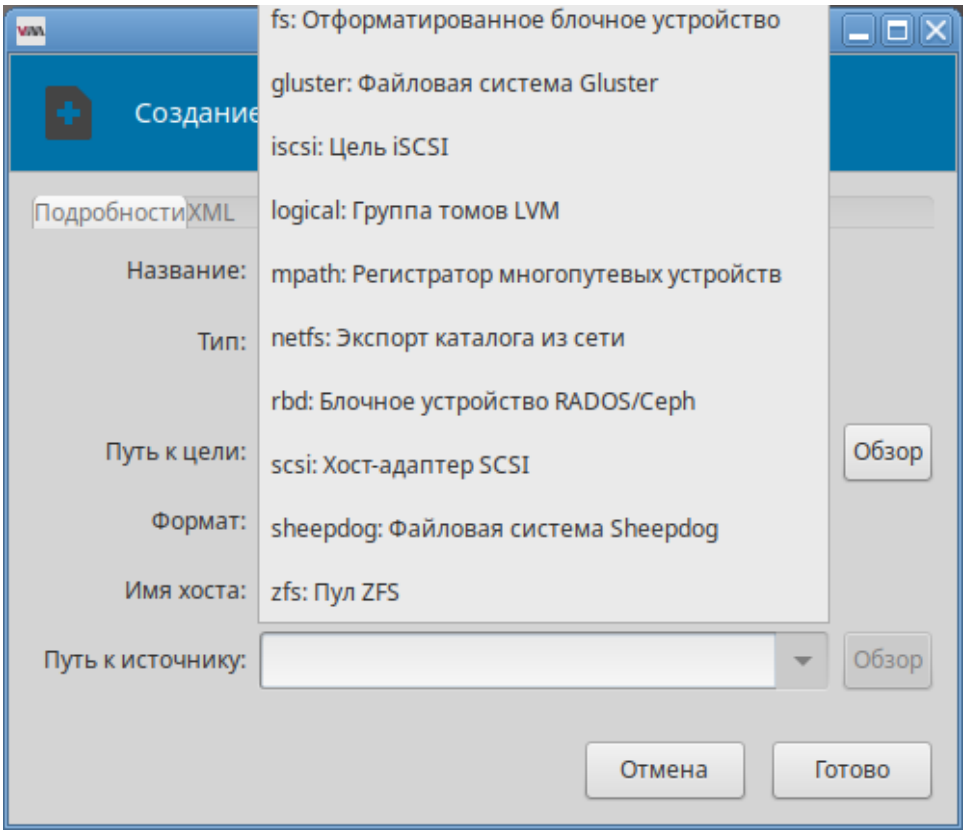

Рис. 20 – Создание пула хранения. Выбор типа пула

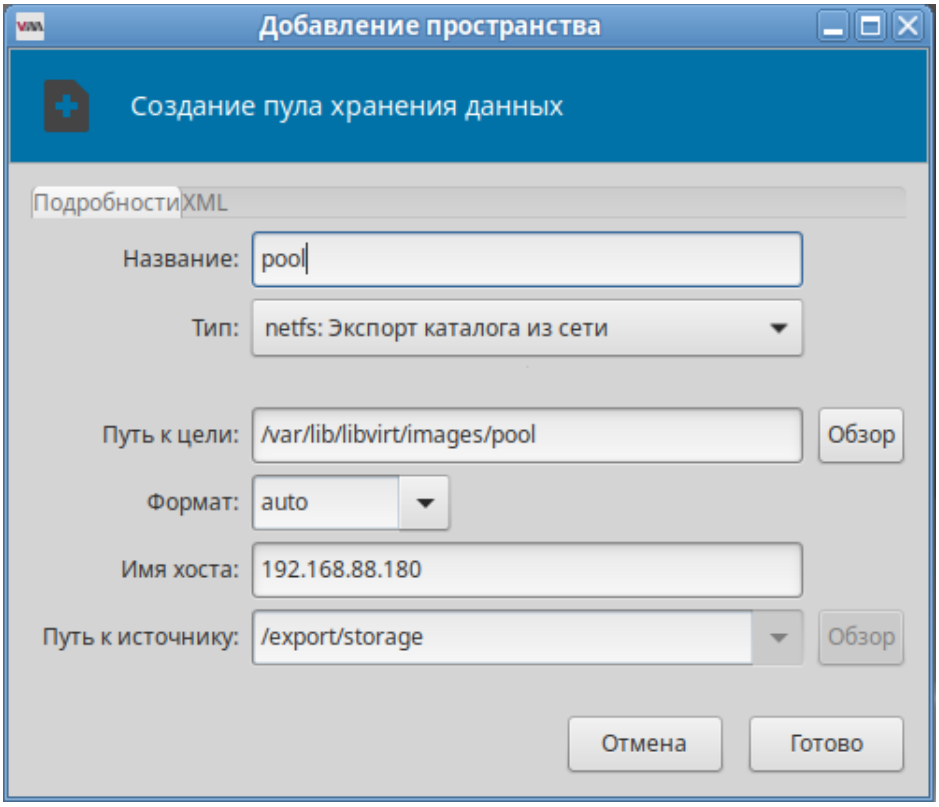

Рис. 21 – Создание пула хранения. Ввод параметров

2.6. Запуск и управление функционированием ВМ

2.6.1. 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 -c qemu:///system destroy alt-server
# virsh -c qemu:///system undefine alt-server
```

2.6.2. Управление состоянием ВМ в менеджере ВМ

Для запуска ВМ в менеджере ВМ virt-manager, необходимо выбрать ВМ из списка и нажать на кнопку «Включить ВМ» (рис. 22).

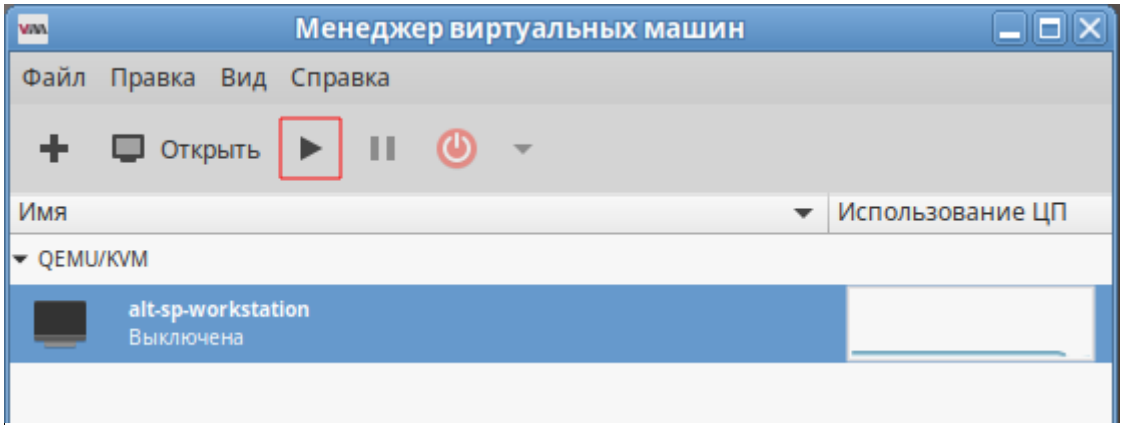

Рис. 22 – Включение ВМ

Для управления запущенной ВМ используются соответствующие кнопки панели инструментов virt-manager (рис. 23).

Управлять состоянием ВМ можно также выбрав соответствующий пункт в контекстном меню ВМ (рис. 24).

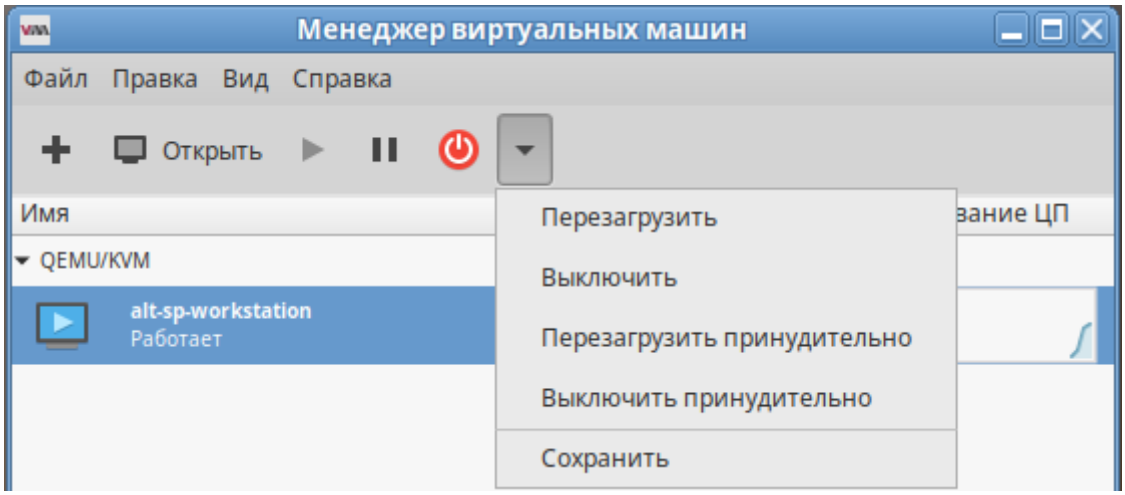

Рис. 23 – Кнопки управления состоянием ВМ

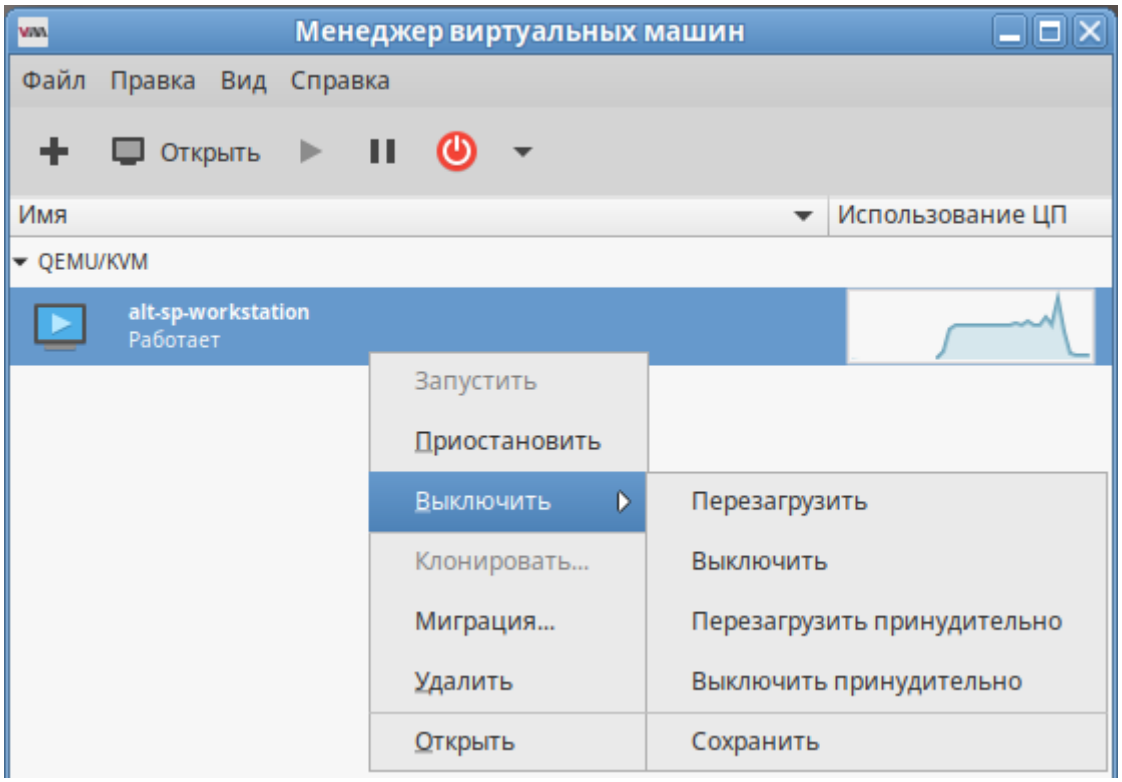

Рис. 24 – Контекстное меню ВМ

2.7. 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. Чтобы миграция была по-настоящему «живой» с незаметным простоем необходимо использовать общее хранилище.

2.7.1. 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, например, alt10.1 на узел 192.168.88.190 с помощью утилиты virsh, необходимо выполнить следующие действия:

1) убедиться, что ВМ запущена:

```
# virsh list
 ID    Имя          Статус
----------------------------------------------------
 1     alt10.1      работает
```

2) выполнить команду, чтобы начать перенос ВМ на узел 192.168.88.190 (после ввода команды будет запрошен пароль пользователя root системы назначения):

```
# virsh migrate --live alt10.1 qemu+ssh://192.168.88.190/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 – выполнить на узле назначения команду:

```
# virsh list
 ID    Имя          Статус
----------------------------------------------------
 1     alt10.1      работает
```

П р и м е ч а н и е . Для того, чтобы миграция ВМ между узлами выполнялась, узлы должны разрешать имена машин друг-друга. Например, на первом узле (Имя машины: libvirt-server-1, ip-адрес: 192.168.88.185) в /etc/hosts добавить запись:

```
192.168.88.190 libvirt-server-2
```

на втором узле (Имя машины: libvirt-server-2, ip-адрес: 192.168.88.190) в /etc/hosts добавить запись:

```
192.168.88.185 libvirt-server-1
```

2.7.2. Миграция с помощью virt-manager

Менеджер ВМ 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. 25) выбрать пункт «Миграция»;
3) в открывшемся окне (рис. 26) выбрать конечный узел и нажать на кнопку «Миграция».

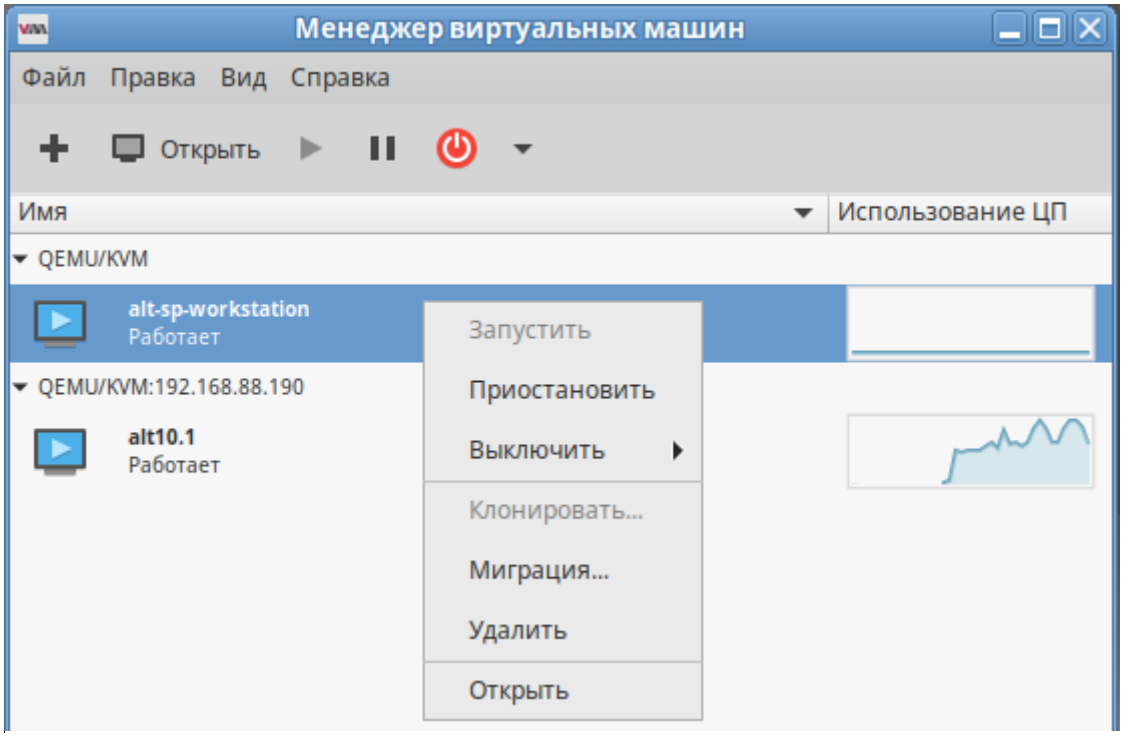

Рис. 25 – Пункт «Миграция» в контекстном меню ВМ

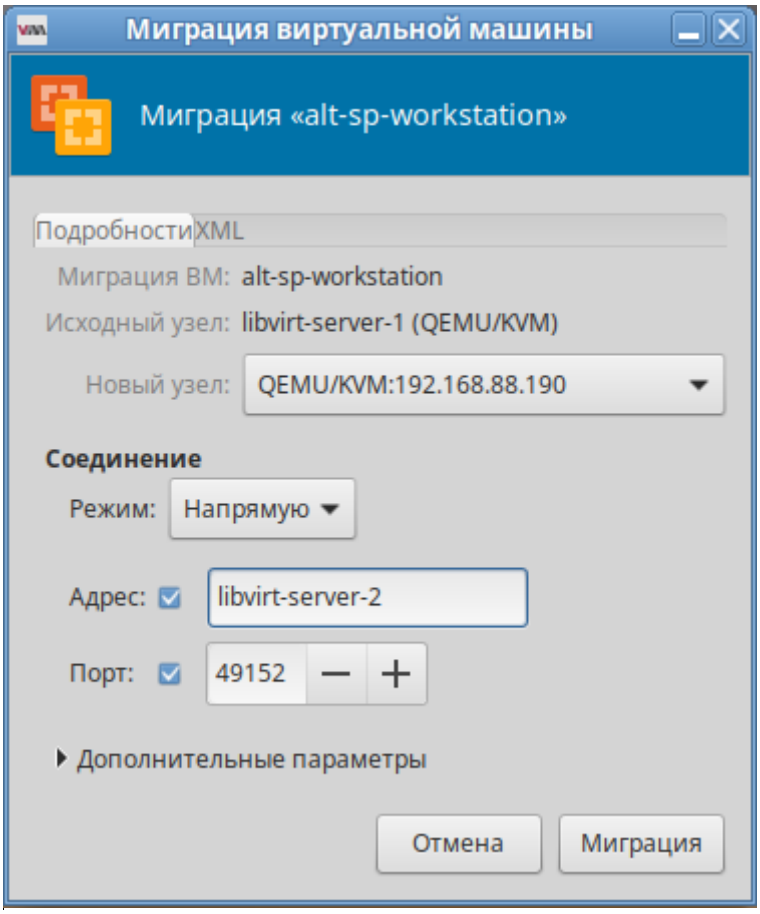

Рис. 26 – Миграция ВМ

При этом конфигурационный файл перемещаемой машины не переходит на новый узел, поэтому при ее выключении 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 \
alt10.1 qemu+ssh://192.168.88.190/system
```

2.8. Снимки машины

Примечание. Снимок (snapshot) текущего состояния машины можно создать только если виртуальный жесткий диск в формате `*.qcow2`.

2.8.1. 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 28nov2024
Снимок домена 28nov2024 создан
```

где:

- `alt-server` – имя ВМ;
- `28nov2024` – 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 05dec2024 --diskspec vda,file=/var/lib/libvirt/images/sn1.qcow2 --disk-only --atomic
Снимок домена 05dec2024 создан
```

Просмотр существующих снимков для домена `alt-server`:

```
# virsh snapshot-list --domain alt-server
 Имя            Время создания                   Статус
------------------------------------------------------------------------------
28nov2024        2024-11-28 08:50:05    +0200    running
05dec2024        2024-12-05 13:14:11    +0200    disk-snapshot
```

Восстановить ВМ из снимка:

```
# virsh snapshot-revert --domain alt-server --snapshotname 28nov2024 --running
```

Удалить снимок:

```
# virsh snapshot-delete --domain alt-server --snapshotname 28nov2024
```

2.8.2. Управления снимками ВМ virt-manager

Для управления снимками ВМ в менеджере ВМ virt-manager, необходимо:

1) в главном окне менеджера выбрать ВМ;
2) нажать на кнопку «Открыть»;
3) в открывшемся окне нажать на кнопку «Управление снимками»  (рис. 27). Появится окно управления снимками ВМ.

Для создания нового снимка следует нажать на кнопку «Создать новый снимок»  , расположенную в нижнем левом углу окна управления снимками ВМ. В открывшемся окне (рис. 28) следует указать название снимка и нажать на кнопку «Готово».

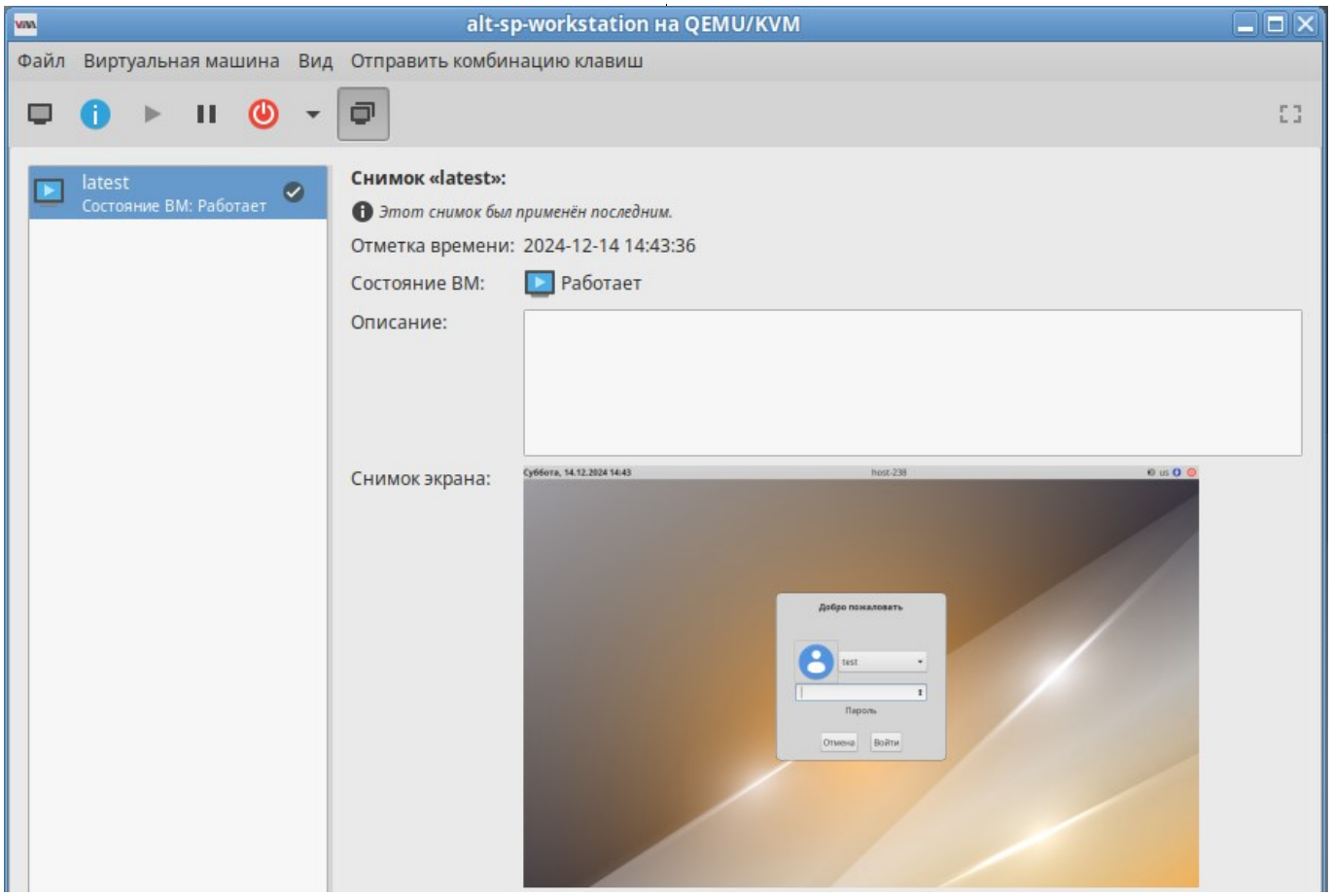

Рис. 27 – Управление снимками ВМ

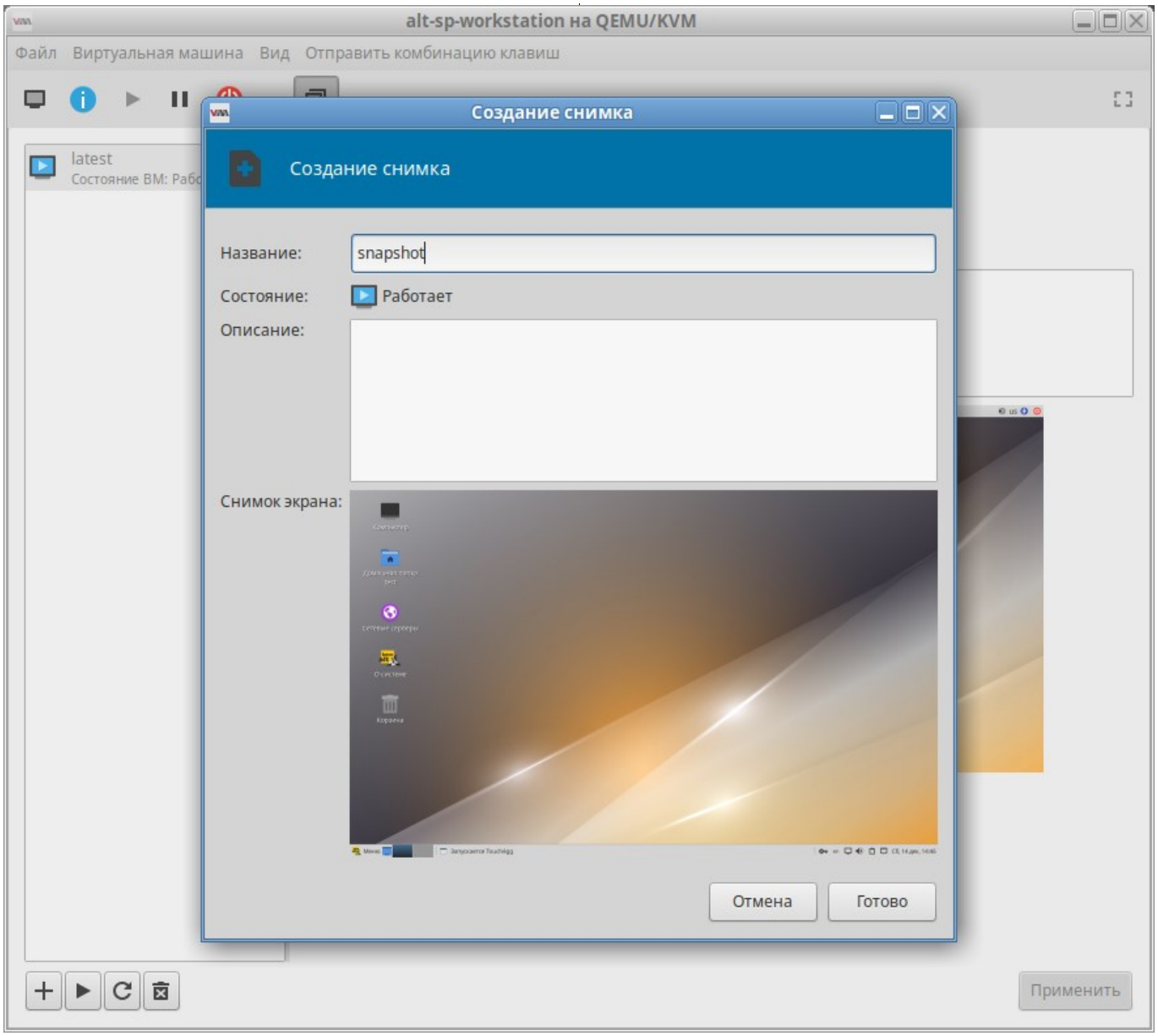

Рис. 28 – Создание снимка

Для того чтобы восстановить ВМ из снимка или удалить снимок, следует воспользоваться контекстным меню снимка (рис. 29).

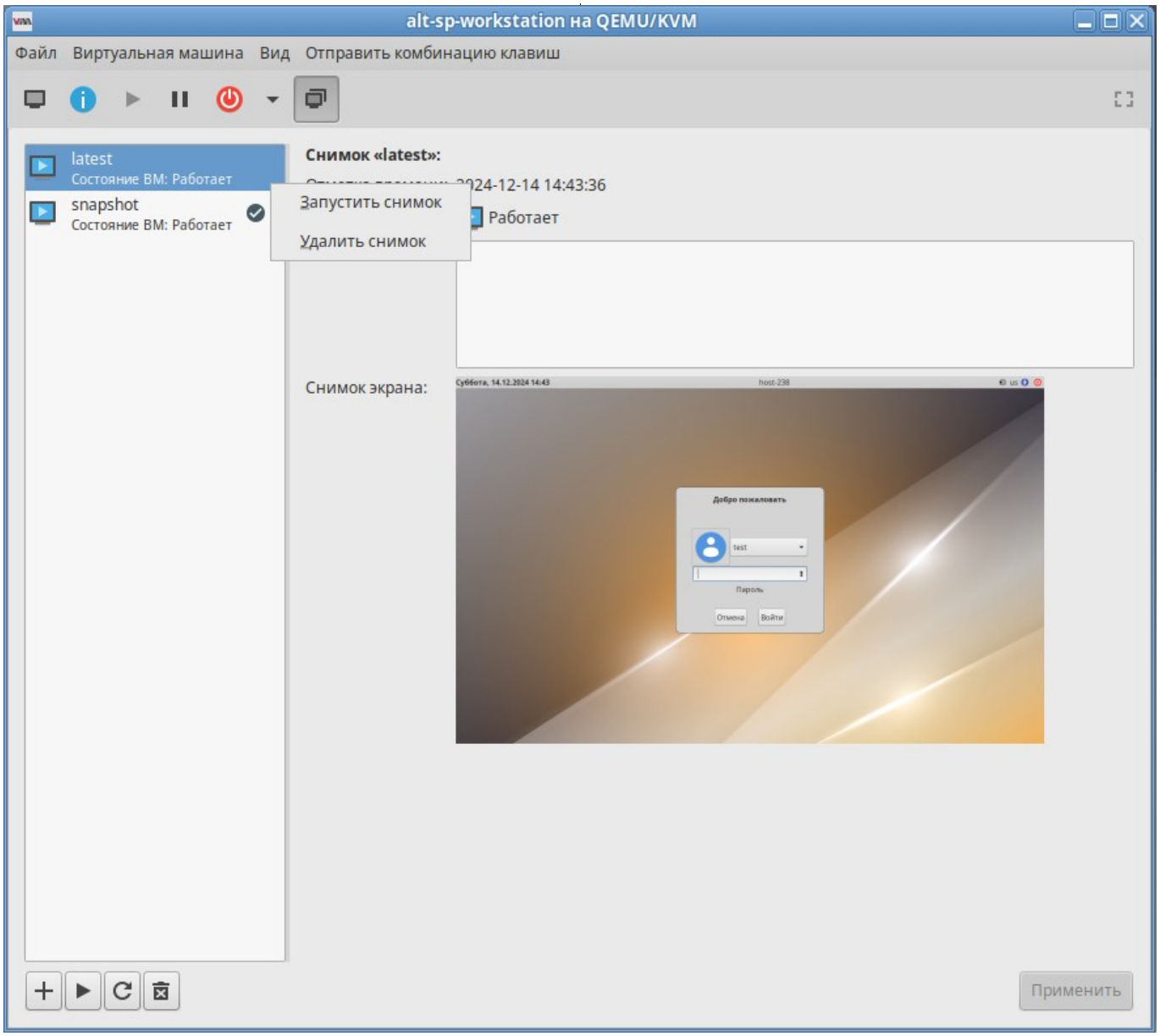

Рис. 29 – Контекстное меню снимка

2.9. 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`.

Например, действие `"Manage local virtualized systems"` в файле `org.libvirt.unix.policy`:

```
<action id="org.libvirt.unix.manage">
     <description>Manage local virtualized systems</description>
     <message>System policy prevents management of local
virtualized systems</message>
     <defaults>
          <allow_any>auth_admin_keep</allow_any>
          <allow_inactive>auth_admin_keep</allow_inactive>
          <allow_active>auth_admin_keep</allow_active>
     </defaults>
</action>
```

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`

Libvirt применяет контроль доступа ко всем основным типам объектов в его API. В таблице 4 приведены объекты, со своими наборами разрешений.

Т а б л и ц а 4 – Типы разрешений к объектам libvirt

Объект	Разрешения	Описание
Connect	detect-storage-pools	Обнаружение хранилищ
	getattr	Подключение
	interface-transaction	Операции с интерфейсом
	pm-control	Управление питанием
	read	Просмотр
	search-domains	Список доменов
	search-interfaces	Список интерфейсов
	search-networks	Список сетей
	search-node-devices	Список узлов
	search-nwfilters	Список сетевых фильтров
	search-secrets	Список секретов
	search-storage-pools	Список хранилищ
	write	Изменение
Domain	block-read	Чтение блочного устройства домена
	mem-read	Просмотр памяти
	migrate	Миграция
	open-device	Устройства
	open-graphics	Графика
	open-namespace	Пространство имен
	inject-nmi	Немаскируемое прерывание (NMI)
	pm-control	Управление питанием
	read	Чтение
	read-secure	Защищенный просмотр
	reset	Перезагрузить
	save	Сохранить
	screenshot	Получить screenshot
	send-input	Отправить ввод
	send-signal	Отправить сигнал
	set-password	Установка паролей
	set-time	Установка времени
	snapshot	Снимок
	start	Запуск
	stop	Останов
	suspend	Приостановка
	write	Изменение
	hibernate	Спящий режим
	init-control	Инициализация-контроль

Продолжение таблицы 4

Объект	Разрешения	Описание
Interface	delete	Удаление
	getattr	Доступ
	read	Просмотр
	save	Сохранение
	start	Запуск
	stop	Останов
	write	Изменение
Network	delete	Удаление
	getattr	Доступ
	read	Просмотр
	save	Сохранение
	start	Запуск
	stop	Останов
	write	Изменение
Node-Device	detach	Отсоединение
	getattr	Доступ
	read	Просмотр
	start	Запуск
	stop	Останов
	write	Изменение
NWFilter	delete	Удаление
	getattr	Доступ
	read	Просмотр
	save	Сохранение
	write	Изменение
Secret	delete	Удаление
	getattr	Доступ
	read	Просмотр
	read-secure	Безопасный просмотр
	save	Сохранить
	write	Записать
Storage-Pool	delete	Удаление
	refresh	Обновление
	format	Форматирование
	getattr	Доступ
	read	Просмотр
	save	Сохранение
	search-storage-vols	Список томов
	start	Запуск
	stop	Останов

Окончание таблицы 4

Объект	Разрешения	Описание
	write	Изменение
	create	Создание
Storage-Vol	data-read	Просмотр данных
	data-write	Запись данных
	delete	Удаление
	format	Форматирование
	getattr	Доступ
	read	Просмотр
	resize	Изменение размера

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 (таблица 5).

Т а б л и ц а 5 – Атрибуты объектов libvirt

Объект	Атрибут	Описание
Connect	`connect_driver`	Название подключения
Domain	`connect_driver`	Название подключения
	`domain_name`	Название домена, уникально для локального хоста
	`domain_uuid`	UUID домена, уникально
Interface	`connect_driver`	Название подключения
	`interface_name`	Название сетевого интерфейса, уникально для локального хоста
	`interface_macaddr`	MAC-адрес сетевого интерфейса, не уникальный глобально
Network	`connect_driver`	Название подключения
	`network_name`	Название сети, уникально для локального хоста
	`network_uuid`	UUID сети, уникально
NodeDevice	`connect_driver`	Название подключения
	`node_device_name`	Название устройства, уникально для локального хоста
NWFilter	`connect_driver`	Название подключения
	`nwfilter_name`	Название сетевого фильтра, уникально для локального хоста
	`nwfilter_uuid`	UUID сетевого фильтра, уникально

Окончание таблицы 5

Объект	Атрибут	Описание
Secret	connect_driver	Название подключения
	secret_uuid	UUID уникально
	secret_usage_volume	Название тома
	secret_usage_ceph	Название Ceph сервера
	secret_usage_target	Название iSCSI
	secret_usage_name	Название TLS
StoragePool	connect_driver	Название подключения
	pool_name	Название хранилища, уникально для локального хоста
	pool_uuid	UUID хранилища, уникально

По умолчанию для запуска virt-manager требуется ввод пароля пользователя с идентификатором root. Для того, чтобы virt-manager запускался от нужного пользователя (в примере – test), необходимо добавить этого пользователя в группу vmusers и перелогиниться, при необходимости перезапустить libvirt, polkit, nscd:

```
# gpasswd -a test vmusers
# service libvirtd restart
# service polkit restart
# service nscd restart
```

Добавить в файл /etc/libvirt/libvirtd.conf строку:

```
access_drivers = [ "polkit" ]
```

Перезапустить libvirt:

```
# service libvirtd restart
```

2.9.1. Пример тонкой настройки

Есть две ВМ: 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) выйти и снова войти в ОС.

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 остановке ВМ:

```
if (action.id.match("org.libvirt.api.domain.stop") ) {
        polkit.log("action=" + action);
        polkit.log("subject=" + subject);
        return polkit.Result.YES;
}
```

2.10. Регистрация событий

2.10.1. 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 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 ВМ под KVM хранятся в `/var/log/libvirt/qemu/`. В этом каталоге libvirt хранит журнал для каждой ВМ. Например, для машины с названием `alt-server` журнал будет находиться по адресу:

```
/var/log/libvirt/qemu/alt-server.log
```

2.10.2. Регистрация событий запуска (завершения) работы компонентов виртуальной инфраструктуры

В каталоге `/var/log/libvirt/qemu/` KVM хранит журнал для каждой ВМ. Например, для машины с названием `alt1` журнал будет находиться по адресу `/var/log/libvirt/qemu/alt1.log`.

В этот журнал попадают записи вида:

```
qemu: terminating on signal 15 from pid 118813
2016-12-16 14:39:41.045+0000: shutting down
qemu: terminating on signal 15 from pid 2056
2016-12-19 14:01:55.917+0000: shutting down
2016-12-19 14:02:09.841+0000: starting up libvirt version: 1.3.2,
package: alt1, qemu version: 2.5.0, hostname: vb.office.alt1.ru
```

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; }
          }
```

Запись в журнал при изменении ВМ:

```
if (action.id.match("org.libvirt.api.domain.write") ) {
            polkit.log("action=" + action);
            polkit.log("subject=" + subject);
```

```
            return polkit.Result.YES; }
        }
```

2.10.3. Регистрация входа (выхода) субъектов доступа в/из гипервизор(а)

Регистрацию событий входа (выхода) субъектов доступа в/из гипервизор(а) можно настроить с помощью правил polkit.

При любом действии с подключениями и хранилищем записывать в журнал:

```
if (action.id.match("org.libvirt.unix. ") ) {
            polkit.log("action=" + action);
            polkit.log("subject=" + subject);
            return polkit.Result.YES; }
        }
```

2.10.4. Регистрация событий входа (выхода) субъектов доступа в/из гостевых ОС

Регистрация событий входа (выхода) субъектов доступа в/из гостевых ОС не производится, так как зависит от ОС, выполняемых в ВМ.

2.10.5. Регистрация изменения прав доступа к файлам-образам ВМ

Регистрация событий изменения прав доступа к файлам-образам ВМ можно настроить с помощью `audit`.

Файлы libvirt:

- `/var/lib/libvirt/boot/` – ISO-образы для установки гостевых систем;
- `/var/lib/libvirt/images/` – образы жестких дисков гостевых систем;
- `/etc/libvirt/` – каталог с файлами конфигурации.

Под учетной записью администратора включить контроль над объектом `/var/lib/libvirt/images/`:

```
# auditctl -w /var/lib/libvirt/images/ -p wa
```

В журнале контроля будут фиксироваться записи, свидетельствующие о регистрации факта создания, просмотра и изменения файлов.

3. PODMAN

3.1. Установка podsec-пакетов

Для работоспособности podman необходимо установить в систему следующие пакеты:

```
# apt-get install -y podsec podsec-k8s-rbac podsec-k8s podsec-inotify
```

3.2. Выделение IP-адресов

Для корректной работы регистратора и веб-сервера подписей необходимо выделить отдельный IP-адрес на одном из сетевых интерфейсов. Это может быть доступный из локальной сети адрес другого интерфейса или дополнительный статический адрес на интерфейсе локальной сети. Основной адрес, используемый для доступа к регистратору и веб-серверу подписей, должны быть статическим и не изменяться после перезагрузки узла.

Например, структура файлов каталога `/etc/net/ifaces/enp1s0` описания интерфейса enp1s0 с адресом 192.168.10.70 для регистратора и веб-сервера подписей:

- options:

```
BOOTPROTO=static
TYPE=eth
CONFIG_WIRELESS=no
SYSTEMD_BOOTPROTO=static
CONFIG_IPV4=yes
DISABLED=no
NM_CONTROLLED=no
SYSTEMD_CONTROLLED=no
```

- ipv4address:

```
192.168.10.70/24
```

- ipv4route:

```
default via 192.168.10.1
```

- resolv.conf:

```
nameserver 192.168.10.1
```

Интерфейс для данных параметров выглядит следующим образом:

```
    # ip a show dev enp1s0
2: enp1s0:  mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 52:54:00:db:e1:57 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.70/24 brd 192.168.122.255 scope global enp1s0
       valid_lft forever preferred_lft forever

    ...
```

3.3. Настройка политики контейнеризации

Для настройки политики контейнеризации необходимо запустить команду:

```
# podsec-create-policy 192.168.10.70 # ip-адрес_регистратора и веб-сервера подписей
Добавление привязки доменов registry.local sigstore.local к IP-адресу 192.168.10.70
Создание группы podman
Инициализация каталога /var/sigstore/ и подкаталогов хранения открытых ключей и
подписей образов
Создание каталога и подкаталогов /var/sigstore/
Создание группы podman_dev
Создание с сохранением предыдущих файла политик /etc/containers/policy.json
Создание с сохранением предыдущих файл /etc/containers/registries.d/default.yaml
описания доступа к открытым ключам подписантов
Добавление insecure-доступа к регистратору registry.local в файле
/etc/containers/registries.conf
Настройка использования образа registry.local/k8s-c10f2/pause:3.9 при запуска pod'ов
в podman (podman pod init)
```

После настройки политики следующие файлы должны содержать указанный текст:

- файл `/etc/hosts` должен содержать строку:

  ```
  ...
  192.168.122.70 registry.local sigstore.local
  ```

- файл `/etc/containers/policy.json`, являющийся symlink к файлу `/etc/containers/policy_YYYY-MM-DD_HH:mm:SS` должен иметь содержимое (запрет доступа по всем ресурсам):

  ```
  {
    "default": [
      {
        "type": "reject"
      }
    ],
    "transports": {
      "docker": {}
    }
  }
  ```

- файл `/etc/containers/registries.d/default.yaml`, являющийся symlink к файлу `/etc/containers/registries.d/default_YYYY-MM-DD_HH:mm:SS` должен иметь содержимое (URLs доступа к серверу подписей):

```
default-docker:
  lookaside: http://sigstore.local:81/sigstore/
  sigstore: http://sigstore.local:81/sigstore/
```

3.4. Создание сервисов регистратора и веб-сервера подписей

Выполнить поднятие сервиса регистратора и веб-сервера подписей командой:

```
# podsec-create-services
Synchronizing state of nginx.service with SysV service script
with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
Created symlink /etc/systemd/system/multi-
user.target.wants/nginx.service →
/lib/systemd/system/nginx.service.
registry
Created symlink /etc/systemd/system/multi-
user.target.wants/docker-registry.service →
/lib/systemd/system/docker-registry.service.
```

Проверьте функционирование сервисов:

```
# netstat -nlpt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
PID/Program name
...
tcp        0      0 0.0.0.0:81                  0.0.0.0:*                   LISTEN
14996/nginx -g daemo
...
tcp        0      0 :::80                       :::*                        LISTEN
15044/docker-registr
...
```

3.5. Создание пользователя разработчика образов контейнеров

Для создания пользователя разработчик образов контейнеров (imagemaker) выполните команду:

```
# podsec-create-imagemakeruser imagemaker

Проверка. Является ли текущий сервер сервером, поддерживающий регистратор
(registry.local) и сервер подписи образов (sigstore.local)
Введите пароль пользователя imagemaker - разработчика образов контейнеров
passwd: updating all authentication tokens for user imagemaker.
You can now choose the new password or passphrase.
A valid password should be a mix of upper and lower case letters, digits, and
other characters.  You can use a password containing at least 7 characters
from all of these classes, or a password containing at least 8 characters
from just 3 of these 4 classes.
An upper case letter that begins the password and a digit that ends it do not
count towards the number of character classes used.
```

```
A passphrase should be of at least 3 words, 11 to 72 characters long, and
contain enough different characters.
Alternatively, if no one else can see your terminal now, you can pick this as
your password: "scant9Idle+Quick".
Enter new password:
Re-type new password:
passwd: all authentication tokens updated successfully.
gpg (GnuPG) 2.2.33; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
gpg: создан каталог '/home/imagemaker/.gnupg'
gpg: создан щит с ключами '/home/imagemaker/.gnupg/pubring.kbx'
Выберите тип ключа:
   (1) RSA и RSA (по умолчанию)
   (2) DSA и Elgamal
   (3) DSA (только для подписи)
   (4) RSA (только для подписи)
  (14) Имеющийся на карте ключ
Ваш выбор?
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072)
Запрошенный размер ключа - 3072 бит
Выберите срок действия ключа.
         0 = не ограничен
      <n>  = срок действия ключа - n дней
      <n>w = срок действия ключа - n недель
      <n>m = срок действия ключа - n месяцев
      <n>y = срок действия ключа - n лет
Срок действия ключа? (0)
Срок действия ключа не ограничен
Все верно? (y/N) y
GnuPG должен составить идентификатор пользователя для идентификации ключа.
Ваше полное имя: ImageMaker
Адрес электронной почты: test@ivk.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
    "ImageMaker <test@ivk.ru>"
Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? O
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: /home/imagemaker/.gnupg/trustdb.gpg: создана таблица доверия
gpg: ключ E7DAAAB099C1C8A8 помечен как абсолютно доверенный
gpg: создан каталог '/home/imagemaker/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/home/imagemaker/.gnupg/openpgp-
revocs.d/001E6716FA9EE98C3CAF6E0EE7DAAAB099C1C8A8.rev'.
открытый и секретный ключи созданы и подписаны.
pub   rsa3072 2023-05-24 [SC]
      001E6716FA9EE98C3CAF6E0EE7DAAAB099C1C8A8
uid                      ImageMaker <test@ivk.ru>
sub   rsa3072 2023-05-24 [E]
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0  достоверных:   1  подписанных:   0  доверие: 0-, 0q, 0n, 0m,
0f, 1u
[root@arm1 podsec-1.0.0]#
```

Файл `/etc/containers/policy.json`, должен изменить symlink на другой файл `/etc/containers/policy_YYYY-MM-DD_HH:mm:SS` с содержимым (разрешение доступа к регистратору `registry.local` с открытым ключом пользователя `imagemaker`):

```
{
  "default": [
    {
      "type": "reject"
    }
  ],
  "transports": {
    "docker": {
      "registry.local": [
        {
          "type": "signedBy",
          "keyType": "GPGKeys",
          "keyPath": "/var/sigstore/keys/imagemaker.pgp"
        }
      ]
    }
  }
}
```

Должен появится каталог `/var/sigstore/` со следующей структурой:

```
├── index.html
├── keys
│   ├── imagemaker.pgp
│   └── policy.json
└── sigstore
```

Проверьте доступ к этому каталогу через http:

```
# curl -s  http://sigstore.local:81/keys/ | jq
[
  {
    "name": "imagemaker.pgp",
    "type": "file",
    "mtime": "Tue, 23 May 2023 05:43:59 GMT",
    "size": 2436
  },
  {
    "name": "policy.json",
    "type": "file",
    "mtime": "Tue, 23 May 2023 05:43:25 GMT",
    "size": 276
  }
]
```

3.6. Создание пользователя информационной системы

Для создания пользователя информационной системы (poduser) выполните команду:

```
# podsec-create-podmanusers poduser
```

```
Введите пароль пользователя 'poduser':
passwd: updating all authentication tokens for user poduser.
You can now choose the new password or passphrase.
A valid password should be a mix of upper and lower case letters, digits, and
other characters.  You can use a password containing at least 7 characters
from all of these classes, or a password containing at least 8 characters
from just 3 of these 4 classes.
An upper case letter that begins the password and a digit that ends it do not
count towards the number of character classes used.
A passphrase should be of at least 3 words, 11 to 72 characters long, and
contain enough different characters.
Alternatively, if no one else can see your terminal now, you can pick this as
your password: "brook=Molten7Taboo".
Enter new password:
Re-type new password:
passwd: all authentication tokens updated successfully.
```

3.7. Проверка работы podman в rootless-режиме

Выполнить авторизацию в роли пользователя от имени учетной записи пользователя imagemaker, скачать и загрузить в локальное хранилище образ ALTLinux:

```
$ podman pull --tls-verify registry.altlinux.org/alt/alt
Trying to pull registry.altlinux.org/alt/alt:latest...
Getting image source signatures
Copying blob 9ab3f3206235 done
Copying blob cedd146c7d35 done
Copying config ff2762c6c8 done
Writing manifest to image destination
Storing signatures
ff2762c6c8cc9468e0651364e4347aa5c769d78541406209e9ab74717f29e641
$ podman tag registry.altlinux.org/alt/alt registry.local/alt/alt
$ podman push --tls-verify=false  --sign-by='<test@ivk.ru>'
registry.local/alt/alt
Getting image source signatures
Copying blob 60bdc4ff8a54 done
Copying blob 9a03b2bc42d8 done
Copying config ff2762c6c8 done
Writing manifest to image destination
Creating signature: Signing image using simple signing
Storing signatures
```

Выполнить запуск образа контейнера:

```
podman run -it registry.local/alt/alt:latest bash
```

4. KUBERNETES

4.1. Подготовка

Подготовьте несколько машин (nodes), одна из которых будет мастером.

Системные требования:

- 2 Гбайт ОЗУ или больше;
- 2 ядра процессора или больше;
- все машины должны быть доступны по сети друг для друга;
- своп должен быть выключен;
- на них должны быть установлены следующие пакеты:

```
# apt-get install kubernetes-kubeadm kubernetes-kubelet cri-tools kubernetes-crio
```

- и запущены сервисы kube-proxy:

```
# systemctl enable --now kubelet kube-proxy crio
```

4.2. Разворачивание кластера

1) На мастере нужно запустить команду для запуска кластера:

```
# kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=SystemVerification
```

где:

- `--pod-network-cidr=10.244.0.0/16` – внутренняя (разворачиваемая Kubernetes) сеть, данное значение рекомендуется оставить для правильной работы `Flannel`.

В конце вывода будет строка вида:

```
kubeadm join <ip адрес>:<порт> --token <токен> --discovery-token-ca-cert-hash sha256:<хэш>
```

2) Настройка kubernetes для работы от пользователя:

- создать каталог `~/.kube`:

```
$ mkdir ~/.kube
```

- от администратора скопировать файл конфигурации:

```
# cp /etc/kubernetes/admin.conf ~<пользователь>/.kube/config
```

- изменить владельца файла конфигурации:

```
# chown <пользователь>: ~<пользователь>/.kube/config
```

3) Подключить к мастеру все остальные ноды:

```
# kubeadm join <ip адрес>:<порт> --token <токен> --discovery-token-ca-cert-hash sha256:<хэш> --ignore-preflight-errors=SystemVerification
```

Проверить наличие нод можно так:

```
$ kubectl get nodes -o wide
```

Вывод примерно следующий:

```
NAME    STATUS     ROLES       AGE  VERSION    INTERNAL-IP    EXTERNAL-IP    OS-IMAGE KERNEL-VERSION       CONTAINER-RUNTIME
master NotReady control-plane  19m  v.24.8   172.16.0.8    <none>        ALT Regular 5.15.90-std-def-alt1  cri-o://1.24.3
node   NotReady <none>         19m  v1.24.8  172.16.0.9    <none>        ALT Regular 5.15.90-std-def-alt1  cri-o://1.24.3
```

Обратите внисание, что ноды находятся в состоянии NotReady. Они перейдут в состояние Ready после настройки сети.

4) Далее следует развернуть сеть.

Для Flannel с использованием образов на базе ALT манифесты находятся по адресу `https://gitea.basealt.ru/alt/flannel-manifests/<платформа>`.

Где `<платформа>` (на дату 25.10.2024): sisyphus, p10, c10f2.

Каждый каталог платформы содержит каталоги:

```
├── <major>
│   ├── <minor>
│   │   └── <patch>
│   │       └── kube-flannel.yml
...
└── latest
    └── kube-flannel.yml
```

Выберите для платформы (например, c10f2) установленной версии kubernetes необходимую версию flannel (например, 0.25.7) и для разворачивания наберите:

```
$ kubectl apply -f https://gitea.basealt.ru/alt/flannel-manifests/raw/branch/main/c10f2/0/25/7/kube-flannel.yml
```

Проверить работу – выполнить команду:

```
$ kubectl get pods --namespace kube-system
```

Пример корректного вывода:

```
          NAME                     READY   STATUS    RESTARTS   AGE
coredns-78fcdf6894-6trk7           1/1     Running   0          2h
coredns-78fcdf6894-nwt5l           1/1     Running   0          2h
etcd-k8s                           1/1     Running   0          2h
kube-apiserver-k8s                 1/1     Running   0          2h
kube-controller-manager-k8s        1/1     Running   0          2h
kube-flannel-ds-894bt              1/1     Running   0          2h
kube-flannel-ds-kbngw              1/1     Running   0          2h
kube-flannel-ds-n7h45              1/1     Running   0          2h
kube-flannel-ds-tz2rc              1/1     Running   0          2h
kube-proxy-6f4lm                   1/1     Running   0          2h
kube-proxy-f92js                   1/1     Running   0          2h
kube-proxy-qkh54                   1/1     Running   0          2h
kube-proxy-szvlt                   1/1     Running   0          2h
kube-scheduler-k8s                 1/1     Running   0          2h
```

Следует обратить внимание, что `coredns` находятся в состоянии `Running`. Количество `kube-flannel` и `kube-proxy` зависит от общего числа нод (в данном случае их четыре).

П р и м е ч а н и е . При разворачивании узла кластера podsec устанавливается последняя версия flannel, доступная на регистраторе (например, 0.25.7). Для этой версии podsec выбирает deployment-файл, входящий в состав пакета из каталога `/etc/podsec/u7s/manifests/kube-flannel/`. На текущий момент каталог имеет следующую структуру:

```
└── 0
    ├── 19
    │   ├── 0
    …
    └── 25
        ├── 0
        └── 1
```

Так как версии образа flannel постоянно обновляются, может сложится ситуация, когда нужный deployment-манифест в каталоге пакета не окажется (как в случае выше для версии 0.25.7). В этом случае перед запуском kubeadm init необходимо указать последнюю существующую в каталоге версию deployment-манифеста. В нашем случае:

```
export U7S_FLANNEL_TAG=v0.25.1
```

4.3. Тестовый запуск nginx

1) Создать Deployment:

```
$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
```

2) Затем создать сервис, с помощью которого можно получить доступ к приложению из внешней сети.

Сохраните в файл `nginx-service.yaml` следующую конфигурацию:

```
apiVersion: v1
kind: Service
metadata:
   name: nginx
   labels:
        app: nginx
spec:
   type: NodePort
   ports:
   - port: 80
        targetPort: 80
   selector:
         app: nginx
```

3) Запустите новый сервис:

```
$ kubectl apply -f nginx-service.yaml
```

4) Чтобы узнать порт nginx выполните команду:

```
$ kubectl get svc nginx
```

Пример вывода:

```
NAME     TYPE      CLUSTER-IP    PORT(S)  EXTERNAL-IP   AGE
nginx   NodePort  10.108.199.141  <none>  80:32336/TCP   4h
```

5) Проверьте работу:

```
$ curl <IP-адрес>:<порт>
```

где:

- `IP-адрес` – это адрес любой из нод;
- `порт` – это порт сервиса, полученный с помощью предыдущей команды. Если использовать данные из примеров, то возможная команда:

  ```
  curl 10.10.3.120:32336
  ```

4.4. Настройка kubernetes для работы в rootless режиме

Запуск Kubernetes в режиме rootless обеспечивает запуск Podов без системных root-привелегий в рамках user namespace системного пользователя u7s-admin. Работа в этом режиме практически не требует никаких модификаций, но обеспечивает повышенный уровень защищенности kubernetes, так как клиентские приложения даже при использовании уязвимостей не могут получить права пользователя root и нарушить работу узла.

Запуск kubernetes версии 1.26.3 и старше в режиме rootless обеспечивает пакет podsec-k8s версии 1.0.5 или выше.

Для разворачивания rootless kubernetes требуется версия ядра ОС 5.15 и выше.

4.4.1. podsec-k8s – быстрый старт

4.4.1.1. Установка master-узла

4.4.1.1.1. Инициализация master-узла

Для запуска kubernetes в режиме rootless установите пакет podsec-k8s версии 1.0.5 или выше:

```
# apt-get install podsec-k8s
```

Измените переменную PATH:

```
export PATH=/usr/libexec/podsec/u7s/bin/:$PATH
```

В каталоге `/usr/libexec/podsec/u7s/bin/` находятся программы, обеспечивающие работу kubernetes в rootless-режиме.

Для разворачивания master-узла запустите команду:

```
kubeadm init
```

Примечание. По умолчанию уровень отладки устанавливается в `0`. Если необходимо увеличить уровень отладки, укажите перед подкомандой `init` флаг `-v n`, где `n` принимает значения от 0 до 9-ти.

После:

- генерации сертификатов в каталоге `/etc/kubernetes/pki`;
- загрузки образов;
- генерации conf-файлов в каталоге `/etc/kubernetes/manifests/`, `/etc/kubernetes/manifests/etcd/`;

- запуска сервиса kubelet и Pod’ов системных kubernetes-образов инициализируется kubernet-кластер из одного узла.

По окончании скрипт выводит строки подключения master (Control Plane) и worker-узлов:

```
    You can now join any number of control-plane nodes by copying certificate authorities
    and service account keys on each node and then running the following as root:

    kubeadm join xxx.xxx.xxx.xxx:6443 --token ... --discovery-token-ca-cert-hash sha256:.. --control-plane

    Then you can join any number of worker nodes by running the following on each as root:

    kubeadm join xxx.xxx.xxx.xxx:6443 --token ... --discovery-token-ca-cert-hash sha256:...
```

4.4.1.1.2. Запуск сетевого маршрутизатора для контейенеров kube-flannel

П р и м е ч а н и е . Для версии podsec 1.0.8 и выше этот шаг выполнять не надо – он выполняется во время `kubeadm init`.

Для перевода узла в состояние Ready, запуска coredns Pod’ов запустите flannel.

На master-узле под пользоваталем root выполните команду:

```
    # kubectl apply -f /etc/kubernetes/manifests/kube-flannel.yml
    Connected to the local host. Press ^] three times within 1s to exit session.
    [INFO] Entering RootlessKit namespaces: OK
    namespace/kube-flannel created
    clusterrole.rbac.authorization.k8s.io/flannel created
    clusterrolebinding.rbac.authorization.k8s.io/flannel created
    serviceaccount/flannel created
    configmap/kube-flannel-cfg created
    daemonset.apps/kube-flannel-ds created
    Connection to the local host terminated.
```

После завершения скрипта в течение минуты настраиваются сервисы master-узла кластера. По ее истечении проверьте работу usernetes (rootless kuber).

4.4.1.1.3. Проверка работы master-узла

На master-узле выполните команды:

```
# kubectl get daemonsets.apps -A
```

На рис. 30, рис. 31, рис. 32 и далее в подразделе приведены примеры вывода работы команд, зависят от версии ядра ОС и модулей kubernetes.

```
# kubectl get daemonsets.apps -A
NAMESPACE      NAME              DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-flannel   kube-flannel-ds   1         1         1       1            1           <none>
102s
kube-system    kube-proxy        1         1         1       1            1           kubernetes.io/os=linux   8h
```

Рис. 30

Число `READY` каждого `daemonset` должно быть равно числу `DESIRED` и должно быть равно числу узлов кластера.

```
# kubectl get nodes -o wide
```

```
# kubectl get nodes -o wide
NAME       STATUS   ROLES           AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION
CONTAINER-RUNTIME
<host>     Ready    control-plane   16m   v1.26.3   10.96.0.1     <none>        ALT SP Server 11100-01   5.15.105-
un-def-alt1   cri-o://1.26.2
```

Рис. 31

Проверьте работу usernetes (rootless kuber) (рис. 32):

```
# kubectl get all -A
```

```
# kubectl get all -A
NAMESPACE     NAME                                      READY   STATUS    RESTARTS   AGE
kube-system   pod/coredns-c7df5cd6c-5pkkm               1/1     Running   0          19m
kube-system   pod/coredns-c7df5cd6c-cm6vf               1/1     Running   0          19m
kube-system   pod/etcd-host-212                         1/1     Running   0          19m
kube-system   pod/kube-apiserver-host-212               1/1     Running   0          19m
kube-system   pod/kube-controller-manager-host-212      1/1     Running   0          19m
kube-system   pod/kube-proxy-lqf9c                      1/1     Running   0          19m
kube-system   pod/kube-scheduler-host-212               1/1     Running   0          19m

NAMESPACE     NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
default       service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP                  19m
kube-system   service/kube-dns     ClusterIP   10.96.0.10   <none>        53/UDP,53/TCP,9153/TCP   19m

NAMESPACE     NAME                          DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR
AGE
kube-system   daemonset.apps/kube-proxy   1         1         1       1            1
kubernetes.io/os=linux   19m

NAMESPACE     NAME                        READY   UP-TO-DATE   AVAILABLE   AGE
kube-system   deployment.apps/coredns   2/2     2            2           19m

NAMESPACE     NAME                                   DESIRED   CURRENT   READY   AGE
kube-system   replicaset.apps/coredns-c7df5cd6c   2         2         2       19m
```

Рис. 32

Состояние всех Pod’ов должны быть в 1/1.

Проверьте состояние дерева rootless-процессов (рис. 33):

```
# pstree
```

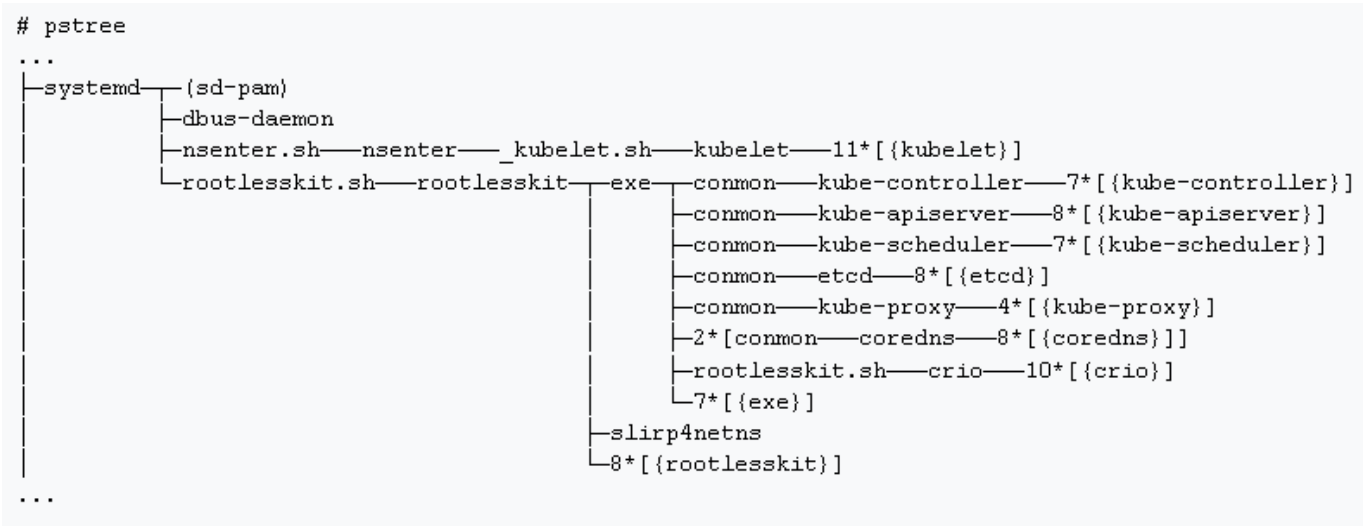

Рис. 33

Процесс kubelet запускается как сервис в user namespace процесса rootlesskit.

Все остальные процессы kube-controller, kube-apiserver, kube-scheduler, kube-proxy, etcd, coredns запускаются как контейнеры от соответствующих образов в user namespace процесса rootlesskit.

4.4.1.1.4. Обеспечение запуска обычных POD’ов на master-узле

По умолчанию на master-узле пользовательские Pods не запускаются. Чтобы снять это ограничение наберите команду:

```
# kubectl taint nodes <host> node-role.kubernetes.io/control-plane:NoSchedule-
```

Вывод команды:

```
node/<host> untainted
```

где `<host>` – имя master-узла, отображаемое в выводе команды:

```
# kubectl get nodes
```

4.4.1.2. Инициализация и подключение worker-узла

Установите пакет podsec-k8s:

```
apt-get install podsec-k8s
```

Измените переменную PATH:

```
export PATH=/usr/libexec/podsec/u7s/bin/:$PATH
```

4.4.1.2.1. Подключение worker-узлов

Скопируйте команду подключения worker-узла, полученную на этапе установки начального master-узла. Запустите ее:

```
kubeadm join xxx.xxx.xxx.xxx:6443 --token ... --discovery-token-ca-cert-hash sha256:...
```

Примечание. По умолчанию уровень отладки устанавливается в `0`. Если необходимо увеличить уровень отладки укажите перед подкомандой `init` флаг `-v n`. Где `n` принимает значения от 0 до 9-ти.

По окончании скрипт выводит текст:

```
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
```

4.4.1.2.2. Проверка состояния процессов

Проверьте состояние дерева rootless-процессов (рис. 34):

```
# pstree
...
├─systemd─┬─(sd-pam)
│         ├─dbus-daemon
│         ├─nsenter.sh──nsenter──_kubelet.sh──kubelet──10*[{kubelet}]
│         └─rootlesskit.sh──rootlesskit─┬─exe─┬─conmon──kube-proxy──4*[{kube-proxy}]
│                                       │     ├─rootlesskit.sh──crio──9*[{crio}]
│                                       │     └─6*[{exe}]
│                                       ├─slirp4netns
│                                       └─8*[{rootlesskit}]
...
```

Рис. 34

Процесс kubelet запускается как сервис в user namespace процесса rootlesskit.

Все остальные процессы kube-proxy, kube-flannel запускаются как контейнеры от соответствующих образов в user namespace процесса rootlesskit.

4.4.1.2.3. Проверка готовности master и worker узлов kubernets

Зайдите на master-узел и проверьте подключение worker-узла (рис. 35):

```
# kubectl get nodes -o wide
```

```
# kubectl get nodes -o wide
NAME       STATUS   ROLES           AGE     VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE                KERNEL-
VERSION         CONTAINER-RUNTIME
<host1>   Ready    control-plane   7h54m   v1.26.3   10.96.0.1     <none>  ALT   cri-o://1.26.2
<host2>   Ready    <none>          8m30s   v1.26.3   10.96.0.1     <none>  ALT   cri-o://1.26.2
...
```

Рис. 35 – Пример вывода

4.4.1.3. Инициализация и подключение дополнительных master-узлов

Установите пакет podsec-k8s:

```
apt-get install podsec-k8s
```

Измените переменную PATH:

```
export PATH=/usr/libexec/podsec/u7s/bin/:$PATH
```

4.4.1.3.1. Подключение master-узлов

Скопируйте команду подключения master-узла, полученную на этапе установки начального master-узла. Она отличается от команды подключения worker-узлов наличием дополнительных параметров `--control-plane`, `--certificate-key`.

Запустите ее:

```
kubeadm join xxx.xxx.xxx.xxx:6443 --token ...
    --discovery-token-ca-cert-hash sha256:...  --control-plane --certificate-key ...
```

П р и м е ч а н и е . По умолчанию уровень отладки устанавливается в `0`. Если необходимо увеличить уровень отладки укажите перед подкомандой `init` флаг `-v n`. Где `n` принимает значения от 0 до 9-ти.

По окончании скрипт выводит текст:

```
This node has joined the cluster and a new control plane instance was created:

* Certificate signing request was sent to apiserver and approval was received.
* The Kubelet was informed of the new secure connection details.
* Control plane label and taint were applied to the new node.
* The Kubernetes control plane instances scaled up.
* A new etcd member was added to the local/stacked etcd cluster.
```

4.4.1.3.2. Проверка состояния процессов

Проверьте состояние дерева процессов (рис. 36):

```
# pstree
```

```
# pstree
...
        ├─systemd─┬─(sd-pam)
        │         ├─dbus-daemon
        │         ├─kubelet.sh───nsenter_u7s───nsenter───_kubelet.sh───kubelet───11*[{kubelet}]
        │         └─rootlesskit.sh───rootlesskit─┬─exe─┬─conmon───kube-controller───4*[{kube-controller}]
        │                                        │     ├─conmon───kube-scheduler───8*[{kube-scheduler}]
        │                                        │     ├─conmon───etcd───9*[{etcd}]
        │                                        │     ├─conmon───kube-proxy───4*[{kube-proxy}]
        │                                        │     ├─conmon───kube-apiserver───8*[{kube-apiserver}]
        │                                        │     ├─rootlesskit.sh───crio───8*[{crio}]
        │                                        │     └─7*[{exe}]
        │                                        ├─slirp4netns
        │                                        └─8*[{rootlesskit}]
```

Рис. 36

Дерево rootless-процессов должно отличаться от дерева процессов основного master-узла отсутствием контейнеров coredns.

4.4.1.3.3. Проверка готовности master и worker узлов kubernets

На одном из master-узлов наберите команду (рис. 37):

```
# kubectl get nodes -o wide
```

```
# kubectl get nodes -o wide
NAME       STATUS   ROLES           AGE     VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE                KERNEL-
VERSION         CONTAINER-RUNTIME
<host1>   Ready    control-plane   7h54m   v1.26.3   10.96.0.1     <none>        ALT    cri-o://1.26.2
<host2>   Ready    <none>          8m30s   v1.26.3   10.96.0.1     <none>  ALT   cri-o://1.26.2
...
<hostN>   Ready    control-plane   55m     v1.26.3   10.96.122.<N> <none>        ALT      cri-o://1.26.2
...
```

Рис. 37 – Пример вывода

4.4.1.3.4. Использование REST-интерефейсов подключенных master-узлов

По умолчанию на подключенных master-узлах в файле `/etc/kubernetes/admin.conf` указан адрес API-интерфейса основного master-узла:

```
apiVersion: v1
clusters:
- cluster:
    ...
    server: https://<master1>:6443
...
```

Для балансировки нагрузки в файлах конфигурации `~user/.kube/config` есть смысл указать адреса API-интерфейсов дополнительных master-узлов:

```
apiVersion: v1
clusters:
- cluster:
    ...
    server: https://<masterN>:6443
...
```

4.4.1.4. Создание гетерогенных кластеров, миграция с rootfull-кластеров на rootless кластера

В рамках одного кластера могут функционировать как rootfull-узлы, так и rootless-узлы. Например, имеет смысл в рамках rootfull-кластера для повышения защищенности кластера подключать в качестве Worker rootless-узлы.

Перед подключением rootless-узлов необходимо выполнить определенные действия.

4.4.1.4.1. Запуск kube-proxy на rootless-узле в rootfull-кластере

Для запуска kube-proxy на rootless-узле в rootfull-кластере на ControlPlane узле:

- набрать команду редактирования ConfigMap kube-proxy:

```
kubectl -n kube-system edit Configmaps kube-proxy
```

- изменить в элементе `data.config.conf` значение переменной `conntrack.maxPerCore` с `null` на `0`;
- выйти из редактора.

4.4.1.4.2. Запуск ControlPlane узла на rootless-узле в rootfull-кластере

Для запуска ControlPlane на rootless-узле в rootfull-кластере на ControlPlane узле:

- набрать команду редактирования ConfigMap kubeadm-config:

```
kubectl -n kube-system edit  Configmaps kubeadm-config
```

- изменить в элементе `data.ClusterConfiguration` значение переменной `etcd.local.dataDir` с `/var/lib/etcd` на `/var/lib/podsec/u7s/etcd`;
- выйти из редактора.

4.4.1.5. Получение строки подключения узла к кластеру

4.4.1.5.1. Получение строки подключения Worker узла к кластеру

В случае, если команда строки подключения утеряна или срок сгенерированного сертификата истек можно сгенерировать новую строку подключения, выполнив команду:

```
joinCommand=$(/usr/bin/kubeadm token create --print-join-command)
```

и выполнить команду подключения:

```
export PATH=/usr/libexec/podsec/u7s/bin/:$PATH
$joinCommand
```

4.4.1.5.2. Получение строки подключения Control-plane узла к кластеру

В определенных случаях `kubeadm init` генерирует только строку подключения worker-узлов. Или срок действия сертификата для подключения истек.

В этом случае есть смысл перегенерировать сертификат:

```
cert=$(/usr/bin/kubeadm init phase upload-certs --upload-certs 2>/dev/null | tail -1)
```

строку подключения `control-plane` и worker-узлов к кластеру:

```
joinCommand=$(/usr/bin/kubeadm token create --print-join-command)
```

и выполнить команду подключения:

```
export PATH=/usr/libexec/podsec/u7s/bin/:$PATH
$joinCommand --control-plane --certificate-key $cert
```

4.4.1.6. Системный пользователь u7s-admin

Все контейнеры в rootless kubernetes, включая системные, работают от имени системного пользователя u7s-admin. Для мониторинга работы системы или запуска дополнительного функционала можете работать в системе от имени этого пользователя.

Для входа в терминальный режим этого пользователя достаточно в пользователе с правами root набрать команду:

```
# machinectl shell u7s-admin@ /bin/bash
```

или задав пароль пользователя:

```
# passwd u7s-admin
```

зайти в него через ssh.

Для входа в namespace пользователя наберите команду:

```
$ nsenter_u7s
#
```

В рамках своего namespace пользователь u7s-admin имеет права root, оставаясь в рамках системы с правами пользователя u7s-admin.

Наличие прав root позволяет использовать системные команды, требующие root-привелегии для работы с сетевым, файловым окружением (эти окружения отличаются от системных): ip, iptables, crictl, ...

С помощью команды crictl можно:

- посмотреть наличие образов в системном кэше;
- удалить, загрузить образы;
- посмотреть состояние контейнеров, pod'ов;
- и т. п.

Кроме этого namespace пользователя u7s-admin присутствуют файлы и каталоги, созданные в рамках данного namespace и отсутствующие в основной системе. Например, можно посмотреть логи контейнеров в каталоге `/var/log/pods` и т. п.

4.4.1.7. Особенности разворачивания приложений в rootless kubernetes

При использовании сервисов типа NodePort поднятые в рамках кластера порты в диапазоне 30000 – 32767 остаются в namespace пользователя u7s-admin. Для их проброса наружу необходимо в пользователе u7s-admin запустить команду:

```
$ nsenter_u7s rootlessctl add-ports 0.0.0.0:<port>:<port>/tcp
```

Сервисы типа NodePort из-за их небольшого диапазона и «нестабильности» портов при переносе решения в другой кластер довольно редко используются. Рекомендуется вместо них использовать сервисы типа ClusterIP с доступом к ним через Ingress-контроллеры.

4.4.2. Разворачивание rootless kubernetes кластера с балансировщиком REST-запросов haproxy

Нижеописанный процесс разворачивания обеспечивает только ручную балансировку запросов (рис. 38).

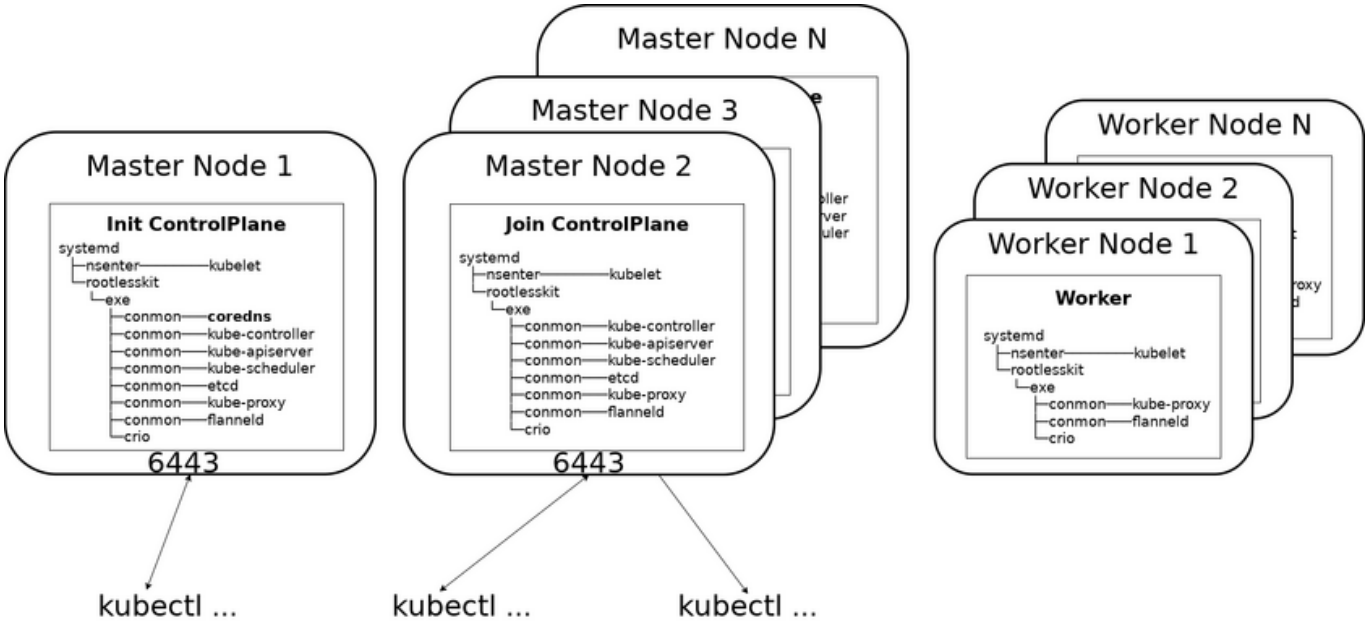

Рис. 38

Ручная балансировка запросов к API-интерфейсам master-узлов путем указания у клиентов адресов различных master-узлов довольно неудобна, так как не обеспечивает равномерного распределения запросов по узлам кластера и не обеспечивает автоматической отказоустойчивости при выходе из строя master-узлов.

Решает данную проблему установка балансировщика нагрузки haproxy (рис. 39).

Перевод кластера в режим балансировки запросов через haproxy возможен, но данная процедура не гарантирует корректный перевод на всех версиях kubernetes и ее не рекомендуют применять на production кластерах.

Так что наиболее надежным способом создания кластера с балансировкой запросов является создание нового кластера.

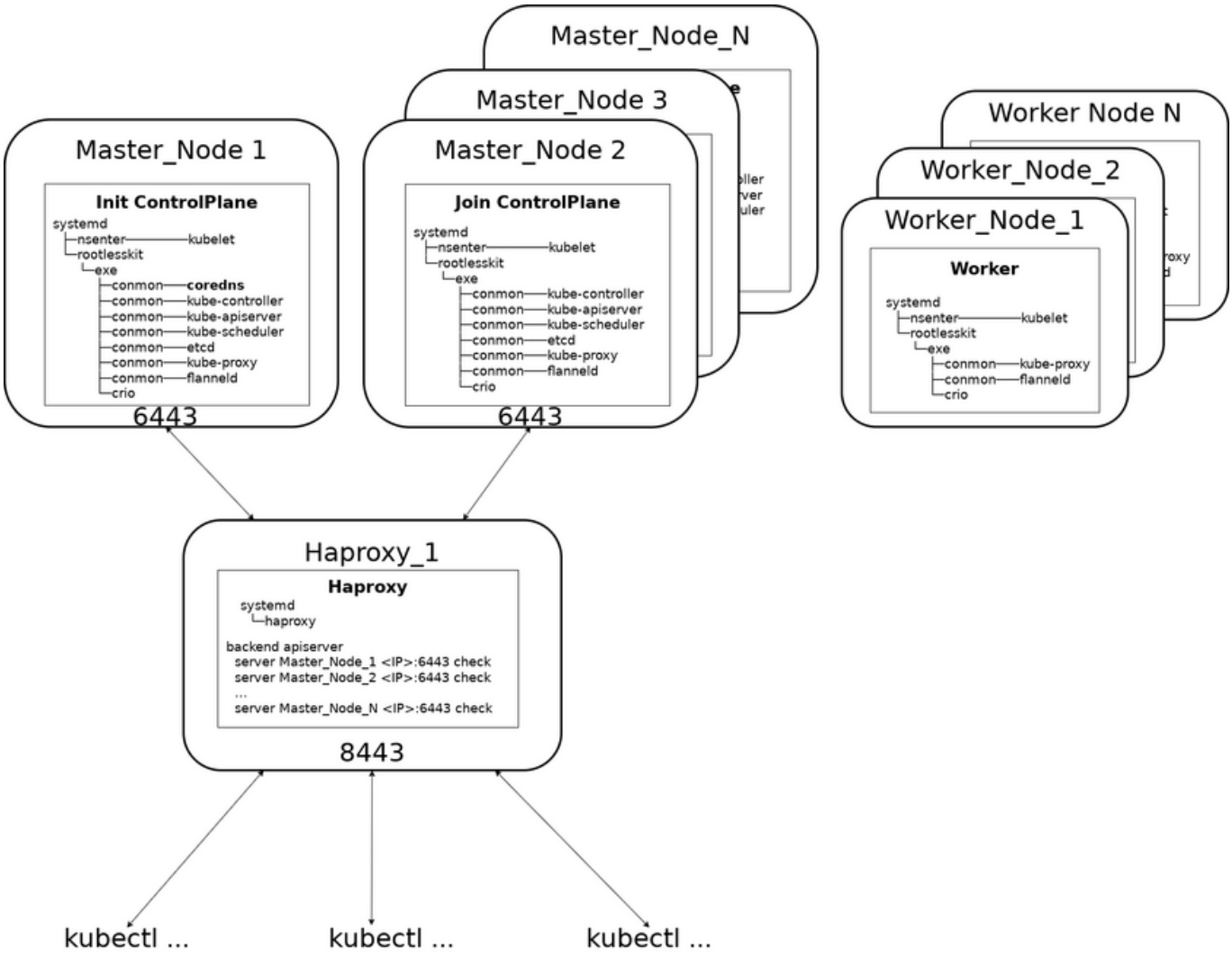

Рис. 39

4.4.2.1. Настройка балансировщика REST-запросов haproxy

Балансировщик REST-запросов haproxy можно устанавливать, как на отдельный сервер, так на один из серверов кластера (рис. 40).

П р и м е ч а н и е . Если балансировщик устанавливается на rootless сервер кластера, то для балансировщика необходимо выделить отдельный IP-адрес. Если на этом же сервере функционируют локальный регистратор (registry.local) и сервер подписей (sigstore.local), то IP-адрес балансировщика может совпадать с IP-адресами этих сервисов. Если планируется создание отказоустойчивого решения на основе нескольких серверов haproxy, то для них, кроме собственного IP-адреса, необходимо будет для всех серверов haproxy выделить один общий IP-адрес, который будет иметь master-балансировщик.

Далее рассмотрим создание и настройку одного сервера haproxy с балансировкой запросов на master-узлы.

Установите пакет haproxy:

```
# apt-get install haproxy
```

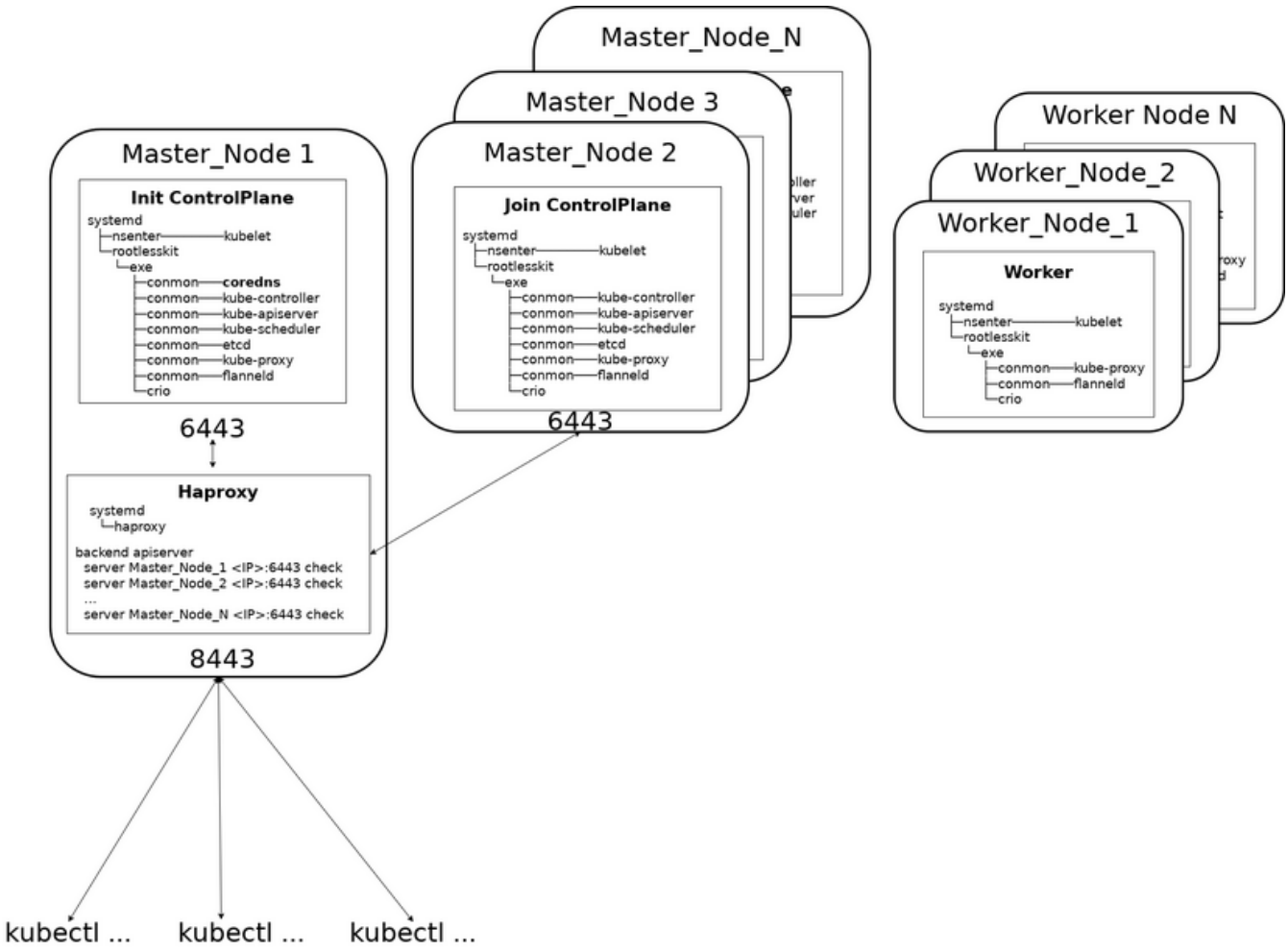

Рис. 40

Отредактируйте конфигурационный файл `/etc/haproxy/haproxy.cfg`:

- добавьте в него описание `frontend main`, принимающего запросы по порту 8443:

```
frontend main
 bind *:8443
 mode tcp
 option tcplog
 default_backend apiserver
```

- добавьте описание `backend apiserver`:

```
backend apiserver
 option httpchk GET /healthz
 http-check expect status 200
 mode tcp
 option ssl-hello-chk
 balance     roundrobin
  server master01 <IP_или_DNS_начального_мастер_узла>:6443 check
```

- запустите haproxy:

```
# systemctl enable haproxy
# systemctl start haproxy
```

4.4.2.2. Инициализация master-узла

4.4.2.2.1. Инициализация master-узла при работе с балансировщиком haproxy

При установке начального master-узла необходимо параметру `control-plane-endpoint` указать URL балансировщика haproxy:

```
# kubeadm init --apiserver-advertise-address 192.168.122.80 \
--control-plane-endpoint <IP_адрес_haproxy>:8443
```

При запуске в параметре `--apiserver-advertise-address` укажите IP-адрес API-интерфейса kube-apiserver.

IP-адреса в параметрах `--apiserver-advertise-address` и `--control-plane-endpoint` должны отличаться. Если развернули haproxy на том же master-узле, поднимите на сетевом нтерфейсе дополнительный IP-адрес и укажите его в параметре `--control-plane-endpoint`.

В результате инициализации kubeadm выведет команды подключения дополнительных control-plane и worker-узлов:

```
...
You can now join any number of the control-plane node running the following command on each as root:

kubeadm join <IP_адрес_haproxy>:8443 --token ... \
      --discovery-token-ca-cert-hash sha256:... \
      --control-plane --certificate-key ...

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 <IP_адрес_haproxy>:8443 --token ... \
      --discovery-token-ca-cert-hash sha256:...
...
```

Обратите внимание – в командах присоединения узлов указывается не URL созданного начального master-узла (`<IP_или_DNS_начального_мастер_узла>:6443`), а URL haproxy.

В сформированных файлах конфигурации `/etc/kubernetes/admin.conf`, `~/.kube/config` также указывается URL haproxy:

```
apiVersion: v1
clusters:
- cluster:
...
    server: https://<IP_адрес_haproxy>:8443
```

То есть вся работа с кластером в дальнейшем идет через балансировщик запросов haproxy.

Для перевода узла в состояние `Ready` и запуска coredns Pod запустите flannel.

4.4.2.2.2. Запуск сетевого маршрутизатора для контейнеров kube-flannel

На master-узле под пользоваталем root выполните команду:

```
# kubectl apply -f /etc/kubernetes/manifests/kube-flannel.yml
Connected to the local host. Press ^] three times within 1s to
exit session.
[INFO] Entering RootlessKit namespaces: OK
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
Connection to the local host terminated.
```

После завершения скрипта в течение минуты настраиваются сервисы master-узла кластера. По ее истечении проверьте работу usernetes (rootless kuber).

4.4.2.3. Подключение дополнительных master-узлов

4.4.2.3.1. Установка тропы PATH поиска исполняемых команд

Измените переменную `PATH`:

```
export PATH=/usr/libexec/podsec/u7s/bin/:$PATH
```

4.4.2.3.2. Подключение master (control plane) узла

Скопируйте строку подключения control-plane узла и вызовите ее:

```
# kubeadm join <IP_адрес_haproxy>:8443 --token ... \
        --discovery-token-ca-cert-hash sha256:... \
        --control-plane --certificate-key ...
```

В результате работы команда `kubeadm` выведет строки:

```
 This node has joined the cluster and a new control plane instance was created:

 * Certificate signing request was sent to apiserver and approval was received.
 * The Kubelet was informed of the new secure connection details.
 * Control plane label and taint were applied to the new node.
 * The Kubernetes control plane instances scaled up.
 * A new etcd member was added to the local/stacked etcd cluster.
 ...
 Run 'kubectl get nodes' to see this node join the cluster.
```

Наберите на вновь созданном (или начальном) control-plane узле команду:

```
# kubectl  get nodes
NAME      STATUS   ROLES          AGE    VERSION
<host1>   Ready    control-plane   4m31s   v1.26.3
<host2>   Ready    control-plane   26s    v1.26.3
```

Обратите внимание, что роль (`ROLES`) обоих узлов – control-plane.

Наберите команду (рис. 41):

```
# kubectl get all -A
```

Примечание. На рис. 41 приведен пример вывода работы команды, зависит от версии ядра ОС и модулей kubernetes

Убедитесь, что сервисы pod/etcd, kube-apiserver, kube-controller-manager, kube-scheduler, kube-proxy, kube-flannel запустились на обоих control-plane узлах.

```
# kubectl get all -A
NAMESPACE      NAME                                       READY   STATUS    RESTARTS       AGE    IP             NODE
NOMINATED NODE   READINESS GATES
kube-flannel   pod/kube-flannel-ds-2mhqg                  1/1     Running   0              153m   10.96.0.1
<host1>   <none>           <none>
kube-flannel   pod/kube-flannel-ds-95ht2                  1/1     Running   0              153m   10.96.122.68
<host2>    <none>           <none>
...
kube-system    pod/etcd-<host1>                           1/1     Running   0              174m   10.96.0.1
<host1>   <none>           <none>
kube-system    pod/etcd-<host2>                           1/1     Running   0              170m   10.96.122.68
<host2>    <none>           <none>

kube-system    pod/kube-apiserver-<host1>                 1/1     Running   0              174m   10.96.0.1
<host1>   <none>           <none>
kube-system    pod/kube-apiserver-<host2>                 1/1     Running   0              170m   10.96.122.68
<host2>    <none>           <none>

kube-system    pod/kube-controller-manager-<host1>        1/1     Running   1 (170m ago)   174m   10.96.0.1
<host1>   <none>           <none>
kube-system    pod/kube-controller-manager-<host2>        1/1     Running   0              170m   10.96.122.68
<host2>    <none>           <none>

kube-system    pod/kube-proxy-9nbxz                       1/1     Running   0              174m   10.96.0.1
<host1>   <none>           <none>
kube-system    pod/kube-proxy-bnmd7                       1/1     Running   0              170m   10.96.122.68
<host2>    <none>           <none>

kube-system    pod/kube-scheduler-<host1>                 1/1     Running   1 (170m ago)   174m   10.96.0.1
<host1>   <none>           <none>
kube-system    pod/kube-scheduler-<host2>                 1/1     Running   0              170m   10.96.122.68
<host2>    <none>           <none>
...

NAMESPACE      NAME                                 DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR
AGE    CONTAINERS     IMAGES                                                SELECTOR
kube-flannel   daemonset.apps/kube-flannel-ds   2         2         2       3            3           <none>
153m   kube-flannel   registry.local/k8s-c10f1/flannel:v0.19.2      app=flannel
kube-system    daemonset.apps/kube-proxy        2         2         2       2            2
kubernetes.io/os=linux   174m   kube-proxy     registry.local/k8s-c10f1/kube-proxy:v1.26.3   k8s-app=kube-proxy
...
```

Рис. 41– Пример вывода

4.4.2.3.3. Добавление master-узла в балансировщик haproxy

Для балансировки запросов по двум серверам добавьте URL подключенного control-plane узла в файл конфигурации `/etc/haproxy/haproxy.cfg`:

```
backend apiserver
    option httpchk GET /healthz
    http-check expect status 200
    mode tcp
    option ssl-hello-chk
    balance     roundrobin
        server master01 <IP_или_DNS_начального_мастер_узла>:6443 check
        server master02 <IP_или_DNS_подключенного_мастер_узла>:6443 check
```

и перезапустите haproxy:

```
# systemctl restart haproxy
```

Логи обращений и балансировку запросов между узлами можно посмотреть командой:

```
# tail -f /var/log/haproxy.log
```

4.4.2.4. Подключение worker-узлов

Подключение дополнительных worker-узлов происходит аналогично описанному в п. 4.4.1.2.

4.4.2.5. Настройка отказоустойчивого кластера серверов haproxy, keepalived

4.4.2.5.1. Масштабирование haproxy, установка пакетов

Если необходимо создать отказоустойчивое решение, допускающее выход haproxy-севрера из строя, установите haproxy на несколько серверов. Файлы конфигурации `haproxy<.code>` на всех серверах должны быть идентичны.

Для контроля доступности haproxy и переназначений виртуального адреса дополнительно установите на каждом сервис keepalived:

```
# apt-get install haproxy keepalived
```

4.4.2.5.2. Конфигурирование keepalived

На рис. 42 приведен kubeenetes кластер с haproxy и keepalived.

Создайте файл конфигурации keepalived `/etc/keepalived/keepalived.conf`:

```
! /etc/keepalived/keepalived.conf
! Configuration File for keepalived
global_defs {
   router_id LVS_K8S
}
vrrp_script check_apiserver {
  script "/etc/keepalived/check_apiserver.sh"
  interval 3
  weight -2
  fall 10
  rise 2
}

vrrp_instance VI_1 {
    state MASTER
    interface br0
    virtual_router_id  51
    priority 101
    authentication {
        auth_type PASS
```

```
        auth_pass 42
    }
    virtual_ipaddress {
        10.150.0.160
    }
    track_script {
        check_apiserver
    }
}
```

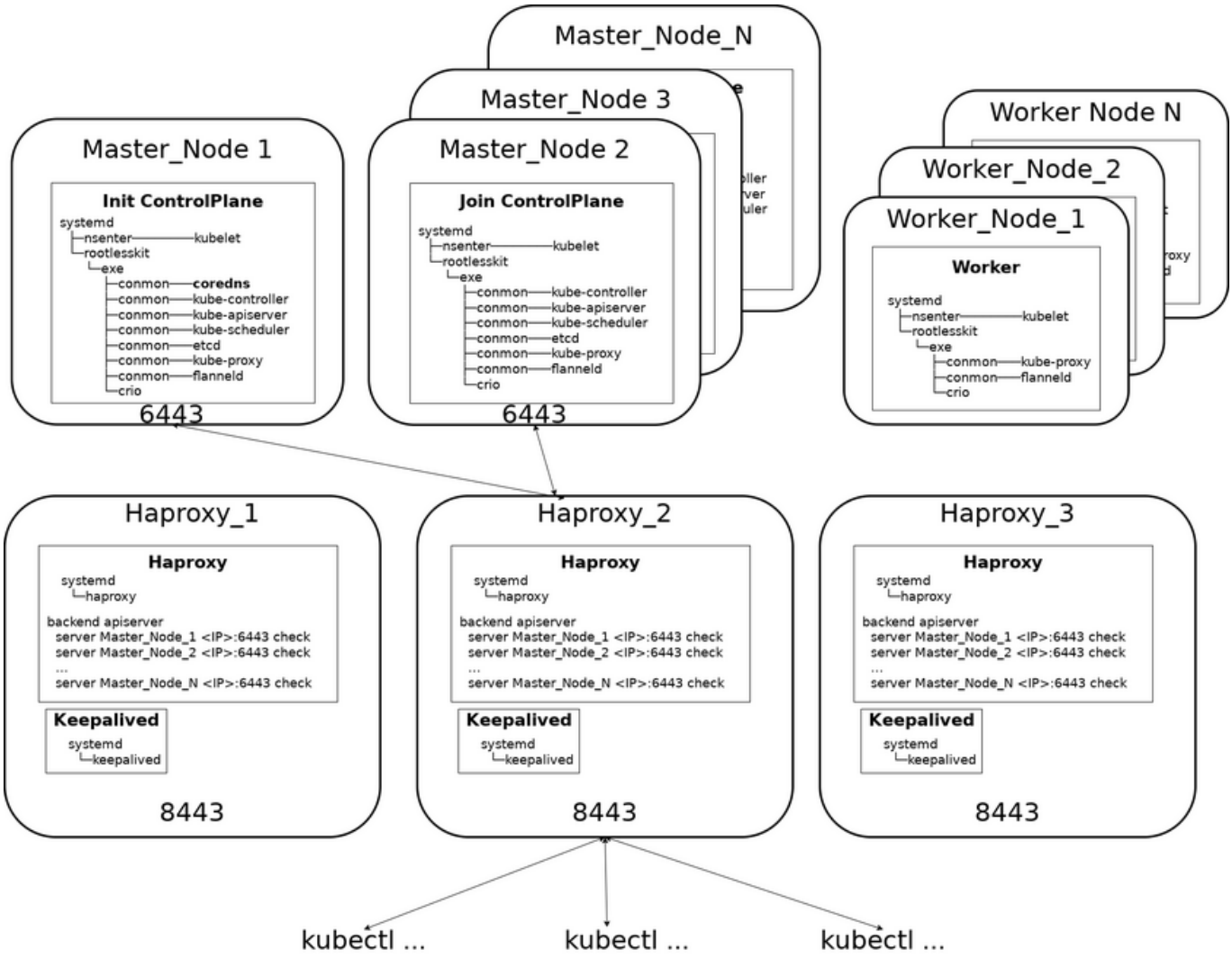

Рис. 42

На одном из узлов установите параметр `state` в значение `MASTER` и параметр `priority` в значение 101. На остальных параметр `state` в значение `BACKUP` и параметр `priority` в значение 100.

Скрипт `/etc/keepalived/check_apiserver.sh` проверяет доступность балансировщика haproxy:

```
#!/bin/sh

errorExit() {
   echo "*** $*" 1>&2
   exit 1
}

APISERVER_DEST_PORT=8443
APISERVER_VIP=10.150.0.160
curl --silent --max-time 2 --insecure https://localhost:${APISERVER_DEST_PORT}/ -o /dev/null || errorExit "Error GET https://localhost:${APISERVER_DEST_PORT}/"
if ip addr | grep -q ${APISERVER_VIP}; then
   curl --silent --max-time 2 --insecure https://${APISERVER_VIP}:${APISERVER_DEST_PORT}/ -o /dev/null || errorExit "Error GET https://${APISERVER_VIP}:${APISERVER_DEST_PORT}/"
fi
```

Параметр `APISERVER_DEST_PORT` задает порт балансировщиков haproxy, параметр `APISERVER_VIP` виртуальный адрес, через который будут взаимодействовать master (control plane) узлы кластера k8s.

Скрипт проверяет работоспособность haproxy на локальной машине.

Если в настоящее время виртуальный адрес принадлежит текущему узлу, то и работоспособность haproxy через виртуальный адрес тоже.

Добавьте флаг на выполнение скрипта:

```
chmod a+x /etc/keepalived/check_apiserver.sh
```

При работающем балансировщике и хотя бы одном доступном порте 6443 на master-узлах скрипт должен завершаться с кодом 0.

4.4.3. Установка и настройка ingress-контролера

Ingress-контроллер обеспечивает переадресацию http(s) запросов по указанным шаблонам на внутренние сервисы kubernetes-кластера. Для bare-metal решений и решений на основе виртуальных машин наиболее приемлимым является ingress-nginx контроллер.

При применении Ingress-контроллера нет необходимости создавать Nodeport-порты и пробрасывать их из namespace пользователя u7s-admin. Ingress-контроллер переадресует http{s) запрос через сервис непосредственно на порты Pod'ов входящих в реплики deployment.

4.4.3.1. Установка и настройка ingress-nginx-контролера в кластере

Использование ingress-контроллера приведено на рис. 43.

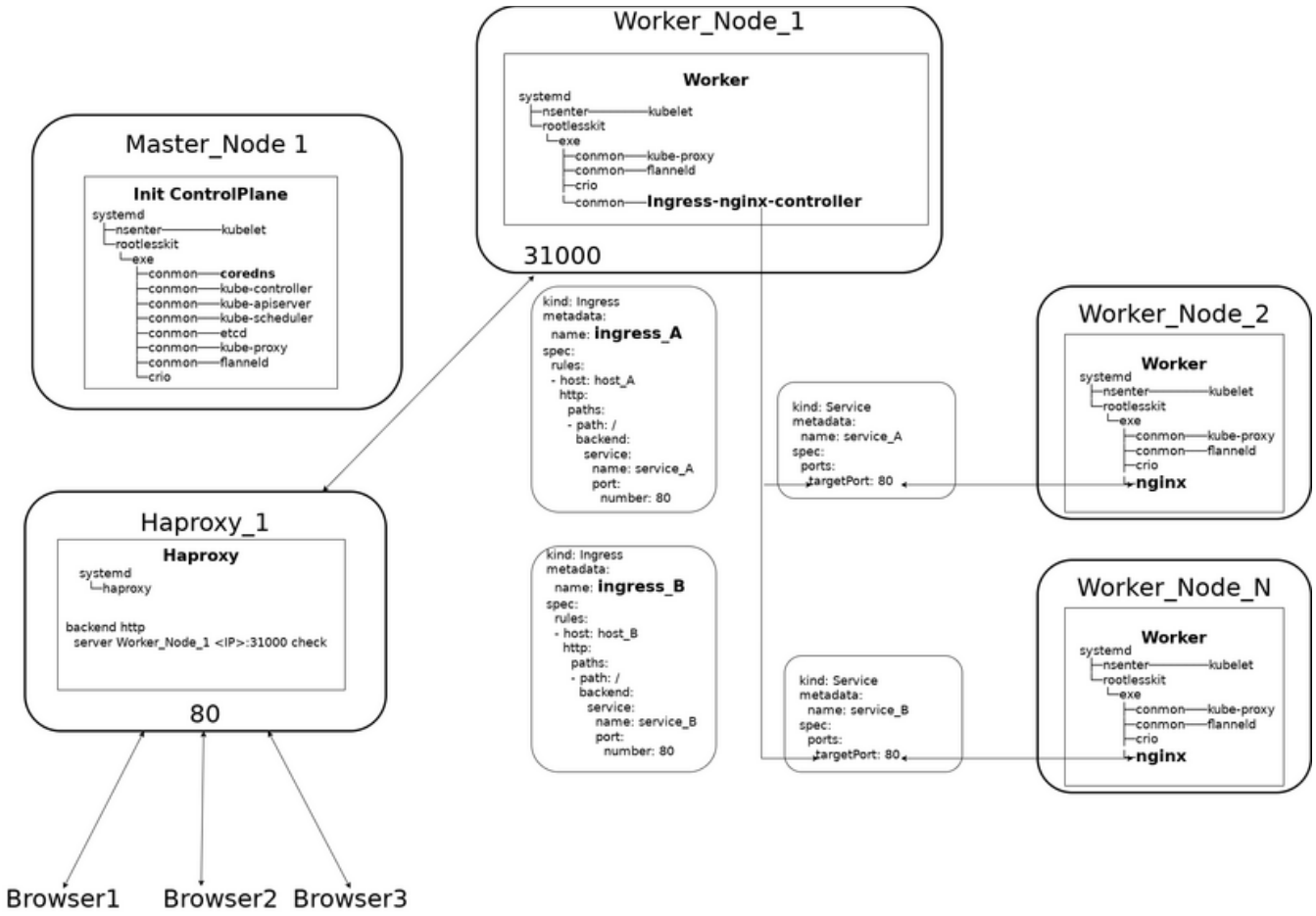

Рис. 43

Для установки Ingress-контроллера скопируйте его YAML-манифест:

```
curl https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.8.0/deploy/static/provider/baremetal/deploy.yaml -o ingress-nginx-deploy.yaml
```

Выберите свободный порт из диапазона 30000 – 32767 (например, 31000) и добавьте его в элемент `spec.ports.appProtocol==http` Yaml-описании `kind==Service`:

```
...
```

```
---
kind: Service
spec:
  ports:
  - appProtocol: http
    ...
    nodePort: 31000
...
```

Если используете только локальный регистратор registry.local:

- создайте алиасы образам nginx:

```
podman tag registry.k8s.io/ingress-nginx/controller:v1.8.0@sha256:744ae2afd433a395eeb13dc03d3313facba92e96ad71d9feaafc85925493fee3 registry.local/ingress-nginx/controller:v1.8.0
podman tag registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20230407@sha256:543c40fd093964bc9ab509d3e791f9989963021f1e9e4c9c7b6700b02bfb227b registry.local/ingress-nginx/kube-webhook-certgen:v20230407
```

и поместите их в локальный регистратор:

```
podman push --tls-verify=false --sign-by='<EMAIL>' registry.local/ingress-nginx/controller
podman push --tls-verify=false --sign-by='<EMAIL>' registry.local/ingress-nginx/kube-webhook-certgen
```

- исправьте имена образов в скачанном манифесте на имена образов в локальном регистраторе.

Запустите Ingress-nginx-контролер:

```
kubectl apply -f ingress-nginx-deploy.yaml
```

На одном или нескольких kubernet-узлах (эти узлы в дальнейшем нужно прописать в файле конфигурации балансировщика haproxy) пробросьте порт nginx-контроллера (31000) из namespace пользователя u7s-admin в сеть kubernetes:

```
nsenter_u7s rootlessctl add-ports 0.0.0.0:31000:31000/tcp
```

4.4.3.2. Настройка Ingress-правил

Kubernetes поддерживает манифесты типа Ingress (kind: Ingress), описывающие правила переадресации запросов URL http-запроса на внутренние порты сервисов (kind: Service) kubernetes. Сервисы в свою очередь перенаправляют запросы на реплики Pod'ов, входящих в данный сервис.

Общий вид Ingress-манифеста:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: <ingress_имя>
spec:
  ingressClassName: nginx
  rules:
  - host: <домен_1>
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: <имя_сервиса_1>
            port:
              number: 80
      - path: /<тропа_1>
        pathType: Prefix
        backend:
          service:
            name: <имя_сервиса_2>
            port:
              number: 80
  - host: <домен_2>
    ...
```

где:

- `host: <домен_1>, <домен_2>, ...` – домены веб-серверов на которых приходит запрос;
- `path:/>, path:/<тропа_1>` – тропы (префиксы запросов после домена);
- `pathType: Prefix` – тип троп: Prefix или Exact;
- `service:` – имя сервиса на который перенаправляется запрос, если полученный запрос соответствует правилу;
- `port` – номер порта на который перенаправляется запрос.

Если запросу соответствует несколько правил, выбирается правило с наиболее длинным префиксом.

4.4.3.3. Настройка haproxy и DNS

Добавьте в файлы конфигурации haproxy `/etc/haproxy/haproxy.conf` переадресацию запросов на порт 80 (http) по IP-адресу балансировщика haproxy на

IP-адреса kubernet-узлов на которых выбранный порт nginx-контроллера (31000) проброшен из namespace пользователя u7s-admin в сеть kubernetes:

```
frontend http
   bind *:80
   mode tcp
   option tcplog
   default_backend http

backend http
   mode tcp
   balance     roundrobin
      server <server1> <ip1>:31000 check
      server <server2> <ip2>:31000 check
```

Заведите DNS-запись, связывающую DNS-имя http-сервиса с IP-адресам haproxy-сервера.

4.4.4. Выбор версии kubernetes, имени регистратора и платформы

Во время разворачивания узла командами:

```
kubeadm init
kubeadm join ...
```

или при создании архива образов командой

```
podsec-k8s-save-oci ...
```

есть возможность установкой переменных среды указать версию kubernetes, имя регистратора и платформы:

- `U7S_KUBEVERSION` – версия kubernetes (например, v1.26.9, v1.27.7, ...);
- `U7S_REGISTRY` – имя регистратора (например, registry.k8s.io, registry.altlinux.org, registry.local);
- `U7S_PLATFORM` – имя платформы (например, k8s-c10f2, k8s-p10, ...).

4.4.4.1. Выбор версии kubernetes

Начиная с версии 1.0.9 поддерживается возможность выбора устанавливаемой версии kubernetes.

4.4.4.1.1. Указание версии основных kubernetes образов

Основные kubernetes-образы загружаются в момент инициализации узла командой `kubeadm`. В список основных образов входят:

```
kube-apiserver:<версия_kubernetes>
kube-controller-manager:<версия_kubernetes>
```

```
kube-scheduler:<версия_kubernetes>
kube-proxy:<версия_kubernetes>
pause:<версия__образа_pause>
etcd:<версия__образа_etcd>
coredns:<версия__образа_coredns>
```

Тег образов `kube-*` совпадает с полным номером версии kubernetes типа `v1.<minor>.<patch>`. Например, v1.26.9.

Теги образов `pause`, `etcd`, `coredns` «зашиты» как статические переменные в `kubeadm` и могут отличаться в разных версиях kubernetes.

Получить список образов для текущей версии kubernetes можно командой:

```
# /usr/bin/kubeadm  config images list 2>/dev/null
```

Пример вывода:

```
registry.k8s.io/kube-apiserver:v1.26.10
registry.k8s.io/kube-controller-manager:v1.26.10
registry.k8s.io/kube-scheduler:v1.26.10
registry.k8s.io/kube-proxy:v1.26.10
registry.k8s.io/pause:3.9
registry.k8s.io/etcd:3.5.9-0
registry.k8s.io/coredns/coredns:v1.9.3
```

Выбор версии определяет переменная среды `U7S_KUBEVERSION`.

При значении `U7S_REGISTRY registry.altlinux.org` переменная `U7S_KUBEVERSION` может принимать следующие значения:

1) минор версия v1.26:

- v1.26.6;
- v1.26.9;
- v1.26.11;

2) минор версия v1.27:

- v1.27.11;

3) минор версия v1.28:

- v1.28.7.

Данный список относится к версиям регистратора `registry.altlinux.org`. При использовании нативного регистратора `registry.k8s.io` (пустое значение `export U7S_REGISTRY=`) можно указывать любую доступную на `registry.k8s.io` версию.

Примеры:

```
export U7S_REGISTRY=registry.altlinux.org
export U7S_KUBEVERSION=v1.26.11

export U7S_REGISTRY=registry.local
export U7S_KUBEVERSION=v1.27.11

export U7S_REGISTRY=
export U7S_KUBEVERSION=v1.27.5
```

По умолчанию (при отсутствии значения переменной `U7S_KUBEVERSION`) принимается максимальная версия образа kube-apiserver в рамках минорной версии, которая определяется по минорной версии пакета kubeadm.

Если номер указанной минорной версии kubernetes отличается от текущего, при вызове команды `kubeadm` производится удаление текущих rpm-пакетов kubernetes*, cri-o и установка rpm-пакетов указанной версии.

Возможна ситуация, когда на регистраторе образов отсутствует версия образа, полученная в результате выполнения команды:

```
/usr/bin/kubeadm  config images list
```

Если в переменные среды добавить переменную:

```
export U7S_SETAVAILABLEIMAGES=yes
```

то в качестве стандартного образа принимается образ с максимальной версией в рамках данной минорной версии (1.26, 1.27, 1.28). Данному образу присваивается тег, полученный в результате выполнения команды `kubeadm config images`.

4.4.4.1.2. Указание версии дополнительных kubernetes образов

Кроме основных образов при разворачивании кластера используются дополнительные образы:

- `flannel:<U7S_FLANNEL_TAG>`;
- `flannel-cni-plugin:<U7S_FLANNELCNIPLUGIN_TAG>`;
- `cert-manager-webhook:<U7S_CERTMANAGER_TAG>`;
- `cert-manager-controller:<U7S_CERTMANAGER_TAG>`;
- `cert-manager-cainjector:<U7S_CERTMANAGER_TAG>`.

Если переменным среды U7S_FLANNEL_TAG, U7S_FLANNELCNIPLUGIN_TAG, U7S_CERTMANAGER_TAG не присвоены значения, то для каждого образа определяется максимальная версия в регистраторе и загружается найденная версия образа.

При необходимости можно изменить версию образа экпортировав перед запуском команды соответствующую переменную.

Например:

```
export U7S_FLANNEL_TAG=v0.19.2
```

4.4.4.2. Выбор исходного регистратора kubernetes-образов

Во время инициализации master-узла кластера (kubeadm init) или во время подключения узла к кластеру (kubeadm join) команда kubeadm может загружать образы с различных регистраторов образов и с различными префиксами.

Выбор регистратора и префикса образов определяет переменная среды U7S_REGISTRY. Если переменная не задана регистратор префикс определяется автоматически на основе конфигурационных файлов /etc/os-release и /etc/hosts.

Переменная среды U7S_REGISTRY может принимать следующие основные значения:

- пустое значение;
- registry.altlinux.org;
- registry.local;
- ...

4.4.4.2.1. Нативные kubernetes-образы

```
export U7S_REGISTRY=
```

Если переменная U7S_REGISTRY установлена в пустое значение, образы загружаются со стандартного регистратора образов kubernetes.

4.4.4.2.2. Образы altlinux

4.4.4.2.2.1. Регистратор registry.altlinux.org

```
export U7S_REGISTRY=registry.altlinux.org
```

С регистратора altlinux устанавливаются образы при наличии доступа в Интернет.

4.4.4.2.2.2. Локальный регистратор

```
export U7S_REGISTRY=registry.local
```

Локальный регистратор используется в сертифицированных дистрибутивах, которые содержат kubernetes-образы на установочном диске.

Локальный регистратор образов `registry.local` может обеспечивать:

- разворачивание кластера без доступа в Интернет;
- ускоренное разворачивание как кластера, так и проектов, разворачиваемых в его рамках, так как образы необходимые для запуска Pod'ов загружаются по локальной сети;
- высокий уровень защищенности системы путем установки политик, разрешающих загрузку только подписанных образов и только с локального регистратора `registry.local`.

Пакет podsec обеспечивает:

- установку на рабочих местах клиентов и узлах kubernetes политик доступа к образам для различных категория пользователей (скрипт podsec-create-policy);
- разворачивание на одном узлов локального регистратора образов и сервера подписей образов (скрипт podsec-create-services);
- загрузку с регистратора `registry.altlinux.org` образов необходимых для разворачивания kubernetes и формирования максимально сжатого (<200Mb) архива (скрипты podsec-k8s-save-oci, podsec-save-oci);
- разворачивание образов из архива, их подпись размещение на локальном регистраторе (скрипт podsec-load-sign-oci).

В зависимости от значения переменных `U7S_REGISTRY`, `U7S_PLATFORM`, `U7S_KUBEVERSION` скрипт podsec-k8s-save-oci формирует архив образов различных версий kubernetes:

- `registry.local/k8s-c10f2` – архив образов для сертифицированного дистрибутива c10f2 на основе набора образов с регистратора `registry.altlinux.org` с платформой `k8s-c10f2`;

- registry.local/k8s-p10 – архив образов для несертифицированного дистрибутива p10 на основе набора образов с регистратора registry.altlinux.org с платформой k8s-p10.

Локальный регистратор registry.local может также хранить подписанные образы и запускаемых в рамках кластера проектов. Необходимо только, чтобы каждый образ в рамках локального регистратор registry.local имел префикс. Образы типа registry.local/<имя_образа> не допускаются, так как для них трудно определить «подписанта» образа.

4.4.4.2.3. podsec-create-policy – настройка политики доступа к образам различным категориям пользователей

Формат:

podsec-create-policy <ip-адрес_регистратора_и_сервера_подписей>

Описание: скрипт podsec-create-policy формирует в файлах /etc/containers/policy.json, /etc/containers/registries.d/default.yaml максимально защищенную политику доступа к образам – по умолчанию допускается доступ только к подписанным образам локального регистратора registry.local. Данная политика распространяется как на пользователей, имеющих права суперпользователя, так и на пользователей группы podsec, создаваемые podsec-скриптом podsec-create-podmanusers.

Пользователи группы podsec-dev, создаваемые podsec-скриптом podsec-create-imagemakeruser имеют неограниченные права на доступ, формирования образов, их подпись и помещение на локальный регистратор registry.local.

В разворачиваниях kubernetes, не требующих таких жестких ограничений в политике доступа и работы с образами, политики могут быть смягчены путем модифицирования системных файлов политик /etc/containers/policy.json, /etc/containers/registries.d/default.yaml или файлов установки политик пользователей ~/.config/containers/policy.json, ~/.config/containers/registries.d/default.yaml.

4.4.4.2.4. podsec-create-services – разворачивание локального регистратора образов и сервера подписей образов

Скрипт podsec-create-services обеспечивает разворачивание локального регистратора образов и сервера подписей образов.

4.4.4.2.5. Загрузка образов, поддержка электронной подписи образов

4.4.4.2.5.1. Загрузка образов kubernetes

Для kubernetes-образов, хранящихся в архиве образов распаковку образов, их подпись и размещение на локальном регистраторе `registry.local` обеспечивает скрипт podsec-load-sign-oci запускаемый пользователем группы podsec-dev.

Формат вызова команды:

```
podsec-load-sign-oci <xz-архив-kubernetes-образов> <архитектура> <e-mail-подписывающего>
```

4.4.4.2.5.2. Загрузка базовых образов

Кроме архива kubernetes-образов есть архив базовых образов с префиксом alt. В состав базовых входят образы:

- `alt/alt:платформа`;
- `alt/distroless-base:платформа`.

Где `платформа` – например, c10f2, p10, p11, sisyphus, ...

Для их загрузки необходимо экспортировать переменную `U7S_PLATFORM`:

```
export U7S_PLATFORM=alt
```

Команда разворачивания архива, подписи образов и размещения их на регистраторе registry.local выглядит следующим образом:

```
U7S_PLATFORM=alt podsec-load-sign-oci <xz-архив-базовых-образов> <архитектура> <e-mail-подписывающего>
```

После размещения образы доступны под именами:

- `registry.local/alt/alt:платформа`;
- `registry.local/alt/distroless-base:платформа`.

4.4.4.2.5.3. Загрузка создаваемых или сторонних образов

Образ в домене `registry.local/</prefix>/` может быть получен после:

- присваивания алиаса стороннему образу:

```
podman tag <сторонний_образ> registry.local/</prefix>/<локальный_образ>
```

- сборки образов через Dockerfile:

```
podman build -t registry.local/</prefix>/<локальный_образ> ...
```

Для этих образов пользователь группы podsec-dev должен создать образ в домене локального регистратора `registry.local/</prefix>/` и поместить его в регистратор командой:

```
podman push --tls-verify=false --sign-by="<email-подписанта" <образ>
```

4.4.4.3. Указание платформы

Кроме имени регистратора kubernetes-образы altlinux содержат в имени (например, `registry.altlinux.org/k8s-p10/kube-apiserver`) название платформы:

- k8s-p10 – образы для дистрибутива p10;
- k8s-c10f* – образы сертифицированного дистрибутива c10;
- test_k8s – тестовые образы;
- ...

Примечание. Вместо * актуальный номер дистрибутива, например, k8s-c10f1, k8s-c10f2 и т. п.

Платформу устанавливаемых образов можно указать в переменной `U7S_PLATFORM`. Например:

```
export U7S_PLATFORM=test_k8s
```

4.4.4.4. Автоматический выбор регистратора образов и платформы

Если переменная `U7S_REGISTRY` не установлена, ее значения вычисляется автоматически по следующему алгоритму:

- если файл `/etc/hosts` содержит описание хоста `registry.local` префикс переменной `U7S_REGISTRY` принимает значение `registry.local/`, иначе `registry.altlinux.org/`;
- если переменная `CPE_NAME` файла `/etc/os-release` содержит значение `spserver` суффикс переменной `U7S_PLATFORM` принимает значение k8s-c10f2, иначе k8s-p10.

4.4.5. Добавление новых образов в локальный регистратор registry.local на платформах c10f

Rootless-kubernetes, разворачиваемый на платформах вида c10f должен обеспечивать работу при отсутствии доступа в Интернет. В этом случае в рамках kubernetes-кластера поднимается локальный регистратор registry.local. На всех узлах кластера в файле `/etc/hosts` производится привязка имени registry.local к IP-адресу основного master-сервера kubernetes. Кроме этого на master-сервере поднимается веб-сервер под именем sigstore.local для доступа к открытым GPG-ключам пользователей, помещающих подписанные образы в регистратор registry.local.

4.4.5.1. Пользователи группы podman_dev

Добавление и корректировка docker-образов производятся пользователями принадлежащей группе podman_dev. При разворачивании кластера эти пользователи создаются скриптом podsec-create-imagemakeruser. Стандартно по документации создается один пользователь с именем imagemaker. В дальнейшем мы будем использовать это имя.

При создании пользователя создаются открытый и закрытый GPG-ключи для подписывания помещаемых в registry.local образов. Открытый ключ помещается в каталог `/var/sigstore/keys/` под именем imagemaker.pgp. Данный файл доступен с любого узла кластера по http-протоколу по адресу `http://sigstore.local:81/keys/imagemaker.pgp`.

4.4.5.2. Структура каталогов и файлов политик доступа к регистраторам для обычных пользователей

Кроме этого в каталоге `/var/sigstore/keys/` master-сервера находится файл `policy.json`, являющийся копией файла политик доступа к регистраторам `/etc/containers/policy.json`. Данный файл доступен с любого узла кластера по http-протоколу по адресу `http://sigstore.local:81/keys/policy.json`. Это файл используется для формирования файлов `/etc/containers/policy.json` на разворачиваемых узлов кластера.

Файл `policy.json` имеет следующее содержание:

```
{
  "default": [
    {
      "type": "reject"
    }
  ],
  "transports": {
    "docker": {
      "registry.local": [
        {
          "type": "signedBy",
          "keyType": "GPGKeys",
          "keyPath": "/var/sigstore/keys/imagemaker.pgp"
        }
      ]
    }
  }
}
```

Это файл запрещает любой доступ к регистаторам, кроме регистратора registry.local. При этом образы на данном регистраторе должны быть подписаны пользователем, имеющим открытый ключ, хранящийся в файле `/var/sigstore/keys/imagemaker.pgp`.

В каталог `/var/sigstore/sigstore/` помещаются электронные подписи всех образов, хранящихся в регистраторе registry.local.

```
/var/sigstore/sigstore/
└── k8s-c10f2
    ├── coredns@sha256=8199b34e550b94f6f2fb0d5539e3d1ac861db3b3cabdde85d72024d26f631e80
    |   └── signature-1
    ...
```

Кроме этого в файле `/etc/containers/registries.d/default.yaml` описываются методы доступа к электронным подписям образов. Он имеет следующее содержание:

```
default-docker:
  lookaside: http://sigstore.local:81/sigstore/
  sigstore: http://sigstore.local:81/sigstore/
```

URL `http://sigstore.local:81/sigstore/` указывает, а каталог `/var/sigstore/sigstore/` на master-сервере используется для доступа к электронным подписям образов. Данные подписи при загрузке образов проверяются на соответствие открытому ключу, хранящемуся в файле `/var/sigstore/keys/imagemaker.pgp`. В случае их соответствия образ загружается.

4.4.5.3. Структура каталогов и файлов политик доступа к регистраторам пользователей группы podman_dev (imagemaker)

Пользователи группы podman_dev в домашнем каталоге содержат файлы:

- ~/.config/containers/policy.json;

- ~/.config/containers/registries.d/default.yaml.

Эти файлы перекрывают содержимое системных файлов /etc/containers/policy.json, /etc/containers/registries.d/default.yaml.

Файл ~/.config/containers/policy.json имеет следующее содержание:

```
{
  "default": [
    {
      "type": "insecureAcceptAnything"
    }
  ],
  "transports": {
    "docker": {
      "registry.local": [
        {
          "type": "signedBy",
          "keyType": "GPGKeys",
          "keyPath": "/var/sigstore/keys/imagemaker.pgp"
        }
      ]
    }
  }
}
```

default.type=insecureAcceptAnything определяет, что данный пользователь может работать с образами любых регистраторов.

Файл /etc/containers/registries.d/default.yaml имеет следующее содержание:

```
default-docker:
  lookaside: http://sigstore.local:81/sigstore/
  sigstore: http://sigstore.local:81/sigstore/
  lookaside-staging: file:///var/sigstore/sigstore/
  sigstore-staging: file:///var/sigstore/sigstore/
```

Описатели lookaside-staging, lookaside-staging указывают каталог, в который записываются электронные подписи образов.

4.4.5.4. Добавление новых образов в локальный регистратор registry.local

Добавление новых образов в локальный регистратор может осуществляться только пользователями, входящими в группу podman_dev.

В регистратор могут помещаться два типа образов:

- копии сторонних образов;
- образы, собранные пользователем из группы podman_dev командой `podman build` или аналогичными.

4.4.5.4.1. Получение образов с префиксом локального регистратора

Все перечисленные образы для размещения в локальном регистраторе registry.local должные иметь в имени префикс, совпадающий с именем регистратора.

Загрузку и добавление нового алиаса к стороннему образу можно осуществить с помощью команд:

```
podman pull <имя_образа>:<тег>
podman tag <имя_образа>:<тег> registry.local.<имя_образа_без_префикса>:<тег>
```

Аналогичного результата можно добиться путем создания в отдельном пустом каталоге файла `Dockerfile`:

```
FROM <имя_образа>:<тег>
```

и создание образа командой:

```
podman build -t registry.local/<имя_образа_без_префикса>:<тег> .
```

При построении собственных образов необходимо создать в отдельном каталоге файл `Dockerfile`:

```
FROM <имя_образа>:<тег>
...
```

и построить на основе этого файла собственный образ, указав в имени собираемого образа префикс registry.local:

```
podman build -f <путь_до_Dockerfile> -t registry.local/<имя_образа_без_префикса>:<тег> <каталог_данных_для_образа>
```

4.4.5.4.2. Подписывание образов и их размещение в локальном регистраторе

Данная операция выполняется одной командой:

```
podman push --tls-verify=false --sign-by="<E-mail_подписанта>" registry.local/<имя_образа_без_префикса>:<тег>
```

4.4.5.5. Пример размещения в локальном регистраторе внешнего образа

1) Зайти на узел под пользователем imagemaker.

2) Загрузить образ `docker.io/library/nginx:1.26.2`:

```
$ podman pull docker.io/library/nginx:1.26.2
Trying to pull docker.io/library/nginx:1.26.2...
Getting image source signatures
Copying blob 692a61bd1d67 done   |
Copying blob a480a496ba95 done   |
Copying blob f7e45c747637 done   |
Copying blob eec32f85414d done   |
Copying blob 8992a25329a6 done   |
Copying blob f8eff2f530ec done   |
Copying blob 7a37000823d1 done   |
Copying config 122ce9f0cb done   |
Writing manifest to image destination
122ce9f0cbb4dfe43ffdb473f28715920b333fdb1a24276feb9164a36dc9e817
```

3) Проверить наличие образа в списке образов пользователя imagemaker:

```
$ podman images
REPOSITORY                  TAG          IMAGE ID      CREATED       SIZE
...
docker.io/library/nginx     1.26.2       122ce9f0cbb4  2 months ago  192 MB
```

4) Присвоить образу `docker.io/library/nginx:1.26.2` альтернативное имя `registry.local/library/nginx:1.26.2` с префиксом registry.local:

```
$ podman tag docker.io/library/nginx:1.26.2 \
registry.local/library/nginx:1.26.2
```

5) Проверить наличие альтернативного имени:

```
$ podman images
REPOSITORY                    TAG     IMAGE ID      CREATED       SIZE
...
registry.local/library/nginx  1.26.2  122ce9f0cbb4  2 months ago  192 MB
docker.io/library/nginx       1.26.2  122ce9f0cbb4  2 months ago  192 MB
```

Обратите внимание, что оба образа имеют одинаковый `IMAGE ID`: `122ce9f0cbb4`.

6) Подписать образ закрытым ключом с идентификатором `kaf@basealt.ru` и поместить его на регистратор registry.local:

```
$ podman push --tls-verify=false --sign-by='kaf@basealt.ru' registry.local/library/nginx:1.26.2
Getting image source signatures
Copying blob 6895d9cc0852 done   |
Copying blob 98b5f35ea9d3 done   |
Copying blob 13de84ad01b1 done   |
Copying blob c3f432d8d95a done   |
Copying blob be367852680a done   |
Copying blob f0a47df3ae96 done   |
Copying blob 244255f1ea0b done   |
Copying config 122ce9f0cb done   |
Writing manifest to image destination
Creating signature: Signing image using simple signing
```

7) Проверить наличие образа с именем `library/nginx` на регистраторе registry.local:

```
$ curl -s http://registry.local/v2/_catalog | jq
{
  "repositories": [
    ...
    "library/nginx"
  ]
}
```

8) Проверить наличие тега `1.26.2` у образа:

```
$ curl -s  http://registry.local/v2/library/nginx/tags/list | jq
{
  "name": "library/nginx",
  "tags": [
    "1.26.2"
  ]
}
```

9) Проверить наличие подписи (файла `signature-1`) в каталоге `/var/sigstore/sigstore/library/nginx@sha256=...`:

```
$ ls -lR /var/sigstore/sigstore/library/
/var/sigstore/sigstore/library/:
итого 4
drwxr-xr-x 2 imagemaker podman 4096 окт 30 17:37 'nginx@sha256=35705f3156d9dc894f5c69e3a60d018a05785d57ad13b966986043c6cef4e394'

'/var/sigstore/sigstore/library/nginx@sha256=35705f3156d9dc894f5c69e3a60d018a05785d57ad13b966986043c6cef4e394':
итого 4
-rw-r--r-- 1 imagemaker podman 595 окт 30 17:37 signature-1
```

10) Создать манифесты Deployment и Service для образа `registry.local/library/nginx:1.26.2` в namespace nginx-ns в файле `nginx.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  namespace: nginx-ns
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: registry.local/library/nginx:1.26.2
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
  namespace: nginx-ns
spec:
  type: NodePort
  ports:
  - port: 80
    targetPort: 80
  selector:
    app: nginx
```

11) От администратора (root) создать namespace nginx-ns и применить созданные в файле `nginx.yaml` манифесты:

```
# kubectl create ns nginx-ns
namespace/nginx-ns created
# kubectl  apply -f nginx.yaml
deployment.apps/nginx-deployment created
service/nginx created
```

Дождаться состояния `1/1` для PODов `nginx-deployment-...`:

```
# kubectl  get all -n nginx-ns
NAME                                    READY    STATUS    RESTARTS   AGE
pod/nginx-deployment-5d54559f98-ffv49   1/1      Running   0          19s
pod/nginx-deployment-5d54559f98-nxlkm   1/1      Running   0          19s

NAME            TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
service/nginx   NodePort   10.103.32.218   <none>        80:32338/TCP   19s

NAME                                  READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deployment   2/2     2            2           19s

NAME                                          DESIRED  CURRENT   READY   AGE
replicaset.apps/nginx-deployment-5d54559f98 2        2         2       19s
```

12) Если поднят единственный master-узел не забудьте предварительно снять ограничение на запуск обычных PODов на master-узле:

```
# kubectl taint nodes <host>  node-role.kubernetes.io/control-plane:NoSchedule-
```

4.4.6. podsec-k8s-rbac – поддержка и управление доступом на основе ролей (RBAC)

В пакет podsec-k8s-rbac входит набор скриптов для работы с RBAC – Role Based Access Control:

- podsec-k8s-rbac-create-user – создание RBAC-пользователя;
- podsec-k8s-rbac-create-kubeconfig – создание ключей, сертификатов и файла конфигурации RBAC-пользователя;
- podsec-k8s-rbac-create-remoteplace – создание удаленного рабочего места;
- podsec-k8s-rbac-bindrole – привязывание пользователя к кластерной или обычной роли;
- podsec-k8s-rbac-get-userroles – получить список кластерных и обычных ролей пользователя;
- podsec-k8s-rbac-unbindrole – отвязывание пользователя от кластерной или обычной роли.

4.4.6.1. podsec-k8s-rbac-create-user – создание RBAC-пользователя

Формат:

```
podsec-k8s-rbac-create-user имя_пользователя
```

Описание: скрипт:

- создает RBAC пользователя;

- создает в домашнем директории каталог `.kube`;
- устанавливаются соответствующие права доступа к каталогам.

4.4.6.2. podsec-k8s-rbac-create-kubeconfig – создание ключей, сертификатов и файла конфигурации RBAC-пользователя

Формат:

```
podsec-k8s-rbac-create-kubeconfig имя_пользователя[@<имя_удаленного_пользователя>] [группа ...]
```

Описание: скрипт должен вызываться администратором безопасности средства контейнеризации.

Для rootless решения имя удаленного пользователя принимается u7s-admin.

Для rootfull решения необходимо после символа @ указать имя удаленного пользователя.

Скрипт в каталоге `~имя_пользователя/.kube` производит:

- создание личного (private) ключа пользователя (файл имя_пользователя.key);
- создание запроса на подпись сертификата (CSR) (файл имя_пользователя.key);
- запись запроса на подпись сертификата CSR в кластер;
- подтверждение запроса на подпись сертификата (CSR);
- создание сертификата (файл `имя_пользователя.crt`);
- проверку корректности сертификата;
- формирование файла конфигурации пользователя (файл `config`);
- добавление контекста созданного пользователя.

4.4.6.3. podsec-k8s-rbac-create-remoteplace – создание удаленного рабочего места

Формат:

```
podsec-k8s-rbac-create-remoteplace ip-адрес
```

Описание: скрипт производит настройку удаленного рабочего места пользователя путем копирования его конфигурационного файла.

4.4.6.4. podsec-k8s-rbac-bindrole – привязывание пользователя к кластерной или обычной роли

Формат:

```
podsec-k8s-rbac-bindrole имя_пользователя role|role=clusterrole|clusterrole роль имя_связки_роли [namespace]
```

Описание: скрипт производит привязку пользователя к обычной или кластерной роли используя `имя_связки_роли`.

Параметры:

- `имя_пользователя` должно быть создано командой `podsec-k8s-rbac-create-user` и сконфигурировано на доступ к кластеру командой `podsec-k8s-rbac-create-kubeconfig`;
- тип роли может принимать следующие значения:
  а) `role` – пользователь привязывется к обычной роли с именем `<роль>` (параметр `namespace` в этом случае обязателен);
  б) `role=clusterrole` – пользователь привязывется к обычной роли используя кластерную роль с именем `<роль>` (параметр `namespace` в этом случае обязателен);
  в) `clusterrole` – пользователь привязывется к кластерной роли используя кластерную роль с именем `<роль>` (параметр `namespace` в этом случае должен отсутствовать);
- `роль` – имя обычной или кластерной роли в зависимости от предыдущего параметра;
- `имя_связки_роли` – имя объекта класса rolebindings или clusterrolebindings в зависимости от параметра тип роли. В рамках этого объекта к кластерной или обычной роли могут быть привязаны несколько пользователей;
- `namespace` – имя `namespace` для обычной роли.

4.4.6.5. podsec-k8s-rbac-get-userroles – получить список кластерных и обычных ролей пользователя

Формат:

```
podsec-k8s-rbac-get-userroles имя_пользователя [showRules]
```

Описание: скрипт формирует список кластерных и обычных ролей которые связаны с пользователем. При указании флага showRules, для каждой роли указывается список правил (`"rules:[...]"`), которые принадлежат каждой роли пользователя.

Результат возвращается в виде json-строки формата:

```
{
  "": {
    "clusterRoles": [...],
    "roles": {
      "allNamespaces": [...],
      "namespaces": [
        {
          "": [...],
          ...
        }
    }
  }
}
```

где `[...]` – массив объектов типа:

```
{
  "bindRoleName": "",
  "bindedRoleType": "ClusterRole|Role",
  "bindedRoleName": "",
  "unbindCmd": "podsec-k8s-rbac-unbindrole ..."
}
```

4.4.6.6. podsec-k8s-rbac-unbindrole – отвязывание пользователя от кластерной или обычной роли

Формат:

`podsec-k8s-rbac-unbindrole имя_пользователя role|clusterrole роль имя_связки_роли [namespace]`

Описание: скрипт производит отвязку роли от кластерной или обычной роли, созданной командой `podsec-k8s-rbac-bindrole`. Полный текст команды можно получить в выводе команды `podsec-k8s-rbac-get-userroles` в поле `unbindCmd`. Если в указанном `имя_связки_роли` объекте класса `rolebindings` или `clusterrolebindings` еще остаются пользователи – объект модифицируется. Если список становится пуст – объект удаляется.

Параметры:

- `имя_пользователя` должно быть создано командой `podsec-k8s-rbac-create-user` и сконфигурировано на доступ к кластеру командой `podsec-k8s-rbac-create-kubeconfig`;
- тип роли может принимать следующие значения:
  а) `role` – пользователь привязывается к обычной роли с именем <роль> (параметр namespace в этом случае обязателен);
  б) `clusterrole` – пользователь привязывается к кластерной роли используя кластерную роль с именем <роль> (параметр namespace в этом случае должен отсутствовать);
- `роль` – имя обычной или кластерной роли в зависимости от предыдущего параметра;
- `имя_связки_роли` – имя объекта класса `rolebindings` или `clusterrolebindings` в зависимости от параметра тип роли. В рамках этого объекта к кластерной или обычной роли могут быть привязаны несколько пользователей;
- `namespace` – имя namespace для обычной роли.

4.4.7. podsec-inotify – мониторинг безопасности системы

В пакет podsec-inotify входит набор скриптов для мониторинга безопасности системы:

- podsec-inotify-check-policy – проверка настроек политики контейнеризации на узле;
- podsec-inotify-check-containers – проверка наличия изменений файлов в директориях rootless контейнерах;
- podsec-inotify-check-images – проверка образов на предмет их соответствия настройкам политики контейнеризации на узле;
- podsec-inotify-check-kubeapi – мониторинг аудита API-интерфейса kube-apiserver control-plane узла;
- podsec-inotify-check-vuln – мониторинг docker-образов узла сканером безопасности trivy.

4.4.7.1. Настройка сервиса trivy

Часть скриптов мониторинга для обнаружения уязвимостей использует сканер trivy.

Сканер безопасности trivy работает как клиент сервера trivy, принимающего соединения по порту 4954 на узле с доменом trivy.local. Если узел работает в составе кластера, то необходимо:

- на одном из узлов кластера поднять сервер trivy командой:

```
systemctl enable --now trivy
```

- на всех узлах кластера прописать в файле `/etc/hosts` строку:

```
<IP-адрес_узла_сервера_trivy> trivy.local
```

Если узел вне кластера необходимо:

- на узле поднять сервер trivy командой:

```
systemctl enable --now trivy
```

- прописать в файле `/etc/hosts` строку:

```
127.0.0.1 trivy.local
```

На платформе вида c10f сервер trivy запускается автоматически скриптом podsec-create-services на master-сервере кластера, привязка домена `trivy.local` к IP-адресу сервера производится автоматически скриптом podsec-create-policy.

Подробнее о trivy приведено в документе «Руководство администратора. ЛКНВ.11100-01 90 03».

4.4.7.2. Мониторинг сообщений об уязвимостях

Для передачи сообщений серверу мониторинга общей инфраструктуры icinga необходимо поднять сервис nagwad:

```
# apt-get install nagwad-service
# systemctl enable --now nagwad
```

Все сообщения об обнаруженных уязвимостях скрипты записывают в системный лог в следующем формате:

```
<месяц> <день> <время> <host> <имя_скрипта>[<id>]: <уровень_уязвимости>: <текст_сообщения>
```

Посмотреть эти сообщения можно командой:

```
journalctl -t <имя_скрипта>
```

Например:

```
journalctl -t podsec-inotify-check-vuln
```

```
июл 16 06:22:36 host-136 podsec-inotify-check-vuln[383501]: Critical: В образе registry.altlinux.org/k8s-sisyphus/kube-apiserver:v1.30.1 пользователя u7s-admin обнаружены критические и высокие уязвимости.
...
```

В файловой системе будет создан каталог `/var/log/nagwad/<boot_uid>/podsec/`. Все сообщения об уязвимостях сервис nagwad будет записывать из системного лога в данный каталог в файлы под именем `podsec.<message_id>.<level>`.

Например, `/var/log/nagwad/3c22e4b3-d4d7-4975-a49c-f630a15c041d/podsec/`:

```
CRITICAL: podsec-inotify-check-vuln(Critical) В образе registry.altlinux.org/k8s-sisyphus/kube-apiserver:v1.30.1 пользователя u7s-admin обнаружены критические и высокие уязвимости.
```

Эти файлы в дальнейшем передаются серверу мониторинга общей инфраструктуры icinga.

4.4.7.3. podsec-inotify-check-policy – проверка настроек политики контейнеризации на узле

Формат:

```
podsec-inotify-check-policy [-v[vv]] [-a интервал] [-f интервал] -c интервал -h интервал [-m интервал] x-w интервалъ [-l интервал] [-d интервал]
```

Описание: плагин проверяет настройки политики контейнеризации на узле.

Проверка идет по следующим параметрам:

- файл `policy.json` установки транспортов и политик доступа к регистраторам (таблица 6);

Т а б л и ц а 6

Параметр контроля пользователей	Вес метрики
имеющих `defaultPolicy != reject`, но не входящих в группу podman_dev	102
не имеющих `registry.local` в списке регистраторов для которых проверяется наличие электронной подписи образов	103
имеющих в политике регистраторы для которых не проверяется наличие электронной подписи образов	104
имеющих в списке поддерживаемых транспорты отличные от docker (транспорт получения образов с регистратора)	105

- файлы привязки регистраторов к серверам, хранящим электронные подписи (файл привязки по умолчанию `default.yaml` и файлы привязки регистраторов `*.yaml` каталога `registries.d`). Наличие (число) пользователей (таблица 7);

Таблица 7

Параметр контроля пользователей	Вес метрики
не использующих хранилище подписей `http://sigstore.local:81/sigstore/` как хранилище подписей по умолчанию	106

- контроль групп пользователей:

  а) наличие пользователей, имеющих образы, но не входящих в группу podman (таблица 8);

Таблица 8

Параметр контроля пользователей	Вес метрики
наличие пользователей, имеющих образы, но не входящих в группу podman	101

  б) наличие пользователей группы podman (за исключением входящих в группу podman_dev) (таблица 9).

Таблица 9

Параметр контроля пользователей	Вес метрики
входящих в группу wheel	101
имеющих каталог `.config/containers/` открытым на запись и изменения	90 * доля_нарушителей
не имеющих файлконфигурации `.config/containers/storage.conf`	90 * доля_нарушителей

`доля_нарушителей` считается как:

`число_нарушителей / число_пользователей_группы_podman`

Все веса метрик суммируются и формируется итоговая метрика.

4.4.7.4. podsec-inotify-check-containers – проверка наличия изменений файлов в директориях rootless контейнерах

Формат:

```
podsec-inotify-check-containers
```

Описание: скрипт:

- создает список директорий rootless контейнеров, существующих в системе;
- запускает проверку на добавление, удаление и изменение файлов в директориях контейнеров;
- отсылает уведомление об изменении в системный лог.

4.4.7.5. podsec-inotify-check-images – проверка образов на предмет их соответствия настройкам политики контейнеризации на узле

Формат:

```
podsec-inotify-check-images [-v[vv]] [-a интервал] [-f интервал] -c интервал -h интервал [-m интервал] x-w интервалъ [-l интервал] [-d интервал]
```

Описание: плагин проверяет образы на предмет их соответствия настройкам политики контейнеризации на узле. Проверка идет по параметрам, указанным в таблице 10.

Т а б л и ц а 10

Параметр контроля пользователей	Вес метрики
наличие в политике пользователя регистраторов не поддерживающие электронную подпись	101
наличие в кэше образов неподписанных образов	101
наличие в кэше образов вне поддерживаемых политик	101

Все веса метрик суммируются и формируется итоговая метрика.

4.4.7.6. podsec-inotify-check-kubeapi – мониторинг аудита API-интерфейса kube-apiserver control-plane узла

Формат:

```
podsec-inotify-check-kubeapi [-d]
```

Описание: скрипт производит мониторинг файла `/etc/kubernetes/audit/audit.log` аудита API-интерфейса kube-apiserver.

Политика аудита располагается в файле `/etc/kubernetes/audit/policy.yaml`:

```
apiVersion: audit.k8s.io/v1
kind: Policy
omitManagedFields: true
rules:
# do not log requests to the following
- level: None
  nonResourceURLs:
  - "/healthz*"
  - "/logs"
  - "/metrics"
  - "/swagger*"
  - "/version"
  - "/readyz"
  - "/livez"

- level: None
  users:
    - system:kube-scheduler
    - system:kube-proxy
    - system:apiserver
    - system:kube-controller-manager
    - system:serviceaccount:gatekeeper-system:gatekeeper-admin

- level: None
  userGroups:
    - system:nodes
    - system:serviceaccounts
    - system:masters

# limit level to Metadata so token is not included in the spec/status
- level: Metadata
  omitStages:
  - RequestReceived
  resources:
  - group: authentication.k8s.io
    resources:
    - tokenreviews

# extended audit of auth delegation
- level: RequestResponse
  omitStages:
  - RequestReceived
  resources:
  - group: authorization.k8s.io
    resources:
```

```
      - subjectaccessreviews

  # log changes to pods at RequestResponse level
  - level: RequestResponse
    omitStages:
    - RequestReceived
    resources:
    - group: "" # core API group; add third-party API services and
your API services if needed
      resources: ["pods"]
      verbs: ["create", "patch", "update", "delete"]

  # log everything else at Metadata level
  - level: Metadata
    omitStages:
    - RequestReceived
```

Текущие настройки производят логирование всех обращений «несистемных» пользователей (в том числе анонимных) к ресурсам kubernetes.

Скрипт производит выборку всех обращений, в ответ на которые был сформирован код более 400 – запрет доступа. Все эти факты записываются в системный журнал и накапливаются в файле логов `/var/lib/podsec/u7s/log/kubeapi/forbidden.log`, который периодически передается через почту системному администратору.

Параметры:

- `-d` – скрипт запускается в режиме демона, производящего онлайн мониторинг файла `/etc/kubernetes/audit/audit.log` и записывающего факты запросов с запретом доступа в системный журнал и файл логов `/var/lib/podsec/u7s/log/kubeapi/forbidden.log`;
- при запуске без параметров скрипт посылает файл логов `/var/lib/podsec/u7s/log/kubeapi/forbidden.log` почтой системному администратору (пользователю root) и обнуляет файл логов.

В состав пакета кроме этого скрипта входит файл описания сервиса `/lib/systemd/system/podsec-inotify-check-kubeapi.service`. Для его запуска необходимо выполнить команды:

```
# systemctl enable  podsec-inotify-check-kubeapi.service
# systemctl start  podsec-inotify-check-kubeapi.service
```

4.4.7.7. podsec-inotify-check-vuln – мониторинг docker-образов узла сканером безопасности trivy

Формат:

```
podsec-inotify-check-vuln
```

Описание: скрипт производит мониторинг docker-образов узла сканером безопасности trivy:

- если скрипт запускается от имени пользователя root скрипт:
  - а) проверяет сканером trivy rootfull образы;
  - б) для всех пользователей каталога `/home/` проверяется наличие rootless-образов. При их наличии проверяет сканером trivy эти образы;
- если скрипт запускается от имени обычного пользователя проверяется наличие rootless-образов. При их наличии проверяет сканером trivy эти образы.

Результат анализа посылается в системный лог. Если при анализе образа число обнаруженных угроз уровня HIGH больше 0, результат посылается почтой системному администратору (root).

Параметры: отсутствуют.

Периодический запуск скрипта: в состав пакета кроме этого входит systemd/timers файл `/lib/systemd/system/podsec-inotify-check-vuln.timer`. При его активации командой:

```
systemctl enable podsec-inotify-check-vuln.timer
```

каждый час запускается скрипт мониторинга.

Период запуска можно указать в `OnCalendar` вышеуказанного systemd/timers файла.

4.5. Проверка работоспособности kubernetes в rootless режиме

П р и м е ч а н и е . Проверка работоспособности kubernetes в rootless режиме проводится после выполнения настройки в соответствии с п. 4.4.

Перед проверкой работоспособности kubernetes необходимо выполнить загрузку образа в регистратор, для примера используется образ nginx.

Зайдите в систему под пользователем imagemaker.

Загрузка исходного образа:

```
$ podman pull --tls-verify  docker.io/library/nginx:1.14.2
Trying to pull docker.io/library/nginx:1.14.2...
Getting image source signatures
Copying blob 8ca774778e85 skipped: already exists
Copying blob 27833a3ba0a5 skipped: already exists
Copying blob 0f23e58bd0b7 skipped: already exists
Copying config 295c7be079 done
Writing manifest to image destination
Storing signatures
295c7be079025306c4f1d65997fcf7adb411c88f139ad1d34b537164aa060369
```

Создание alias'a для помещения на локальный регистратор:

```
$ podman  tag docker.io/library/nginx:1.14.2 registry.local/nginx
```

Помещение на локальный регистратор:

```
$ podman push --tls-verify=false   --sign-by='<EMAIL>'   registry.local/nginx
Getting image source signatures
Copying blob 5dacd731af1b done
Copying blob 82ae01d5004e done
Copying blob b8f18c3b860b done
Copying config 295c7be079 done
Writing manifest to image destination
Creating signature: Signing image using simple signing
Storing signatures
```

Во время помещения образа (если прошло достаточно много времени после последнего `podman push`) необходимо ввести пароль для подписи.

Выполните запуск образов ngix в виде deployment.

Под пользователем администратор (root):

- создайте манифест `deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2 # tells deployment to run 2 pods matching the template
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
```

```
        - name: nginx
          image: registry.local/nginx
          ports:
          - containerPort: 80
    ---
    apiVersion: v1
    kind: Service
    metadata:
      name: nginx
      labels:
        app: nginx
    spec:
      type: NodePort
      ports:
      - port: 80
        targetPort: 80
      selector:
        app: nginx
```

- запустите deployment:

```
# kubectl apply -f deployment.yaml
```

- дождитесь разворачивания deployment и POD'ов:

```
# kubectl  get pods,service -o wide
```

```
NAME                                   READY  STATUS   RESTARTS AGE   IP           NODE
NOMINATED NODE   READINESS GATES
pod/nginx-deployment-7f688b6459-h2p7k 1/1    Running  0        20s   10.244.0.4   host-99
pod/nginx-deployment-7f688b6459-sz86q 1/1    Running  0        20s   10.244.1.2   host-26

NAME                 TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)     AGE   SELECTOR
service/kubernetes ClusterIP  10.96.0.1      443/TCP                   42m
service/nginx      NodePort   10.111.222.98  80:31280/TCP              20s   app=nginx
```

На одном из узлов, где развернулся POD, зайдите под пользователем u7s-admin и перейдите в namespace пользователя:

```
$ nsenter_u7s
```

Выберите любой из IP-адресов интерфейсов `tap0` или `cni0`:

```
# ip a show dev tap0
2: tap0:  mtu 65520 qdisc fq_codel state UP group default qlen 1000
    link/ether 12:98:24:5b:ac:8d brd ff:ff:ff:ff:ff:ff
    inet 10.96.122.26/32 scope global tap0
# ip a show dev cni0
4: cni0:  mtu 65470 qdisc noqueue state UP group default qlen 1000
    link/ether 7e:8e:4e:7f:f7:5c brd ff:ff:ff:ff:ff:ff
    inet 10.244.1.1/24 brd 10.244.1.255 scope global cni0
```

Обратитесь к сервису nginx по выбранному IP-адресу и выделенному порту (31280):

```
# curl http://10.244.1.1:31280
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully
installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Если необходим доступ к данному порту из внешней сети, необходимо выполнить проброс порта:

```
# rootlessctl add-ports 0.0.0.0:31280:31280/tcp
```

Запросите доступ к Pod'y nginx по внешнему порту:

```
# curl http://192.168.10.11:31280
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
```

5. УДАЛЕННОЕ ПОДКЛЮЧЕНИЕ К ВМ

5.1. VNC подключение к ВМ

К консоли или рабочему столу ВМ KVM можно удаленно подключиться по протоколу VNC. Настройка удаленного доступа к ВМ KVM по протоколу VNC выполняется с помощью менеджера ВМ (virt-manager).

В virt-manager необходимо открыть панель свойств ВМ и раздел «Дисплей Spice». В поле тип выберите VNC-сервер (рис. 44).

VNC-сервер, по умолчанию, обслуживает только локальные VNC-запросы (Только Localhost). При таких параметрах удаленный доступ к ВМ будет запрещен.

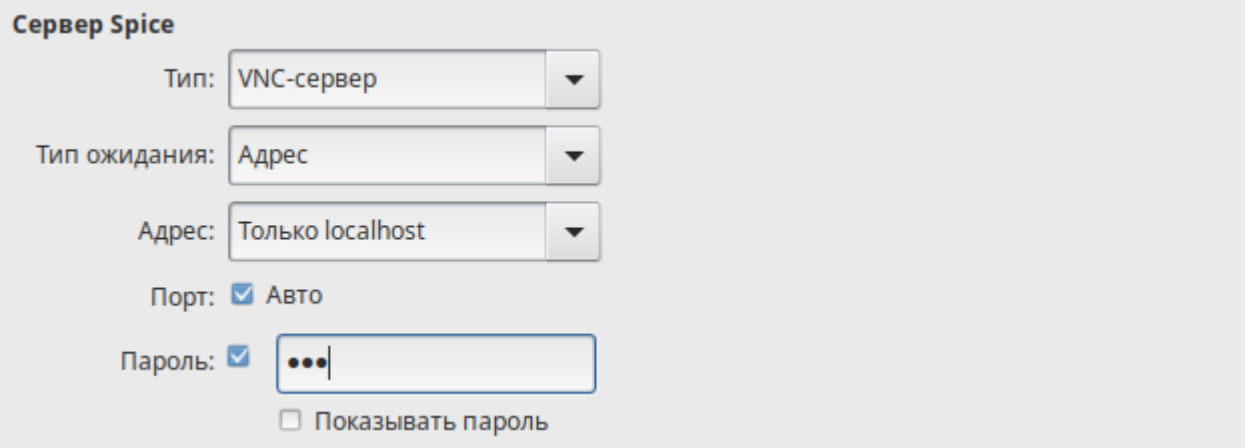

Рис. 44 – VNC-сервер

Для удаленного подключения к ВМ KVM по VNC-протоколу VNC-сервер хост системы должен обслуживать запросы с общедоступных сетевых интерфейсов. Для разрешения удаленного доступа необходимо в поле тип выбрать «Сервер SPICE» и на месте адреса выбрать «Все интерфейсы».

Если на хосте несколько ВМ, к каждой из них можно будет подключиться через один IP-адрес и разные порты. Порт доступа к ВМ может быть назначен вручную или автоматически. Удаленный доступ к ВМ можно защитить паролем.

5.1.1. Подключение VNC-клиента к удаленному компьютеру

Для подключения VNC-клиента к удаленному компьютеру требуется указать его IP-адрес или DNS-имя, и номер дисплея (по умолчанию, 0) или номер TCP-порта (по умолчанию, `5900`). Если VNC-сервер требует авторизации, то при подключении к нему VNC-клиент запросит пароль. Пароль доступа к VNC-серверу не связан с каким-либо аккаунтом (учетной записью пользователя) на удаленном компьютере, а служит только для ограничения доступа к дисплею VNC-сервера.

После установки соединения и открытия экрана, в зависимости от настроек VNC-сервера, может потребоваться авторизация пользователя на виртуальном сервере или может быть открыта уже запущенная рабочая сессия какого-либо пользователя.

Так как на компьютере одновременно могут работать несколько VNC-серверов, для их разделения используют параметр номер дисплея. Например, один VNC-сервер может быть запущен на дисплее `:0`, другой – на дисплее `:1`. Каждому номеру дисплея соответствует номер TCP-порта, на котором VNC-сервер принимает соединения. Номер порта для дисплея получается прибавлением номера дисплея к базовому номеру порта – `5900`. Дисплею `:0` соответствует TCP-порт `5900`, дисплею `:1` – порт `5901`.

5.1.2. Отключение VNC-клиента от удаленного компьютера

При закрытии окна VNC-клиента или после выхода из окружения средствами рабочего стола, в зависимости от настроек VNC-сервера, рабочая сессия пользователя может закрыться с остановкой всех используемых программ, или продолжать работу и быть доступной снова при повторном подключении к VNC-серверу.

5.2. SPICE подключение к ВМ

К консоли или рабочему столу ВМ KVM, можно удаленно подключиться по протоколу SPICE.

SPICE (Simple Protocol for Independent Computing Environments) – открытый протокол удаленного доступа к компьютеру или ВМ.

Протокол SPICE состоит из следующих элементов:

- SPICE-сервер – обычно QEMU/KVM гипервизор;
- SPICE-клиент – клиент удаленного доступа;
- SPICE-агент – гостевое дополнение, расширяющее интеграцию гостя и хоста;
- SPICE-сервер – графическая подсистема гипервизора.

Чтобы задействовать протокол SPICE в QEMU/KVM необходимо подключить протокол удаленного доступа к графической подсистеме SPICE.

Настройка удаленного доступа к ВМ KVM по протоколу SPICE выполняется с помощью менеджера ВМ (virt-manager).

В virt-manager нужно открыть панель свойств ВМ и раздел «Дисплей Spice». В поле тип необходимо выбрать «Сервер spice» (рис. 45).

SPICE-сервер, по умолчанию, обслуживает только локальные SPICE-запросы (только Localhost). При таких параметрах удаленный доступ к ВМ будет запрещен.

Для удаленного подключения к ВМ KVM по SPICE-протоколу SPICE-сервер хост системы должен обслуживать запросы с общедоступных сетевых интерфейсов. Для разрешения удаленного доступа на месте адреса выберите «Все интерфейсы».

Порт доступа к ВМ может быть назначен вручную или автоматически. Удаленный доступ к ВМ можно защитить паролем.

SPICE-агент – дополнительный механизм связи гостя с хостом. Предоставляет следующие возможности:

- передача доступа к мыши гостю;
- совместная работа с буфером обмена.

Также необходимо в конфигурацию добавить SPICE-канал (`spicevmc`) для обеспечения связи между хостом и гостем. Это можно сделать в `virt-manager`.

Сохраните настройки, нажав на кнопку «Готово».

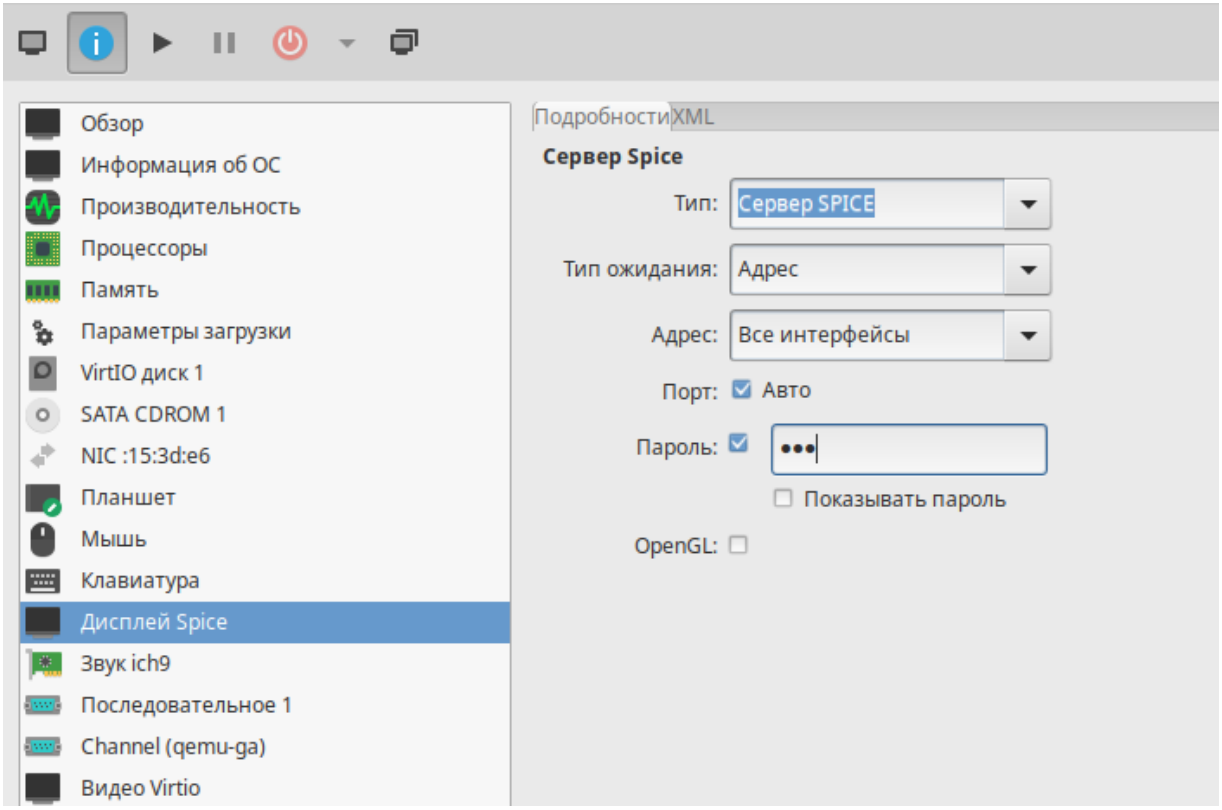

Рис. 45 – Раздел «Дисплей Spice»

5.3. Проброс USB-устройств в ВМ через SPICE

Протокол удаленного доступа SPICE позволяет не только передавать данные устройств ввода-вывода, но и передавать по сети трафик USB-устройств – пробрасывать USB-устройства клиента без использования дополнительных программ USB-серверов, таких как usbip (см. подраздел «USB/IP» в документе «Руководство администратора. ЛКНВ.11100-01 90 03»).

Проброс устройств может использоваться для получения и передачи данных на удаленные устройства из ВМ, например, на принтеры, USB-ключи, FLASH-накопители и другие низкоскоростные устройства.

Для настройки возможности проброса устройства необходимо разместить на сервере ВМ KVM под управлением SPICE с поддержкой USB redirect. В самой ВМ установка гостевых дополнений не требуется. На ВМ клиента необходимо установить Linux и SPICE и получить права администратора.

Далее с помощью SPICE клиента, например, virt-viewer, требуется осуществить подключение к ВМ.

Для установки выполните:

```
# apt-get install virt-viewer
```

Запустить данный клиент: «Приложения» → «Интернет» → «Удаленный рабочий стол».

Введите адрес подключения, например:

```
spice://<ip адрес или доменное имя компьютера>:<номер порта>
```

Введите пароль, если SPICE-сервер требует авторизации.

После чего пробросьте USB-устройство через меню «Выбор USB-устройств для перенаправления».

Отключение устройств осуществляется аналогичным способом.

6. АУДИТ СОБЫТИЙ БЕЗОПАСНОСТИ

В средстве контейнеризации/виртуализации должен быть определен перечень событий, необходимых для регистрации и учета.

Регистрации подлежат как минимум следующие события безопасности:

- неуспешные попытки аутентификации пользователей средства контейнеризации/виртуализации;
- создание, модификация и удаление образов контейнеров/виртуальных машин;
- получение доступа к образам контейнеров/виртуальным машинам;
- запуск и остановка контейнеров/виртуальных машин с указанием причины остановки;
- изменение ролевой модели;
- модификация запускаемых контейнеров;
- изменение конфигурации средства виртуализации/виртуальных машин;
- выявление известных уязвимостей в образах контейнеров и некорректности конфигурации;
- факты нарушения целостности объектов контроля.

6.1.1. Аудит средств виртуализации

Для просмотра записей журнала ОС, связанных с нарушением целостности объектов контроля и иных событий безопасности может использоваться система сигнализации на основе icinga. Оповещения о нарушении целостности и иных событиях доступны администратору в веб-интерфейсе icinga2 после авторизации.

Примечание. Настройка системы сигнализации на основе icinga выполняется в соответствии с документом «Руководство администратора. ЛКНВ.11100-01 90 03».

Также для просмотра аудита событий безопасности ОС, связанных со средствами виртуализации, для анализа и отладки работы системных компонентов может использоваться утилита journalctl, далее в примерах рассмотрены примеры команд.

Подробнее о команде `journalctl` приведено в документе «Руководство по комплексу средств защиты. ЛКНВ.11100-01 99 03».

6.1.1.1. Примеры команд journalctl

Для просмотра записей журнала выполнить авторизацию в роли Администратора от имени учетной записи root в окне аутентификации консольного интерфейса ОС, выполнить поиск, например, следующими командами:

- неуспешные попытки аутентификации пользователей средств виртуализации:

```
# journalctl -r | grep pam_tcb
# journalctl -r | grep pam
# journalctl -r | grep pvedaemon
```

- создание, модификация и удаление виртуальных машин:

```
# journalctl -r | grep image
# journalctl -r | grep build
# journalctl -r | grep qmcreate
# journalctl -r | grep qmdestroy
```

- поиск по степени критичности события безопасности:

```
# journalctl -p alert -o verbose
# journalctl -b  -p 3 -o verbose
```

- изменение ролевой модели:

```
# journalctl -r | grep pverbac
```

6.1.2. Аудит средств контейниризации

6.1.2.1. Настройки icinga для средств контейниризации

Для аудита событий средств контейнеризации можно настроить АРМ «Рабочее место контролера событий безопасности» согласно документу «Руководство администратора. ЛКНВ.11100-01 90 03» подраздел «Настройка системы сигнализации на основе icinga». При настройке использовать icinga-director. АРМ со средствами контейнеризации будет является агентом.

Все административные действия в веб-интерфейсе icinga выполняются от администратора icinga_admin.

1) На АРМ «Рабочее место контролера событий безопасности» необходимо создать учетную запись imagemaker для чтения событий с АРМ со средствами контейнеризации (рис. 46):

```
# useradd imagemaker
# passwd imagemaker
```

```
[root@master1 ~]# useradd imagemaker
[root@master1 ~]# passwd imagemaker
passwd: updating all authentication tokens for user imagemaker.

You can now choose the new password or passphrase.

A valid password should be a mix of upper and lower case letters, digits, and
other characters.  You can use a password containing at least 7 characters
from all of these classes, or a password containing at least 8 characters
from just 3 of these 4 classes.
An upper case letter that begins the password and a digit that ends it do not
count towards the number of character classes used.

A passphrase should be of at least 3 words, 11 to 72 characters long, and
contain enough different characters.

Alternatively, if no one else can see your terminal now, you can pick this as
your password: "Merge2Call4Scale".

Enter new password:
Weak password: too short.
Re-type new password:
passwd: all authentication tokens updated successfully.
```

Рис. 46 – Создание учетной записи imagemaker

2) На АРМ «Рабочее место контролера событий безопасности» после успешного разворачивания icinga и добавления АРМ со средствами контейнеризации в качестве агента необходимо развернуть корзину podsec. Для этого необходимо выполнить установку пакета podsec-icinga:

```
# apt-get install podsec-icinga
```

Затем развернуть корзину podsec как описано в документе «Руководство администратора. ЛКНВ.11100-01 90 03» подраздел «Импорт конфигурации Director из корзины». Корзина находится в:

```
/usr/share/doc/podsec-icinga-1.1.6/podsec-icinga2.json.
```

3) После успешного разворачивания корзины на АРМ «Рабочее место контролера событий» необходимо добавить службы podsec на АРМ со средствами контейнеризации (рис. 47). Для этого перейти в раздел «Управление Icinga» затем в нем выбрать раздел «Узлы», выбрать АРМ со средствами контейнеризации в правом разделе меню перейти на вкладку «Службы», нажать «Добавить набор

служб» и из выпадающего списка выбрать «d-nagwad-podsec-set» и нажать «Добавить».

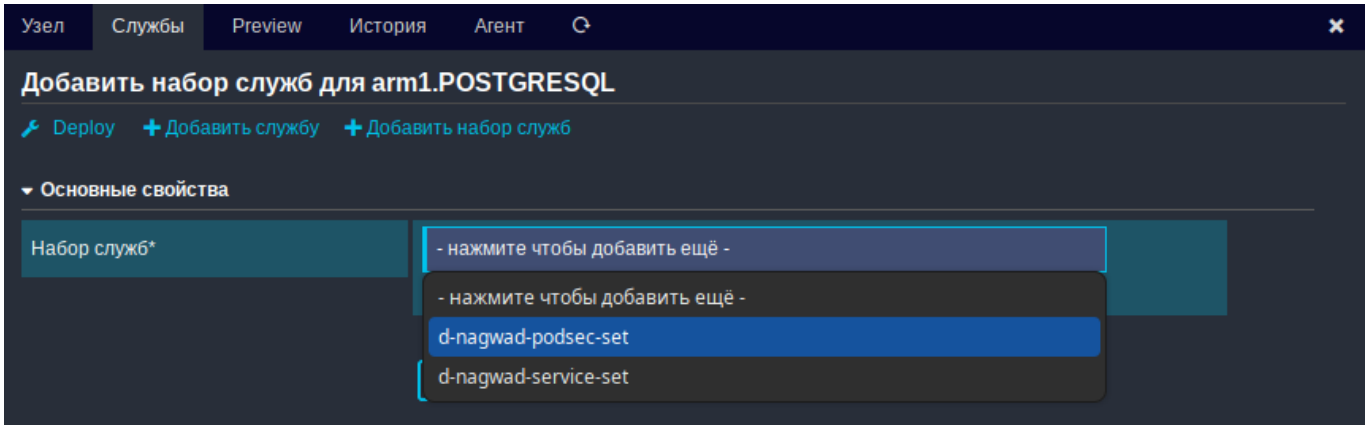

Рис. 47 – Добавление служб podsec на АРМ со средствами контейнеризации

4) На АРМ «Рабочее место контролера событий» создать пользователя imagemaker в icinga и назначить ему роль, разрешающую отслеживание событий только с агента, связанного со средствами контейнеризации, для этого необходимо:

- перейти в раздел «Настройки» → «Контроль доступа». Перейти на вкладку «Пользователи» – добавить нового пользователя и задать ему пароль (рис. 48, рис. 49);

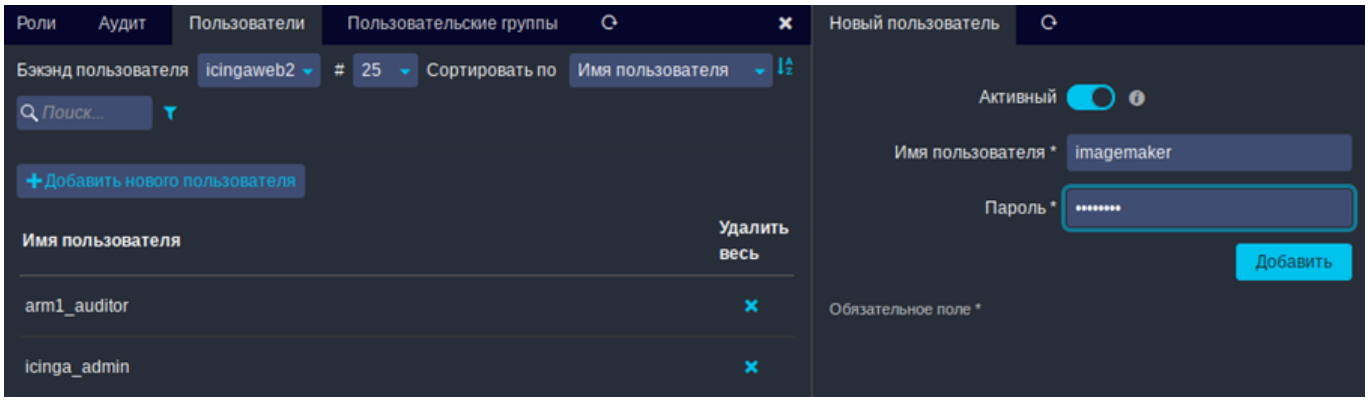

Рис. 48 – Добавление пользователя в icinga

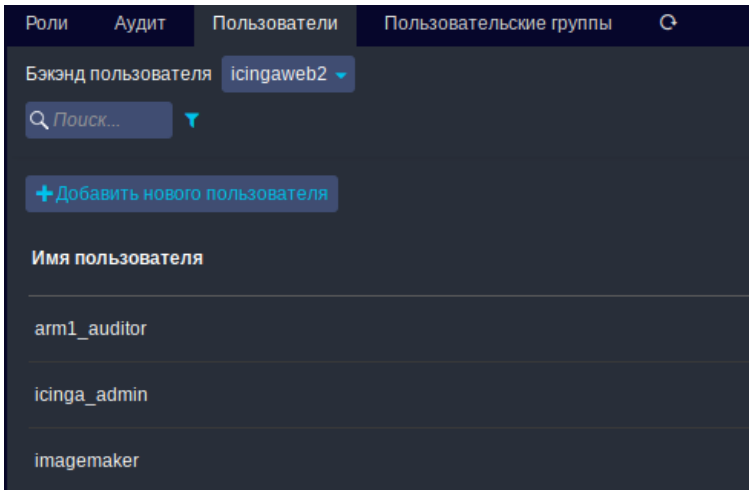

Рис. 49 – Добавлен пользователь imagemaker

- перейти на вкладку «Роли» и нажать «Создание новой роли» (рис. 50);

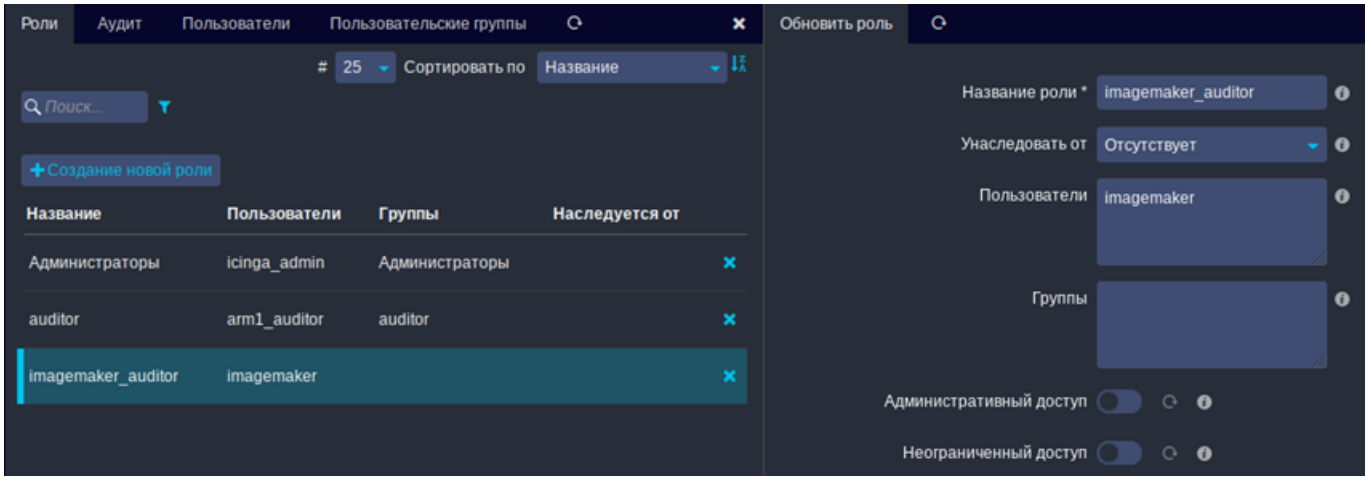

Рис. 50 – Создание новой роли

- в разделе «Разрешения» выбрать модуль monitoring и указать разрешения в виде «Общий доступ к модулю» (рис. 51);

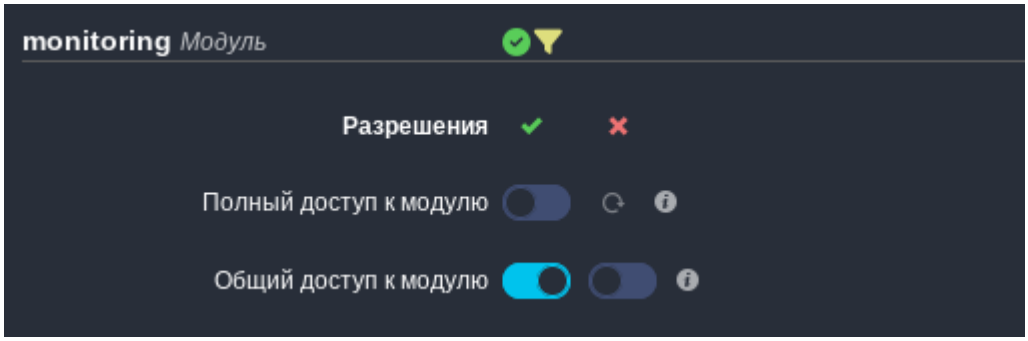

Рис. 51 – Указание разрешений

- разрешить отслеживать только события с хоста (в примере arm1) со средствами контейнеризации – в модуле monitoruing перейти в «Ограничения» и указать значение хоста, например: `host_name=arm1.POSTGRESQL` для поля monitoring/filter/objects (рис. 52).

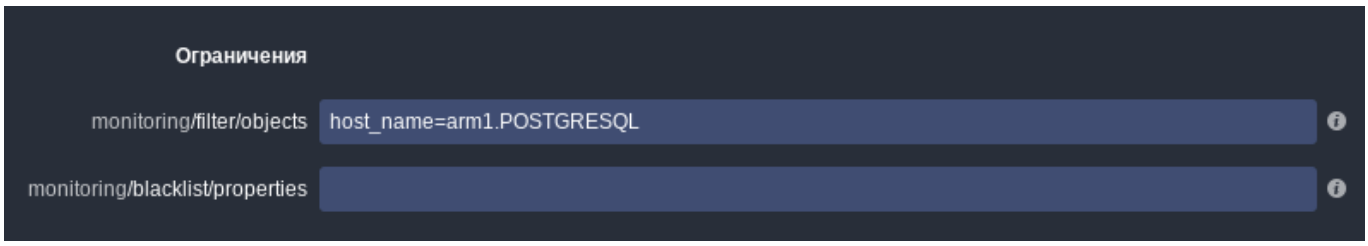

Рис. 52

Для просмотра записей журнала аудита об инцидентах безопасности можно на АРМ «Рабочее место контролера событий безопасности»:

- авторизоваться в системе от имени учетной записи imagemaker;
- затем открыть веб-браузер и перейти в веб-интерфейс icinga;
- авторизоваться в нем от пользователя imagemaker.

Примеры регистрируемых событий:

- неуспешные попытки аутентификации пользователей средства контейнеризации;
- модификация запускаемых контейнеров;
- выявление известных уязвимостей в образах контейнеров и некорректности конфигурации;
- факты нарушения целостности объектов контроля.

6.1.2.2. Примеры команд journalctl

Для просмотра аудита событий безопасности ОС, связанных со средствами контейнеризации, для анализа и отладки работы системных компонентов может использоваться утилита journalctl, далее в примерах рассмотрены примеры команд.

Подробнее о команде `journalctl` приведено в документе «Руководство по комплексу средств защиты. ЛКНВ.11100-01 99 03».

Для просмотра записей журнала выполнить авторизацию в роли Администратора от имени учетной записи root в окне аутентификации консольного интерфейса ОС, выполнить поиск, например, следующими командами:

- неуспешные попытки аутентификации пользователей средств контейнеризации:

```
# journalctl -r | grep pam_tcb
```

- создание образов контейнеров:

```
# journalctl -r | grep build
```

- модификация образов контейнеров:

```
# journalctl -r | grep pull
```

- удаление образов контейнеров:

```
# journalctl -r | grep remove
```

- запуск и остановка контейнеров:

```
# journalctl -r -o json | grep "podman run"
# journalctl -r -o json | grep "podman stop"
# journalctl --user -r | grep start
# journalctl -r | grep "podman.*start"
# journalctl -r | grep "podman.*died"
```

- изменение ролевой модели:

```
# journalctl -r -o json | grep "useradd"
```

- факты нарушения целостности объектов контроля средств контейнеризации:

```
# journalctl -r  | grep 'osec:integalert_container\|integalert'
```

- получение доступа к образам контейнеров:

```
# journalctl -r | grep "podman.*image.*pull"
# journalctl -r | grep pull
```

- модификация запускаемых контейнеров:

```
# journalctl -r -o json | grep "inotify-overlays"
```

Для выявления уязвимостей в образах контейнеров и некорректных конфигураций может использоваться скрипт:

```
# podsec-inotify-check-vuln
```

Для просмотра записей журнала ОС, связанных с мониторингом уязвимостей в образах контейнеров – выполнить авторизацию в роли Администратора от имени учетной записи root в окне аутентификации консольного интерфейса ОС, выполнить команду:

```
# journalctl -r -o json | grep "Total"
```

7. OPENUDS

OpenUDS это многоплатформенный брокер подключений для создания и управления виртуальными рабочими местами.

Основные компоненты решения VDI на базе OpenUDS:

- OpenUDS Server (openuds-server) – брокер подключений пользователей, а также интерфейс администратора для настройки;
- SQL Server. Для работы django-приложения, которым является openuds-server, необходим SQL сервер, например, mysql или mariadb. SQL Server может быть установлен на отдельном сервере;
- платформа для запуска клиентских окружений и приложений. OpenUDS совместима со множеством систем виртуализации: PVE, OpenNebula, oVirt, OpenStack. Также возможно использование с отдельным сервером без виртуализации (аналог терминального решения);
- OpenUDS Client (openuds-client) – клиентское приложение для подключения к брокеру соединений и дальнейшего получения доступа к виртуальному рабочему окружению;
- OpenUDS Tunnel (openuds-tunnel) – решение для туннелирования обращений от клиента к виртуальному рабочему окружению. OpenUDS Tunnel предназначен для предоставления доступа из недоверенных сегментов сети, например, из сети Интернет. Устанавливается на отдельный сервер;
- OpenUDS Actor (openuds-actor) – ПО для гостевых виртуальных машин, реализует связку виртуальной машины и брокера соединений.

Рекомендуемые системные требования для решений на базе OpenUDS приведены в таблице 11.

Т а б л и ц а 11 – Системные требования

Компонент	ОЗУ	ЦП	Диск
OpenUDS Server	2 Гбайт	2 vCPUs	8 Гбайт
SQL Server	1 Гбайт	2 vCPUs	10 Гбайт
OpenUDS Tunnel	2 Гбайт	2 vCPUs	13 Гбайт

П р и м е ч а н и е . Если сервер с базой данных установлен на той же машине, где и OpenUDS Server, требуемое количество памяти нужно просуммировать.

7.1. Установка

7.1.1. Установка базы данных MySQL (MariaDB)

Установить MySQL (MariaDB):

```
# apt-get install mariadb-server
```

Запустить сервер mariadb и добавить его в автозагрузку:

```
# systemctl enable --now mariadb.service
```

Задать пароль root для mysql и настройки безопасности:

```
# mysql_secure_installation
```

Создать базу данных dbuds, пользователя базы данных dbuds с паролем password и предоставить ему привилегии в базе данных dbuds:

```
$ mysql -u root
Enter password:

MariaDB> CREATE DATABASE dbuds CHARACTER SET utf8 COLLATE utf8_general_ci;
MariaDB> CREATE USER 'dbuds'@'%' IDENTIFIED BY 'password';
MariaDB> GRANT ALL PRIVILEGES ON dbuds.* TO 'dbuds'@'%';
MariaDB> FLUSH PRIVILEGES;
MariaDB> exit;
```

7.1.2. Установка OpenUDS Server

Установка OpenUDS Server выполняется следующей командой:

```
# apt-get install openuds-server-nginx
```

Для работы будут установлены следующие пакеты:

- openuds-server – django приложение;
- gunicorn – сервер приложений (обеспечивает запуск django как стандартного WSGI приложения);
- nginx – http-сервер, используется в качестве reverse-proxy для доступа к django приложению, запущенному с помощью gunicorn.

Настройка OpenUDS Server:

- отредактировать файл `/etc/openuds/settings.py`, указав корректные данные для подключения к SQL серверу:

```
DATABASES = {
```

```
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'OPTIONS': {
            'isolation_level': 'read committed',
        },
        'NAME': 'dbuds',
        'USER': 'dbuds',
        'PASSWORD': 'password',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
```

- заполнить базу данных начальными данными:

```
# su -s /bin/bash - openuds
$ cd /usr/share/openuds
$ python3 manage.py migrate
$ exit
```

- запустить gunicorn:

```
# systemctl enable --now openuds-web.service
```

- запустить nginx:

```
# cd /etc/nginx/sites-enabled.d/
#  ln  -s  ../sites-available.d/openuds.conf  /etc/nginx/sites-enabled.d/openuds.conf
# systemctl enable --now nginx.service
```

- запустить менеджер задач OpenUDS:

```
# systemctl enable --now openuds-taskmanager.service
```

ВАЖНО

Перед запуском nginx, необходимо остановить, если она запущена, службу apache2:

```
# systemctl disable --now httpd2
```

Веб-интерфейс OpenUDS будет доступен по адресу `https://IP-адрес_сервера/` (рис. 53).

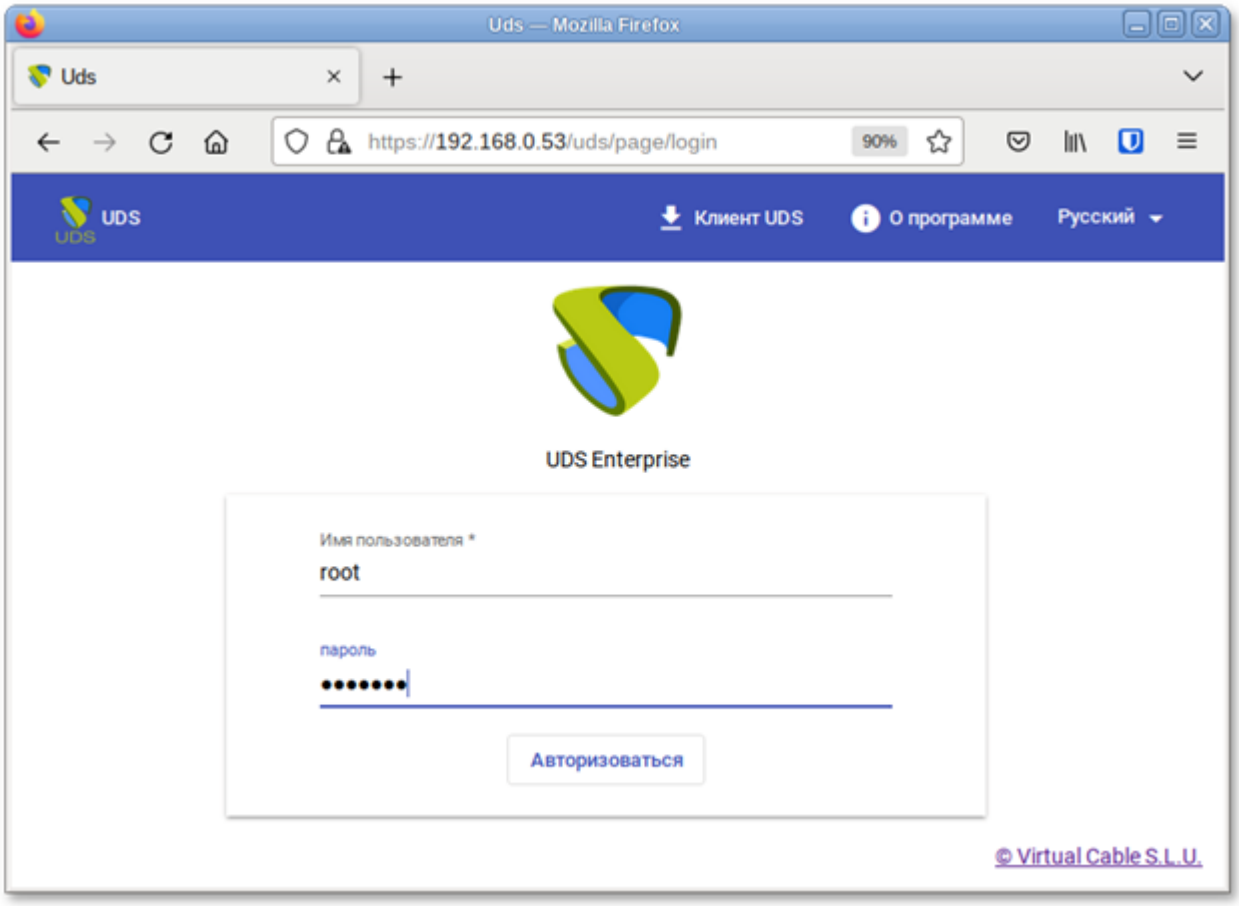

Рис. 53

П р и м е ч а н и я :

1. Имя/пароль по умолчанию: `root`/`udsmam0`.
2. Для получения доступа к панели администрирования OpenUDS, следует в меню пользователя выбрать пункт «Панель управления» (рис. 54).

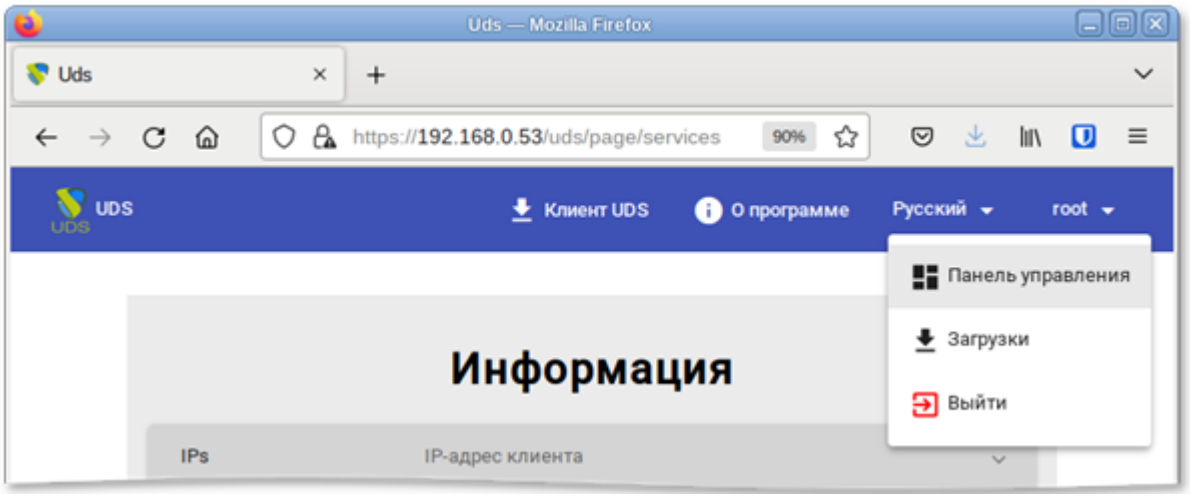

Рис. 54

7.1.3. Установка OpenUDS Tunnel

Установка OpenUDS Tunnel должна выполняться на отдельной от OpenUDS Server системе:

```
# apt-get install openuds-tunnel
```

Примечание. При установке openuds-tunnel в `/etc/openuds-tunnel/ssl` генерируются сертификаты. Их можно заменить на свои, выпущенные внутри организации или Удостоверяющим Центром.

7.1.3.1. Настройка OpenUDS Tunnel

На системе с OpenUDS Tunnel:

- указать адрес сервера OpenUDS (брокера) в файле `/etc/openuds-tunnel/udstunnel.conf`:

```
uds_server = http://192.168.0.53/uds/rest/tunnel/ticket
uds_token = 5ba9d52bb381196c2a22e495ff1c9ba4bdc03440b726aa8b
```

где 192.168.0.53 – адрес OpenUDS сервера (брокера);

- запустить и добавить в автозагрузку сервис OpenUDS Tunnel:

```
# systemctl enable --now openuds-tunnel.service
```

На сервере OpenUDS зарегистрировать туннельный сервер, выполнив команду:

```
# openuds_tunnel_register.py -H 192.168.0.88 -n Tunnel -t 5ba9d52bb381196c2a22e495ff1c9ba4bdc03440b726aa8b
Tunnel token register success. (With token: 5ba9d52bb381196c2a22e495ff1c9ba4bdc03440b726aa8b)
```

где:

- `-H` – задает IP-адрес туннельного сервера;
- `-n` – задает название туннеля;
- `-t` – позволяет указать токен туннельного сервера (из файла `udstunnel.conf`).

При создании туннельного транспорта, на вкладке «Туннель» указать IP-адрес и порт туннельного-сервера: 192.168.0.88:7777.

7.1.3.2. Настройка HTML5

На OpenUDS Tunnel:

1) в файле `/etc/guacamole/guacamole.properties` привести значение параметра uds-base-url к виду:

```
http://<IP openuds сервера>/uds/guacamole/auth/<Токен из файла udstunnel.conf>/
```

Например:

```
uds-base-
url=http://192.168.0.53/uds/guacamole/auth/5ba9d52bb381196c2a22e495ff1c9ba4bdc03440b726aa8b
```

2) настроить tomcat:

- для подключения по http: так как tomcat по умолчанию работает на порту 8080, то перед запуском tomcat необходимо, либо остановить службу ahttpd:

```
# systemctl disable --now ahttpd
```

либо изменить в файле `/etc/tomcat/server.xml` порт 8080 на другой допустимый номер порта, например, 8081:

```
<Connector port="8081" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

а) для подключения по https: в файл `/etc/tomcat/server.xml` добавить новый Connector, в котором указать порт (в примере 10443), сертификат (файл `.crt`, `.pem` и т. д.), закрытый ключ (`.key`, `.pem` и т. д.):

```
    <Connector                                                    port="10443"
protocol="org.apache.coyote.http11.Http11AprProtocol" SSLEnabled="true"
               ciphers="A-CHACHA20-POLY1305,ECDHE-RSA-CHACHA20-POLY1305,
    ECDHE-ECDSA-AES128-GCM-SHA256,ECDHE-RSA-AES128-GCM-SHA256,
    ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AES256-GCM-SHA384,
    DHE-RSA-AES128-GCM-SHA256,DHE-RSA-AES256-GCM-SHA384,
    ECDHE-ECDSA-AES128-SHA256,ECDHE-RSA-AES128-SHA256,
    ECDHE-ECDSA-AES128-SHA,ECDHE-RSA-AES256-SHA384,
    ECDHE-RSA-AES128-SHA,ECDHE-ECDSA-AES256-SHA384,
    ECDHE-ECDSA-AES256-SHA,ECDHE-RSA-AES256-SHA,
    DHE-RSA-AES128-SHA256,DHE-RSA-AES128-SHA,
    DHE-RSA-AES256-SHA256,DHE-RSA-AES256-SHA,
    ECDHE-ECDSA-DES-CBC3-SHA,ECDHE-RSA-DES-CBC3-SHA,
    EDH-RSA-DES-CBC3-SHA,AES128-GCM-SHA256,AES256-GCM-SHA384,
    AES128-SHA256,AES256-SHA256,AES128-SHA,AES256-SHA,DES-CBC3-SHA"
               maxThreads="500" scheme="https" secure="true"
               SSLCertificateFile="/etc/openuds-tunnel/ssl/certs/openuds-
tunnel.pem"
```

```
                SSLCertificateKeyFile="/etc/openuds-
tunnel/ssl/private/openuds-tunnel.key"
                maxKeepAliveRequests="1000"
                clientAuth="false" sslProtocol="TLSv1+TLSv1.1+TLSv1.2" />
```

3) запустить сервисы guacd и tomcat:

```
# systemctl enable --now guacd tomcat
```

На сервере OpenUDS при создании нового туннельного транспорта HTML5RDP на вкладке «Туннель» указать IP-адрес и порт туннельного-сервера:

- `http://192.168.0.88:8080` – для подключения по http;
- `https://192.168.0.88:10443` – для подключения по https.

7.2. Обновление OpenUDS

После обновления openuds-server до новой версии необходимо выполнить следующие действия:

1) перенести изменения, если они есть, из нового конфигурационного файла `/etc/openuds/settings.py.rpmnew` в файл `/etc/openuds/settings.py`.

Проверить, что изменилось можно, выполнив команду:

```
# diff -u --color /etc/openuds/settings.py /etc/openuds/settings.py.rpmnew
```

2) выполнить миграцию базы данных:

```
# su -s /bin/bash - openuds -c "cd /usr/share/openuds; python3 manage.py migrate"
```

3) перезагрузить систему, так как при обновлении не создается файл `/run/openuds/socket`.

7.3. Настройка OpenUDS

7.3.1. Поставщики услуг

В разделе «Поставщики услуг» можно подключить один из поставщиков («Service providers») (рис. 55):

- поставщик платформы Proxmox;
- поставщик платформы OpenNebula;
- отдельный сервер без виртуализации: Поставщик машин статических IP.

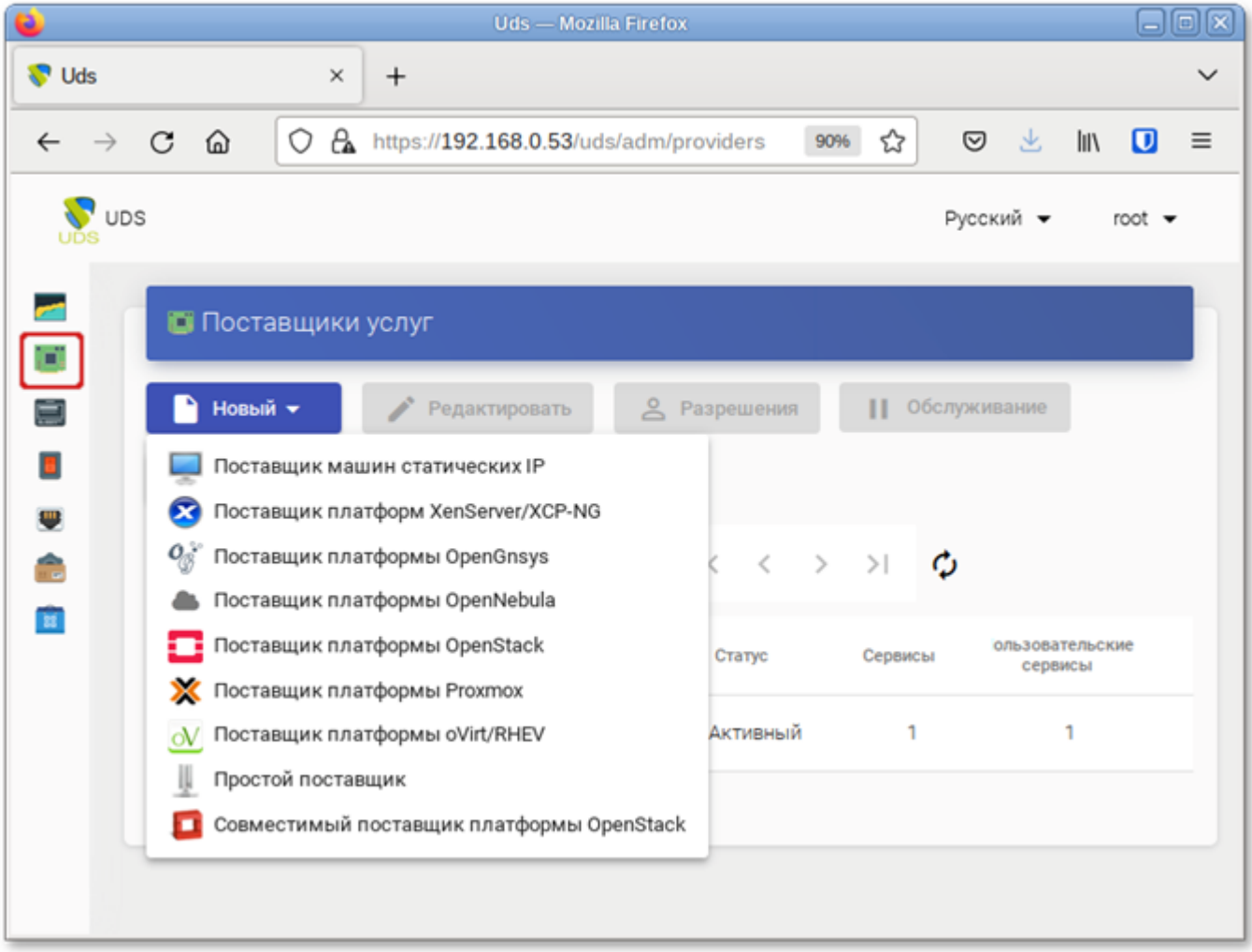

Рис. 55

7.3.1.1. OpenNebula

Минимальные параметры для настройки «Поставщик платформы OpenNebula»:

- вкладка «Основной»: название, IP-адрес сервера OpenNebula (поле «Хост»), порт подключения, имя пользователя (с правами администратора) и пароль (рис. 56);

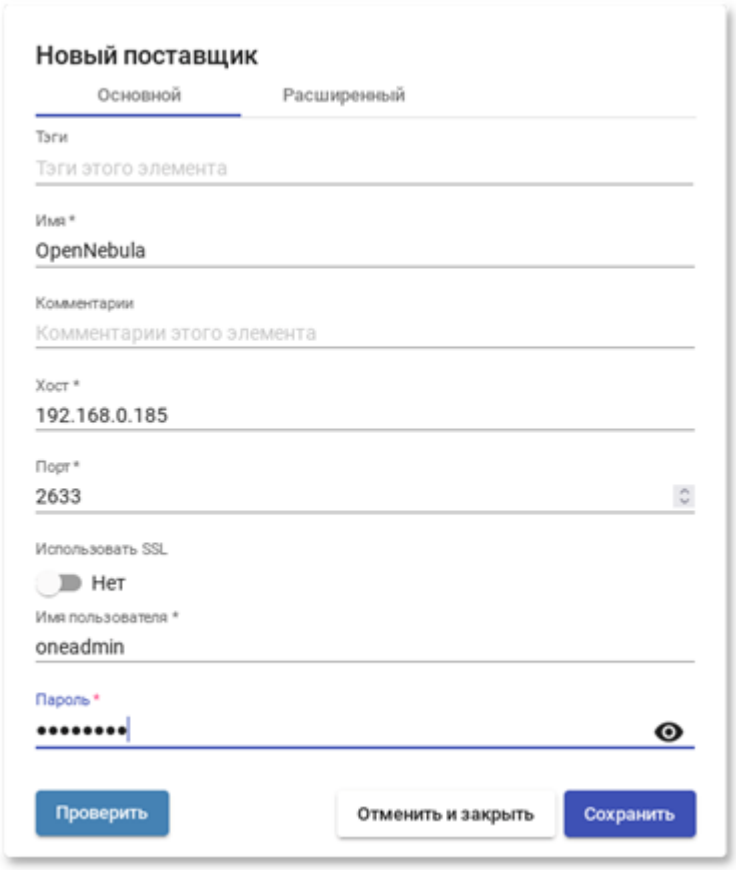

Рис. 56

- вкладка «Расширенный»: максимальное количество одновременно создаваемых ВМ, максимальное количество одновременно удаляемых ВМ, таймаут подключения к OpenNebula в секундах (рис. 57).

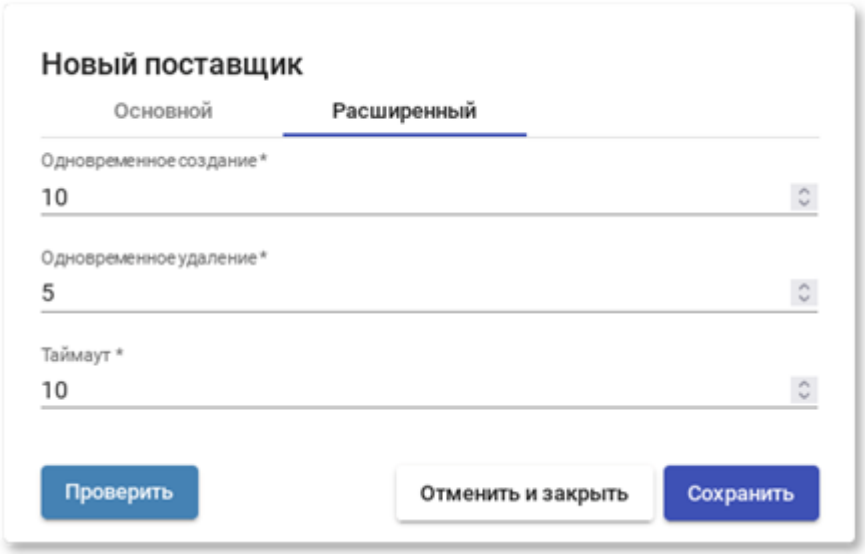

Рис. 57

Используя кнопку «Проверить», можно убедиться, что соединение установлено правильно.

После интеграции платформы OpenNebula в OpenUDS необходимо создать базовую службу типа «Действующие образы OpenNebula». Для этого дважды щелкнуть мышью по строке созданного поставщика услуг или в контекстном меню поставщика выбрать пункт «Подробность» (рис. 58).

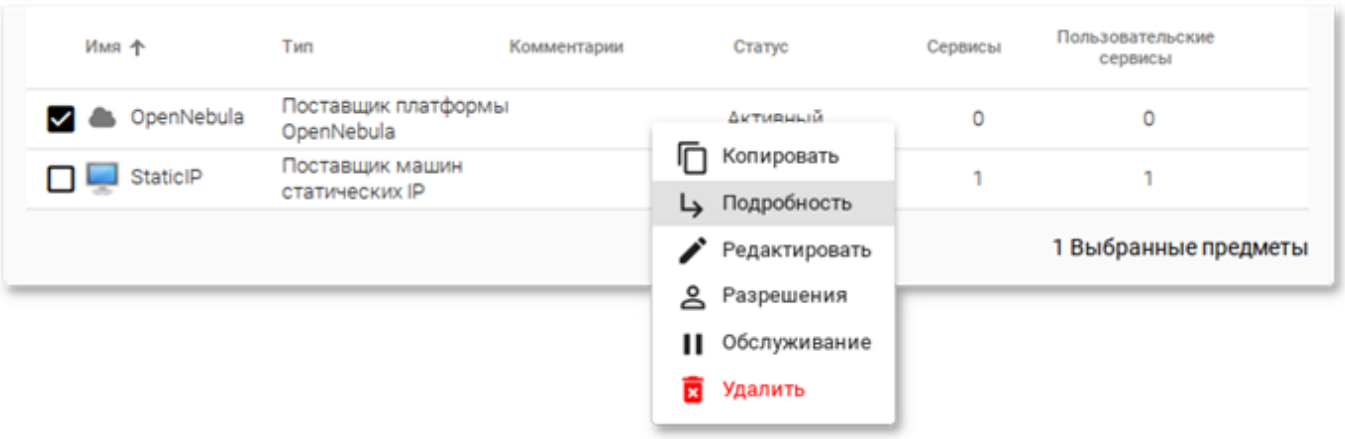

Рис. 58

П р и м е ч а н и е . Выбрав пункт «Обслуживание», можно приостановить все операции, выполняемые сервером OpenUDS для данного поставщика услуг. Поставщик услуг рекомендуется переводить в режим обслуживания в случаях, когда связь с этим поставщиком была потеряна или запланирован перерыв в обслуживании.

В открывшемся окне, на вкладке «Поставщики услуг» нажать на кнопку «Новый» → «Действующие образы OpenNebula» (рис. 59).

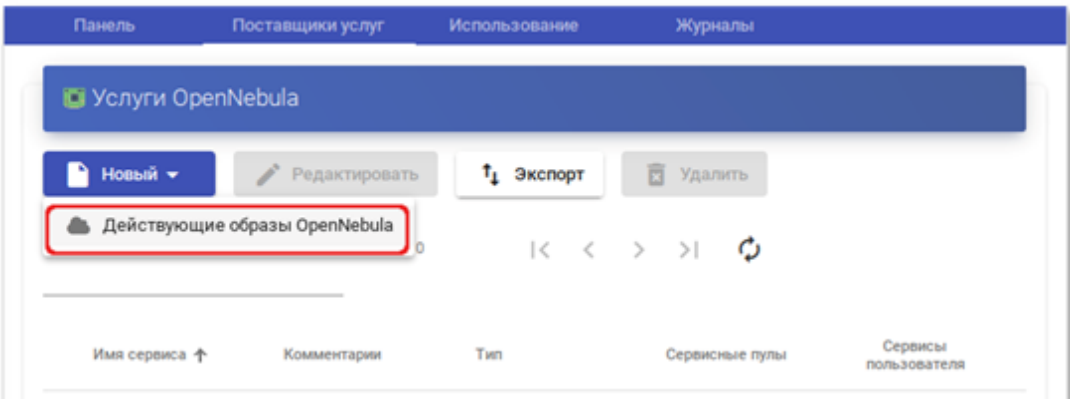

Рис. 59

Заполнить минимальные параметры конфигурации:

1) вкладка «Основной» (рис. 60):

- «Имя» – название службы;
- «Хранилище» – место, где будут храниться сгенерированные виртуальные рабочие столы;

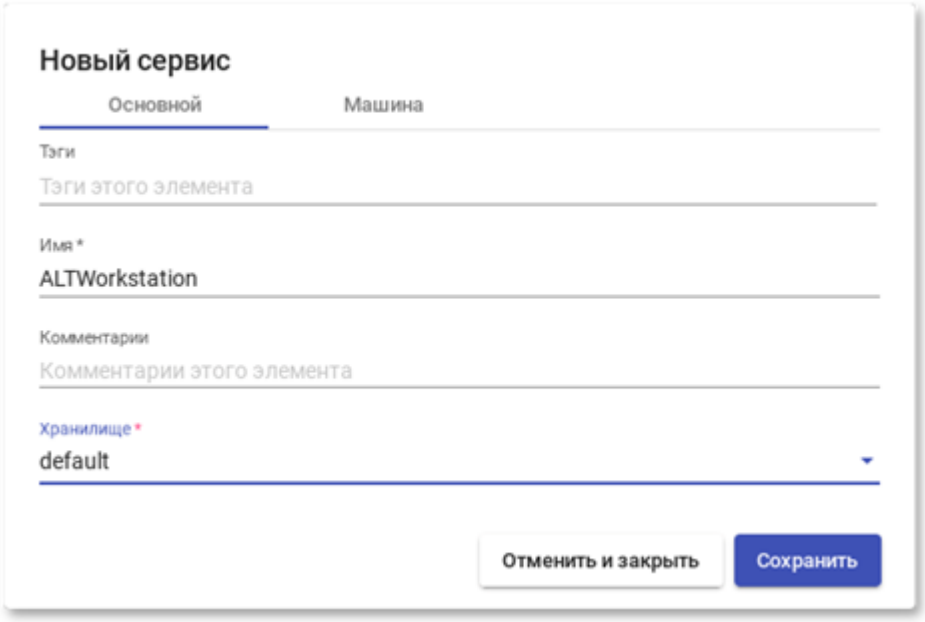

Рис. 60

2) вкладка «Машина» (рис. 61):

- «Базовый шаблон» – шаблон ВМ, используемый системой OpenUDS для развертывания виртуальных рабочих столов (см. п. 7.3.10);
- «Имена машин» – базовое название для клонов с этой машины (например, Desk-work-);
- «Длина имени» – количество цифр счетчика, прикрепленного к базовому имени рабочих столов (например, если Длина имени = 3, названия сгенерированных рабочих столов будут: Desk-work-000, Desk-work-001 ... Desk-work-999).

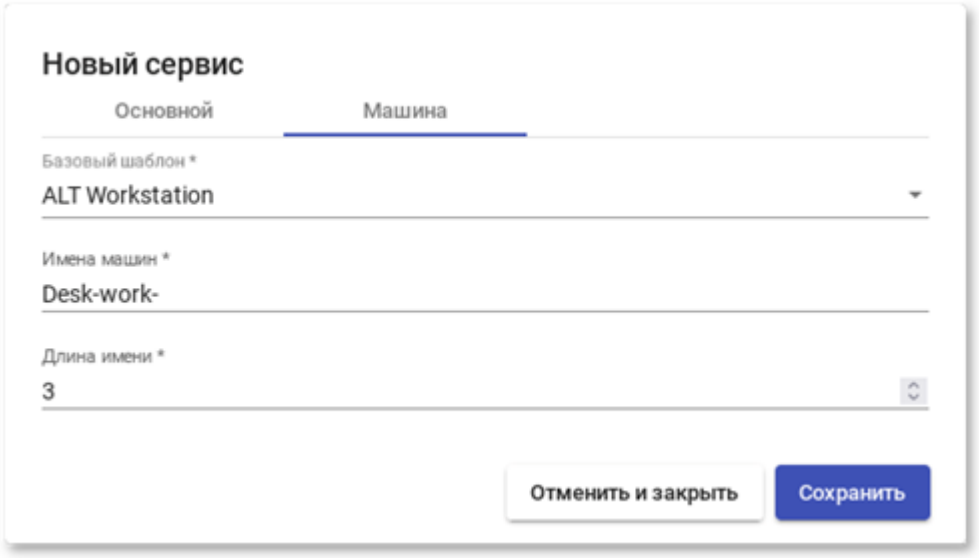

Рис. 61

После того, как среда OpenUDS будет настроена и будет создан первый «пул служб», в среде OpenNebula можно будет наблюдать, как разворачиваются рабочие столы. Сначала будет создан образ ВМ («UDSP-pool_name-DSK») – клон образа, шаблон («UDSP-pool_name-publishing-number») – клон ВМ, выбранной при регистрации службы. После завершения процесса создания клона будут созданы рабочие столы («Machine_Name-Name_Length»).

7.3.1.2. PVE

Минимальные параметры для настройки Поставщик платформы Proxmox:

- вкладка «Основной»: название, IP-адрес/имя сервера или кластера PVE (поле «Хост»), порт подключения, имя пользователя с достаточными привилегиями в PVE (в формате пользователь@аутентификатор) и пароль (рис. 62);

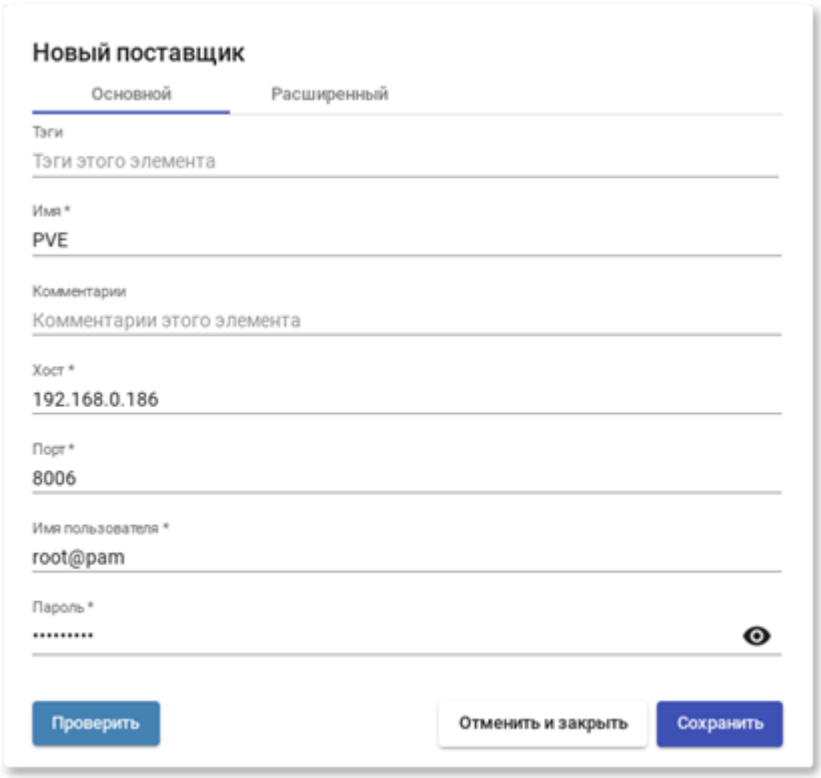

Рис. 62

- вкладка «Расширенный»: максимальное количество одновременно создаваемых ВМ, максимальное количество одновременно удаляемых ВМ, таймаут подключения к Proxmox в секундах, идентификатор ВМ, с которым OpenUDS начнет генерировать ВМ на Proxmox (>=10000) (рис. 63).

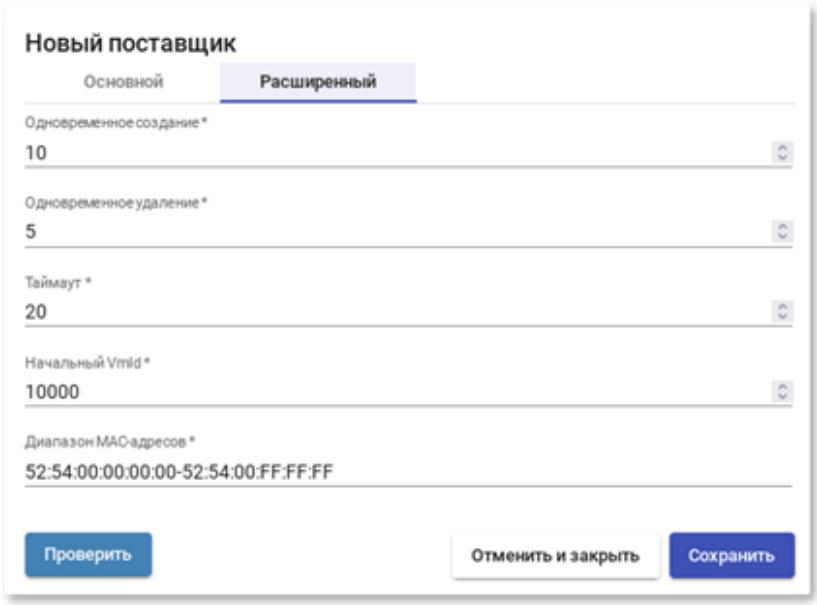

Рис. 63

Используя кнопку «Проверить», можно убедиться, что соединение установлено правильно.

После интеграции платформы PVE в OpenUDS необходимо создать базовую службу типа «Связанный клон Proxmox». Для этого дважды щелкнуть мышью по строке созданного поставщика услуг или в контекстном меню поставщика выбрать пункт «Подробность» (рис. 64).

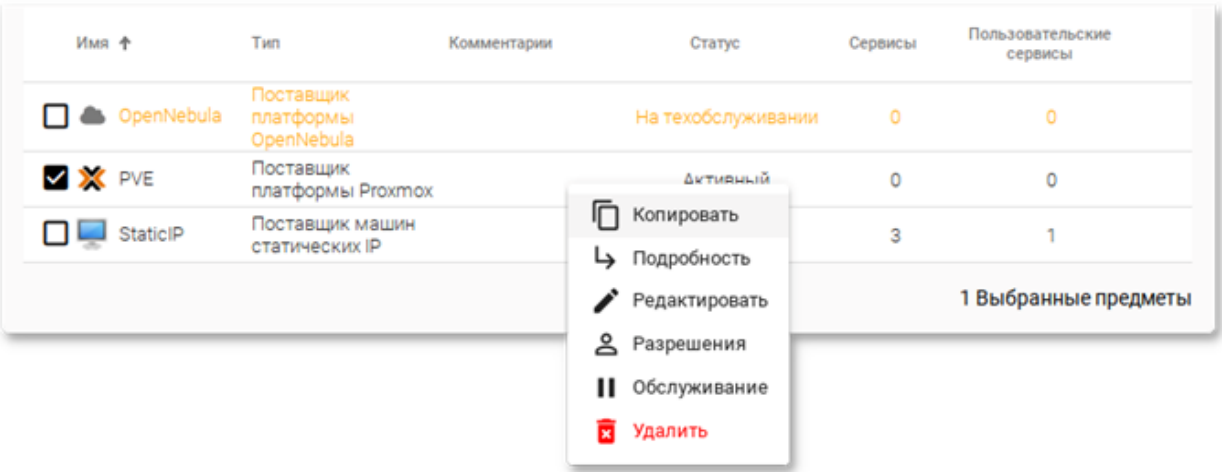

Рис. 64

Примечание. Выбрав пункт «Обслуживание», можно приостановить все операции, выполняемые сервером OpenUDS для данного поставщика услуг. Для типа поставщик услуг рекомендуется переводить в режим обслуживания в случаях, когда связь с этим поставщиком была потеряна или запланирован перерыв в обслуживании.

В открывшемся окне, на вкладке «Поставщики услуг» нажать на кнопку «Новый» → «Связанный клон Proxmox» (рис. 65).

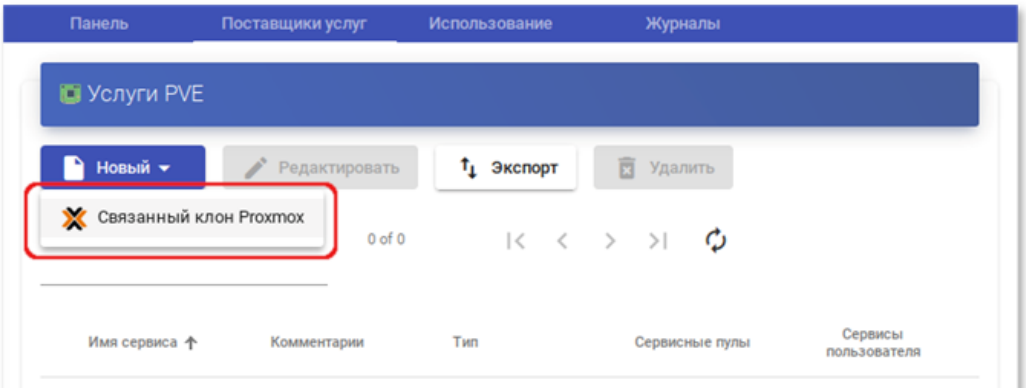

Рис. 65

Заполнить минимальные параметры конфигурации (рис. 66, рис. 67):

1) вкладка «Основной»:
   - «Имя» – название службы;
   - «Пул» – пул, в котором будут находиться ВМ, созданные OpenUDS;
   - «Высокая доступность» – включать созданные ВМ в группу HA PVE;
   - «Сначала попробовать SOFT Shutdown» – если активно, OpenUDS попытается, перед уничтожением автоматически сгенерированного виртуального рабочего стола, выполнить контролируемое отключение машины;
2) вкладка «Машина»:
   - «Базовая машина» – шаблон ВМ, используемый системой OpenUDS для развертывания виртуальных рабочих столов (см. п. 7.3.10);
   - «Хранилище» – место, где будут храниться сгенерированные виртуальные рабочие столы (поддерживаются хранилища, позволяющие создавать «Снимки»);
   - «Имена машин» – базовое название для клонов с этой машины (например, Desk-SL-);
   - «Длина имени» – количество цифр счетчика, прикрепленного к базовому имени рабочих столов (например, если «Длина имени» = 3, названия сгенерированных рабочих столов будут: Desk-SL-000, Desk-SL-001 ... Desk-SL-999).

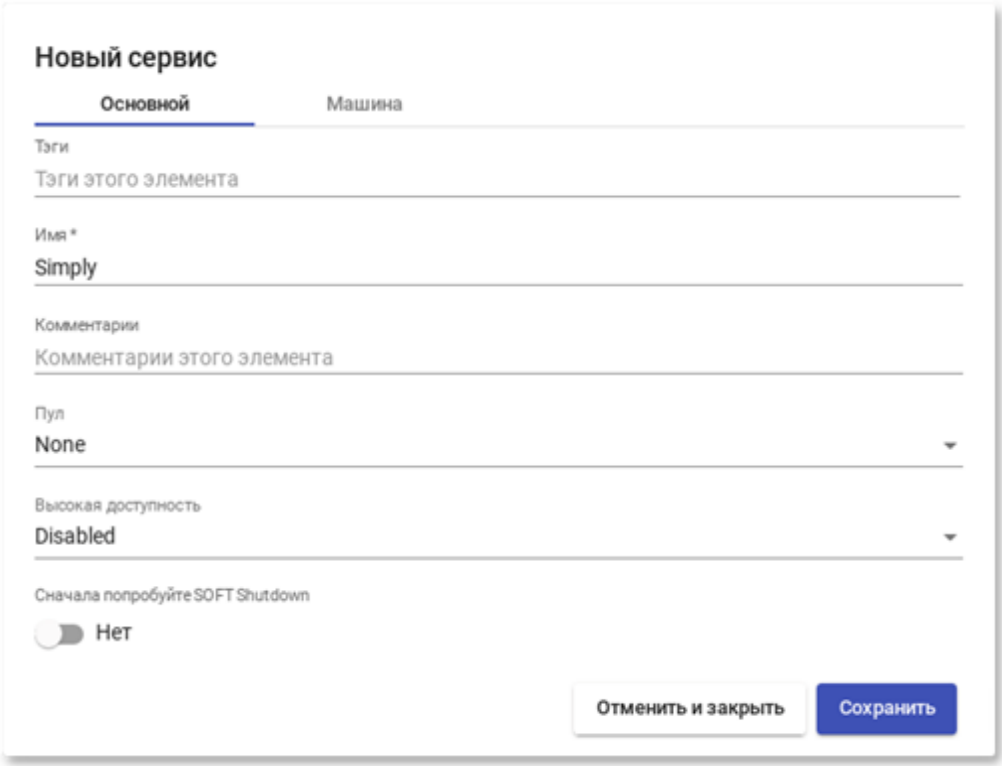
Новый сервис
Основной
Машина
Тэги
Тэги этого элемента
Имя *
Simply
Комментарии
Комментарии этого элемента
Пул
None
Высокая доступность
Disabled
Сначала попробуйте SOFT Shutdown
Нет
Отменить и закрыть
Сохранить

Рис. 66

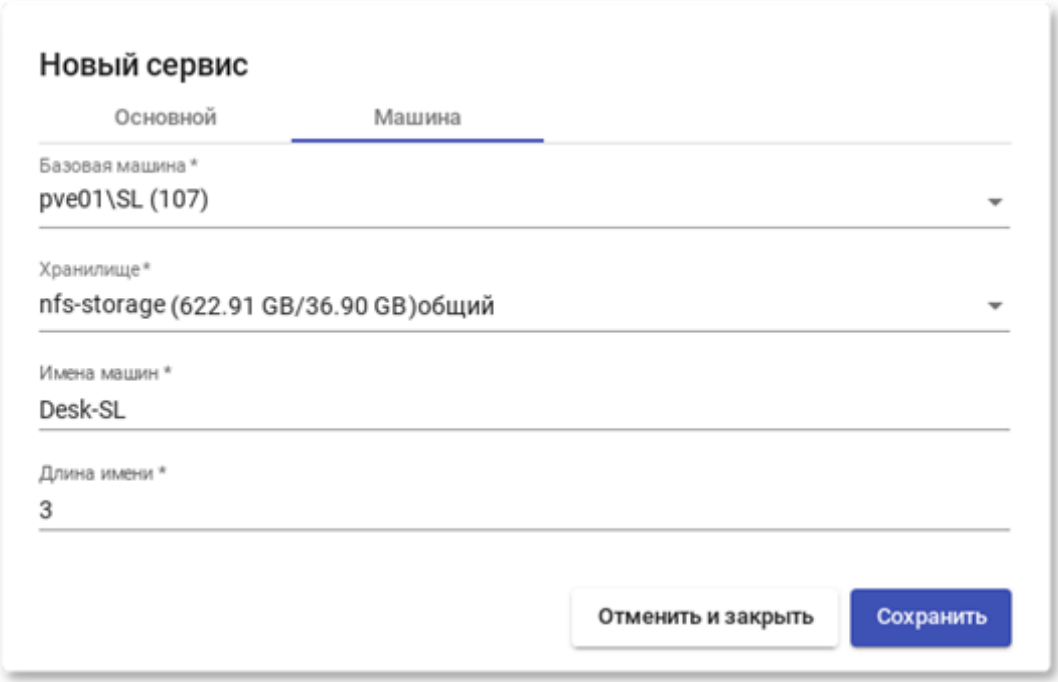
Новый сервис
Основной
Машина
Базовая машина *
pve01\SL (107)
Хранилище *
nfs-storage (622.91 GB/36.90 GB)общий
Имена машин *
Desk-SL
Длина имени *
3
Отменить и закрыть
Сохранить

Рис. 67

После того, как среда OpenUDS будет настроена и будет создан первый «пул услуг», в среде PVE можно будет наблюдать, как разворачиваются рабочие столы. Сначала будет создан шаблон («UDS-Publication-pool_name-publishing-number») – клон ВМ, выбранной при регистрации службы. После завершения процесса создания клона будут созданы рабочие столы («Machine_Name-Name_Length») (рис. 68).

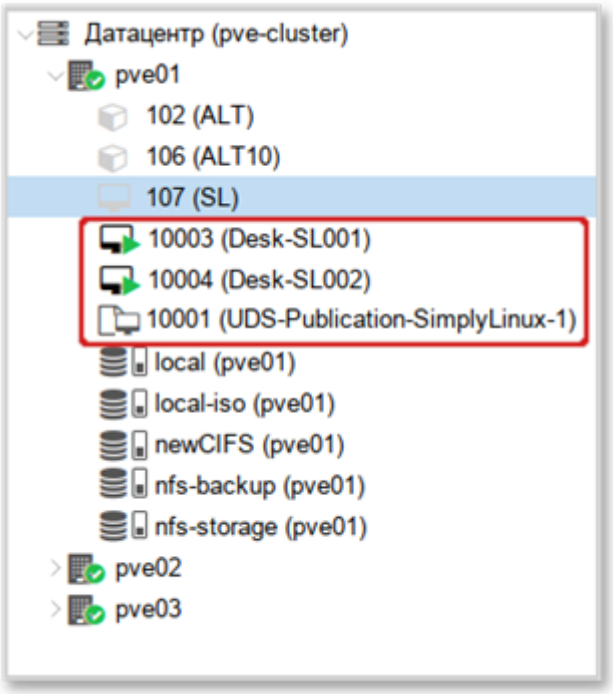

Рис. 68

7.3.1.3. Удаленный доступ к отдельному серверу

В OpenUDS есть возможность предоставить доступ к постоянным устройствам (физическим или виртуальным). Доступ к отдельному серверу осуществляется путем назначения IP-адресов пользователям.

Для регистрации поставщика данного типа следует в разделе «Поставщики услуг» (п. 7.3.1) нажать на кнопку «Новый» и выбрать пункт «Поставщик машин статических IP».

Для настройки «Поставщика машин статических IP» достаточно задать название поставщика (рис. 69).

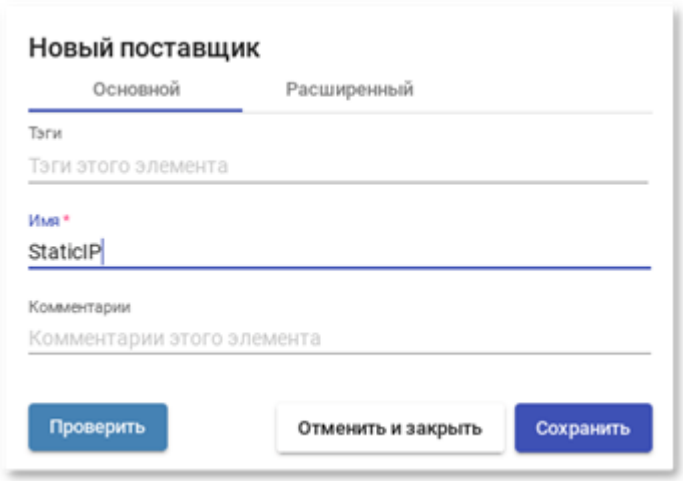

Рис. 69

Для создания базовых услуг «Поставщика машин статических IP» следует дважды щелкнуть мышью по строке созданного поставщика или в контекстном меню поставщика выбрать пункт «Подробность» (рис. 70).

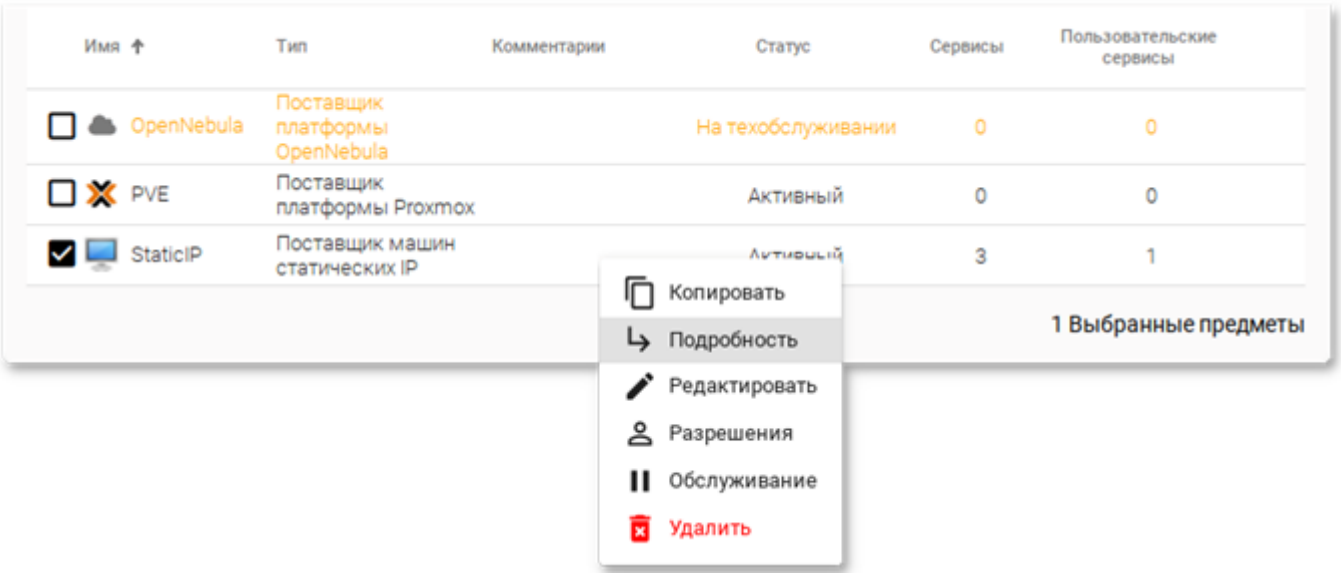

Рис. 70

В открывшемся окне, на вкладке «Поставщики услуг» нажать на кнопку «Новый» → «Статический множественный IP-адрес» или «Новый» → «Статический одиночный IP-адрес» (рис. 71).

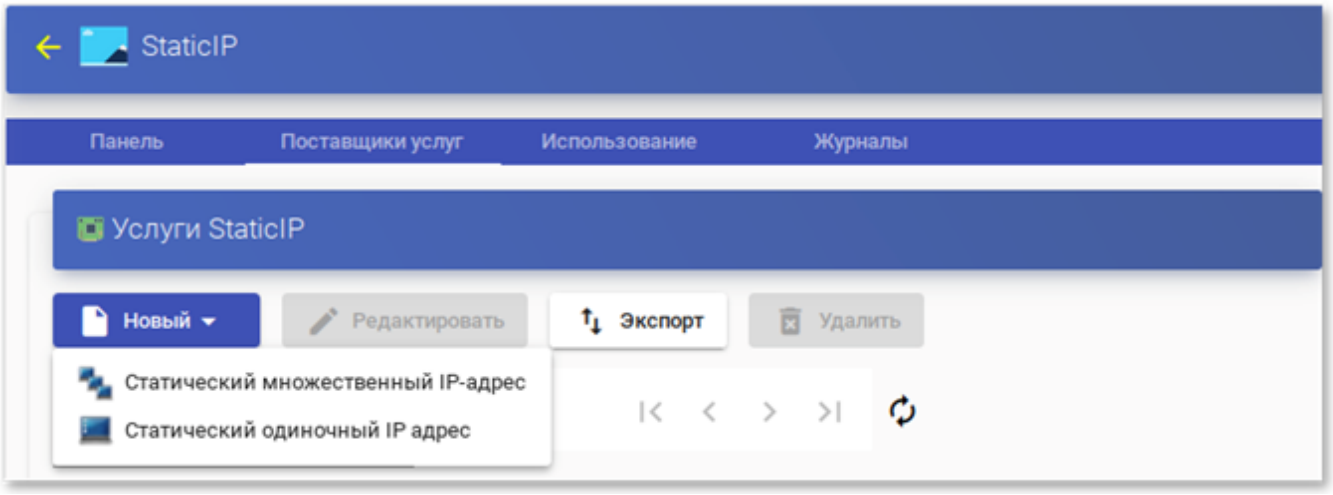

Рис. 71

Статический множественный IP-адрес используется для подключения одного пользователя к одному компьютеру. Поддерживается неограниченное количество IP-адресов (можно включить в список все устройства, которые должны быть доступны удаленно). По умолчанию система будет предоставлять доступ к устройствам в порядке очереди (первый пользователь, получивший доступ к этому пулу, получает доступ к машине с первым IP-адресом из списка). Также можно настроить выборочное распределение, чтобы определенному пользователю назначался определенный компьютер (IP-адрес).

Примечание. Для настройки привязки конкретного пользователя к конкретному IP необходимо в п. 7.3.4.7 для созданной услуги на вкладке «Назначенные сервисы» нажать на кнопку «Назначить услугу» и задать привязку пользователя устройству (рис. 72).

Статический одиночный IP-адрес используется для подключения нескольких пользователей к одному компьютеру. При обращении каждого нового пользователя будет запускаться новый сеанс.

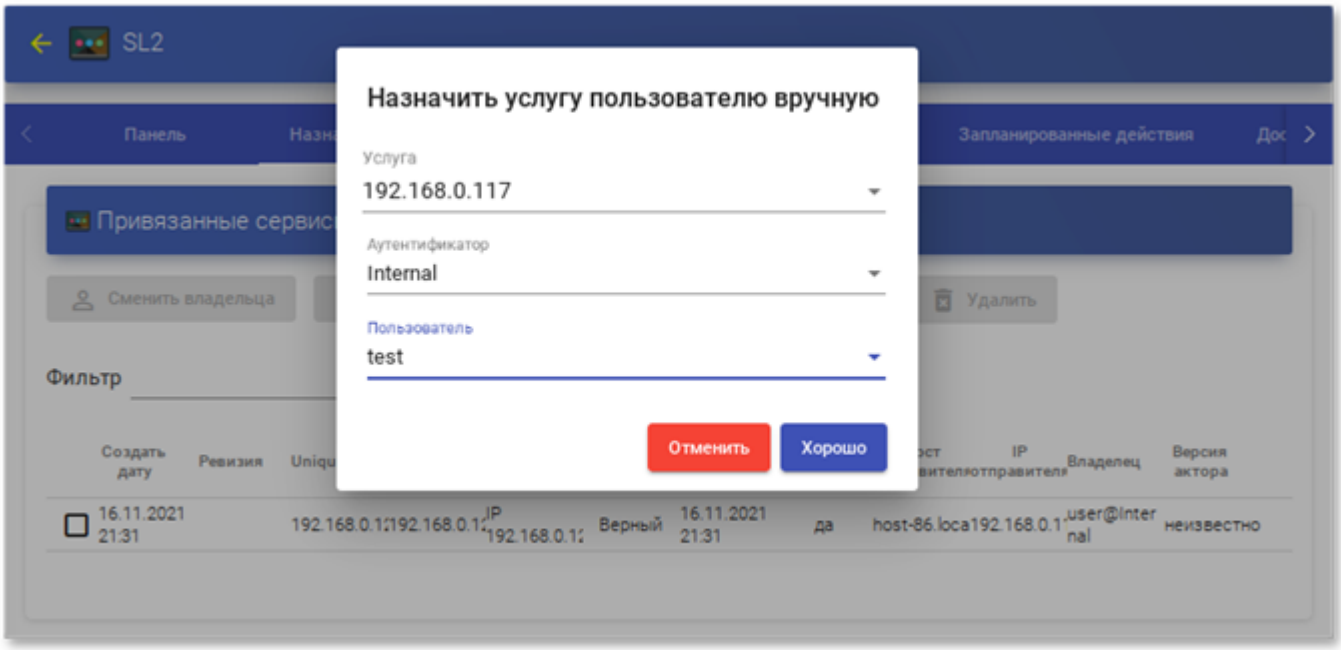

Рис. 72

Параметры конфигурации для услуги «Статический множественный IP-адрес» (рис. 73, рис. 74):

1) вкладка «Основной»:

- «Имя» – название службы;
- «Список серверов» – один или несколько IP-адресов машин, к которым будет осуществляться доступ (машины должны быть включены и настроены см. п. 7.3.10);
- «Ключ услуги» – токен, который будет использоваться клиентами для связи с сервисом. Если в этом поле не указан токен (пусто), система не будет контролировать сеансы пользователей на компьютерах. Таким образом, когда компьютер назначается пользователю, это назначение будет сохраняться до тех пор, пока администратор не удалит его вручную. При наличии токена сеансы пользователей будут контролироваться (при выходе из сеанса, компьютеры снова становятся доступными для доступа других пользователей). Если токен указан, необходимо, чтобы на компьютерах (IP-адрес, которых указан в поле «Список серверов») был установлен Unmanaged UDS Actor;

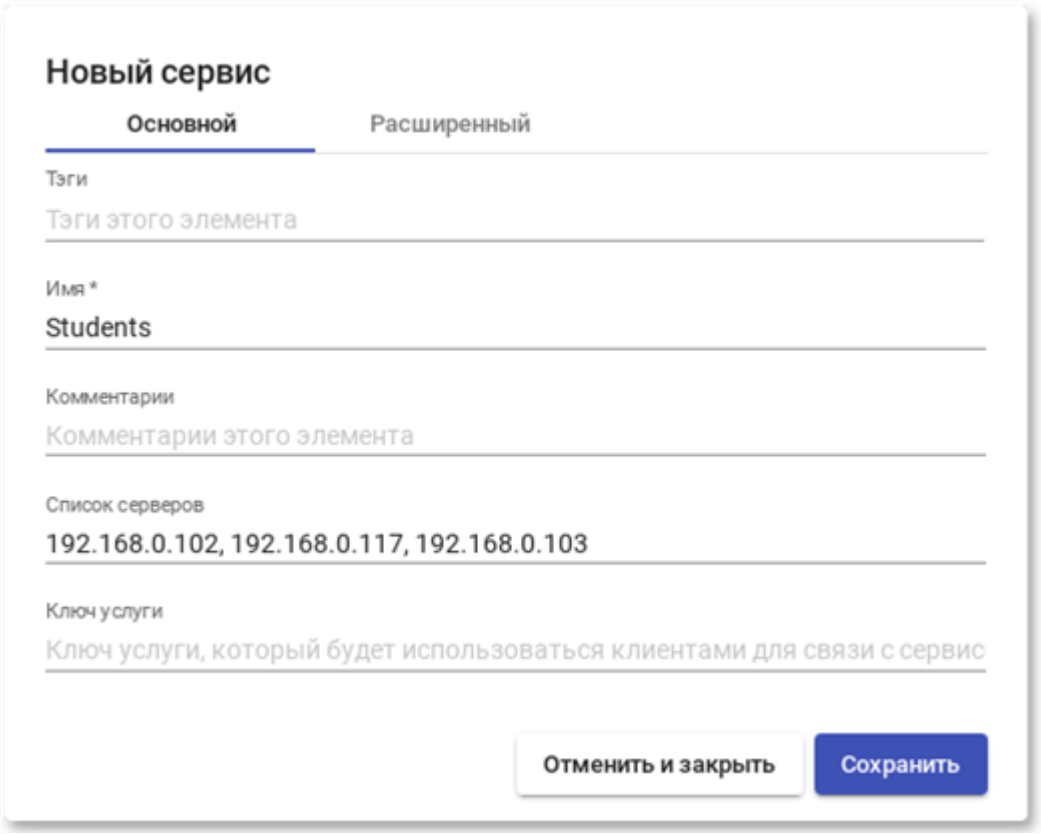

Рис. 73

2) вкладка «Расширенный»:

- «Проверьте порт» – порт, по которому система может проверить, доступен ли компьютер. Если компьютер не доступен, система автоматически предоставит следующее устройство в списке. 0 – не проверять доступность компьютера;
- «Пропустить время» – период, в течение которого не будет проверяться доступность недоступной машины;
- «Максимальное количество сеансов на машину» – максимальная продолжительность сеанса (в часах), прежде чем OpenUDS решит, что эта машина заблокирована и освободит ее (0 означает «никогда»).

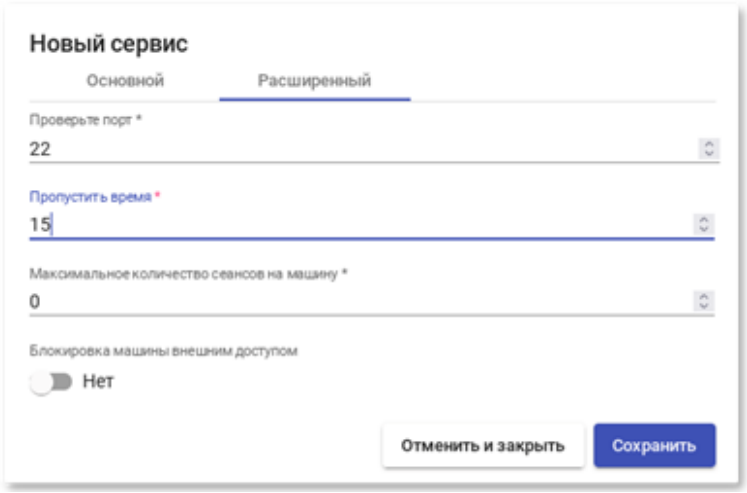

Рис. 74

Примечание. Назначение IP-адресов будет осуществляться в порядке доступа, то есть первому пользователю, который обращается к службе, будет назначен первый IP-адрес в списке. IP-адрес будет привязан пользователю, даже после выхода пользователя из системы (пока администратор не удалит привязку вручную).

Просмотреть/изменить привязанные сеансы можно в п. 7.3.4.7 на вкладке «Назначенные сервисы» (рис. 75).

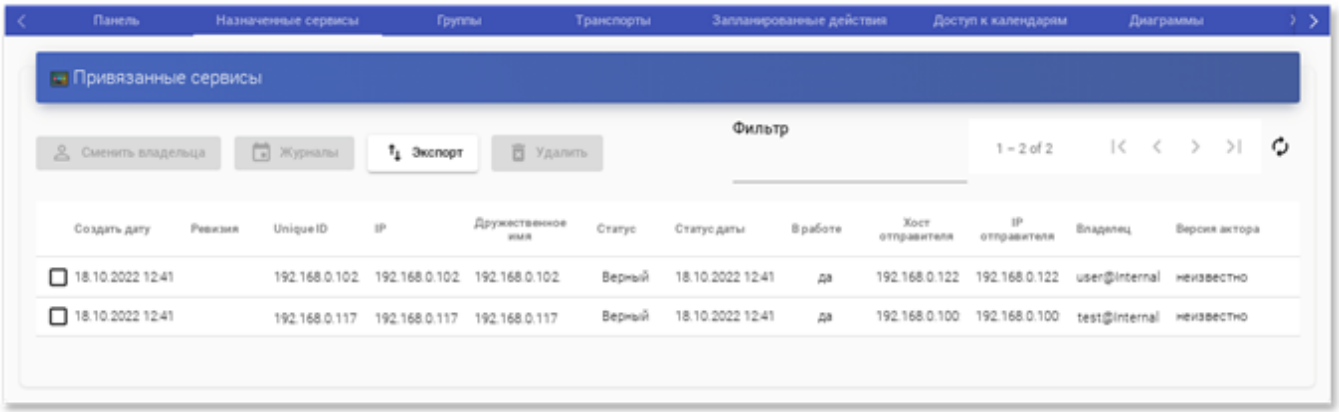

Рис. 75

Параметры конфигурации для услуги «Статический одиночный IP-адрес» (рис. 76):

- «Имя» – название службы;
- «IP-адрес машины» – IP-адрес машины, к которой будет осуществляться доступ (машина должна быть включена и настроена см. п. 7.3.10).

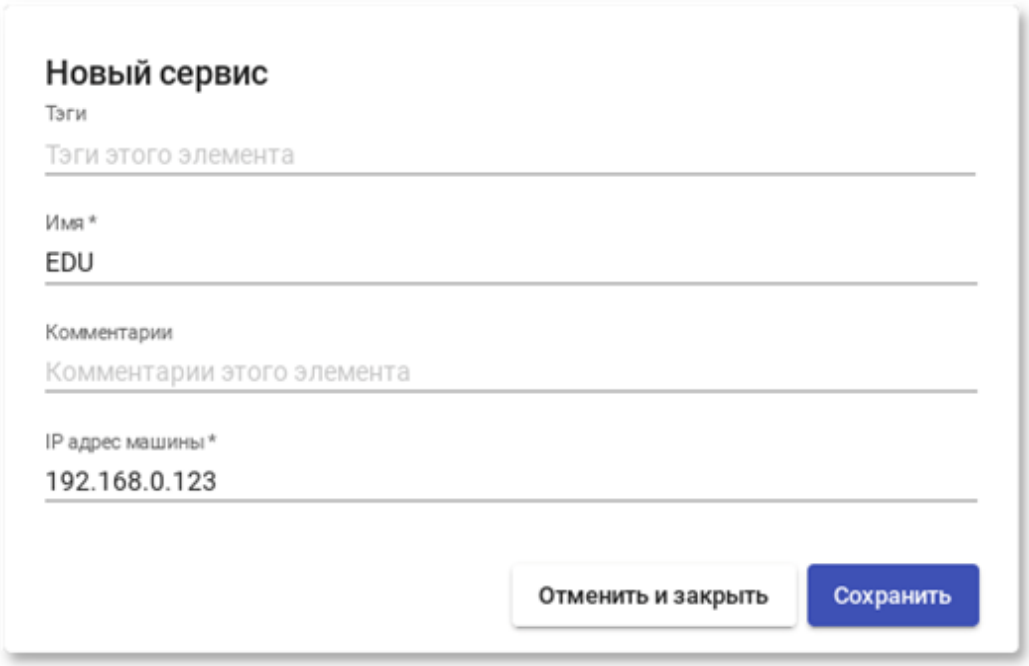

Рис. 76

7.3.2. Настройка аутентификации пользователей

Аутентификатор проверяет подлинность пользователей и предоставляет пользователям и группам пользователей разрешения на подключение к различным виртуальным рабочим столам.

Аутентификатор не является обязательным компонентом для создания «пула услуг», но, если не создан хотя бы один аутентификатор, не будет пользователей, которые смогут подключаться к службам на платформе OpenUDS.

Примечание. Если в системе зарегистрировано более одного аутентификатора, и они не отключены, на экран входа будет добавлено поле Аутентификатор с раскрывающимся списком. В этом списке можно выбрать аутентификатор, который система будет использовать для проверки пользователя (рис. 77).

При создании любого аутентификатора заполняется поле «Метка». Пользователь может пройти проверку подлинности с помощью указанного аутентификатора, даже если в среде OpenUDS настроено несколько аутентификаторов. Для этого нужно получить доступ к экрану входа OpenUDS в формате: `OpenUDS-server/uds/page/login/метка` (например, `https://192.168.0.53/uds/page/login/AD`).

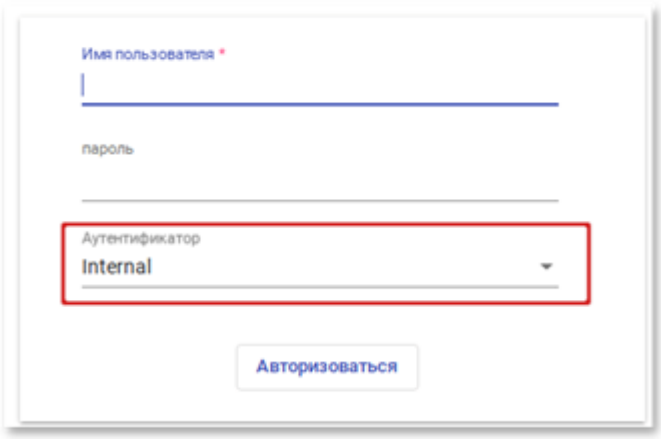

Рис. 77

Для настройки аутентификации в разделе «Аутентификаторы» (Authentificators) (рис. 78) необходимо выбрать тип аутентификации пользователей. Можно выбрать как внешние источники (Active Directory, OpenLDAP и т. д.), так и внутренние (внутренняя база данных, IP-аутентификация).

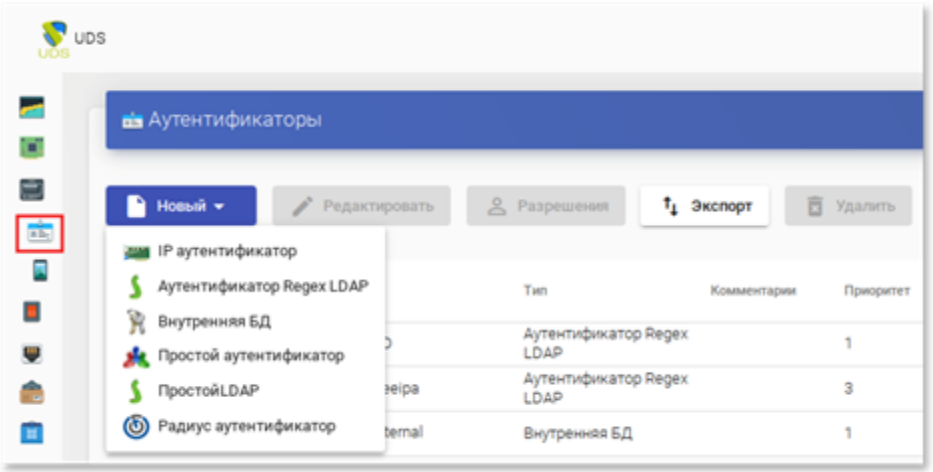

Рис. 78

7.3.2.1. Внутренняя БД

При аутентификации «Внутренняя БД» данные пользователей и групп хранятся в базе данных, к которой подключен сервер OpenUDS.

Для создания аутентификации типа «Внутренняя БД» в разделе «Аутентификаторы» (рис. 78) следует нажать на кнопку: «Новый» → «Внутренняя БД».

Минимальные параметры конфигурации (вкладка «Основной»): имя аутентификатора, приоритет и метка (рис. 79).

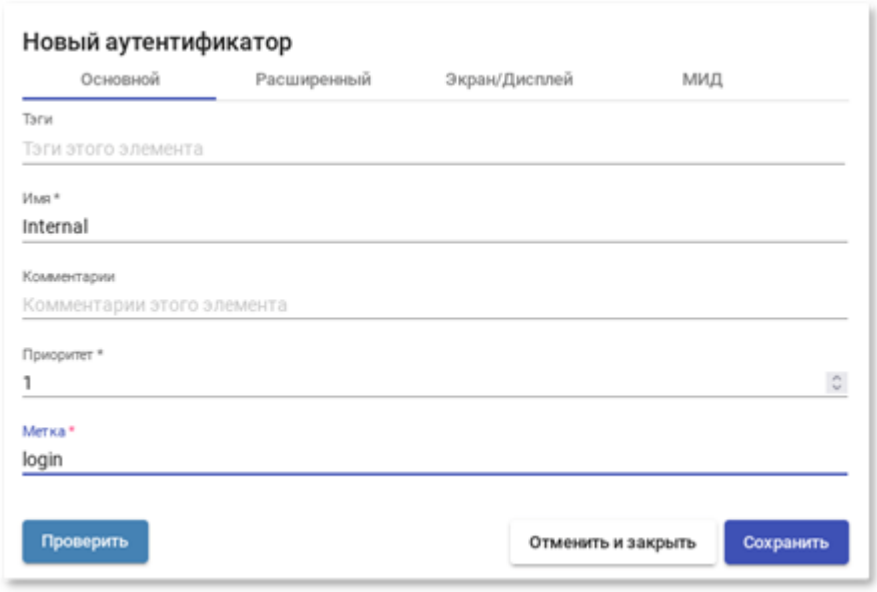

Рис. 79

После того, как аутентификатор типа «Внутренняя БД» создан, нужно зарегистрировать пользователей и группы пользователей. Для этого следует выбрать созданный аутентификатор, затем во вкладке «Группы» создать группы пользователей, во вкладке «Пользователи» создать пользователей (рис. 80).

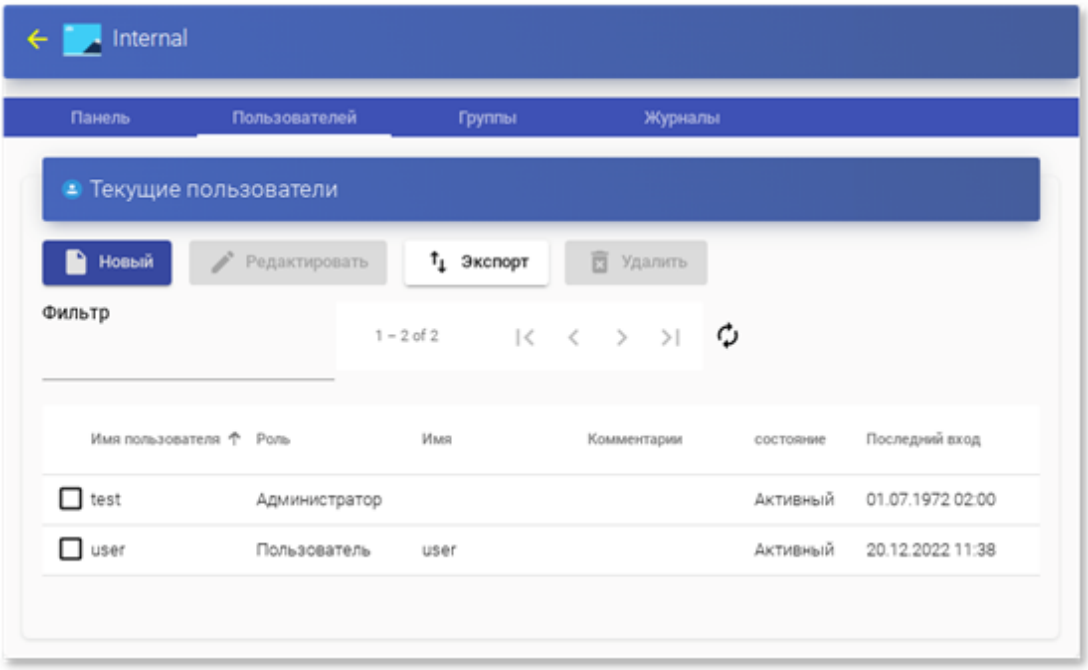

Рис. 80

7.3.2.2. Аутентификатор Regex LDAP

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

ВАЖНО

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

7.3.2.2.1. FreeIPA

Настройка интеграции с FreeIPA (сервер ipa.example.test).

1) в разделе «Аутентификаторы» нажать на кнопку: «Новый» → «Аутентификатор Regex LDAP»;
2) заполнить поля первых трех вкладок:
   - вкладка «Основной»: имя аутентификатора, приоритет, метка, IP-адрес FreeIPA-сервера, порт (обычно 389 без ssl, 636 с ssl) (рис. 81);

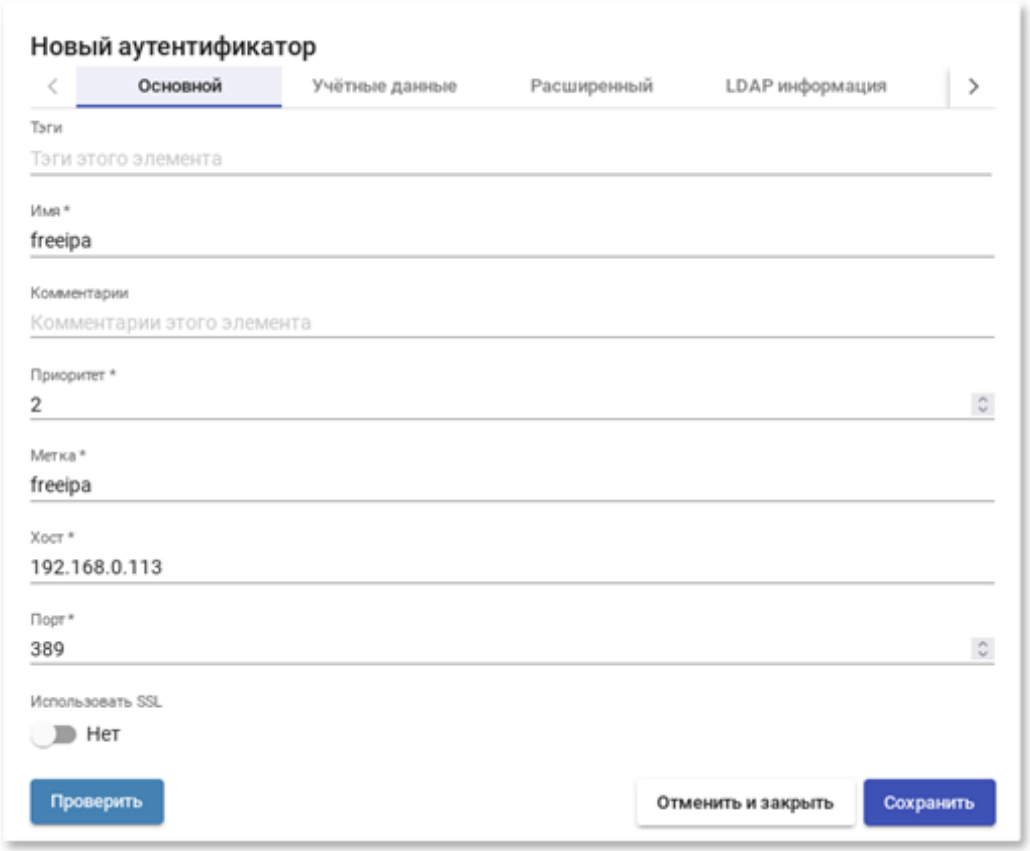

Рис. 81

- вкладка «Учетные данные»: имя пользователя (в формате `uid=user_freeipa,cn=users,cn=accounts,dc=example,dc=test`) и пароль (рис. 82);

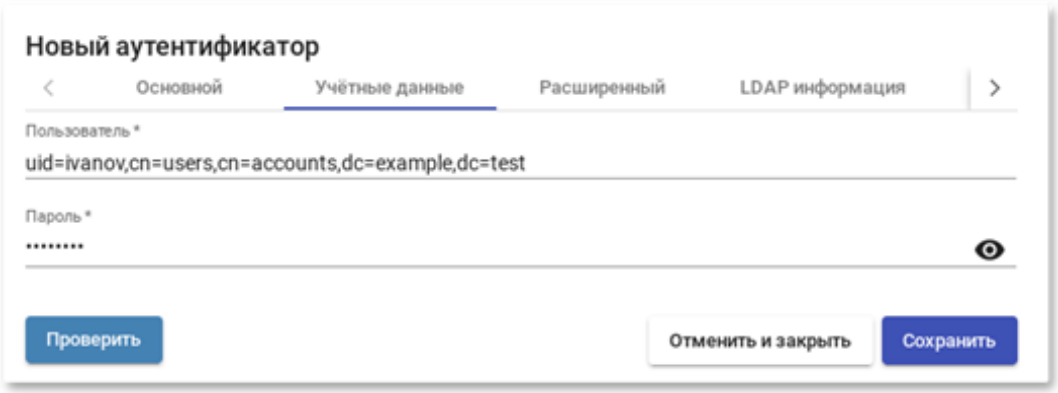

Рис. 82

- вкладка «LDAP информация»: общая база пользователей, класс пользователей LDAP, идентификатор атрибута пользователя, атрибут имени пользователя, атрибут имени группы (рис. 83);

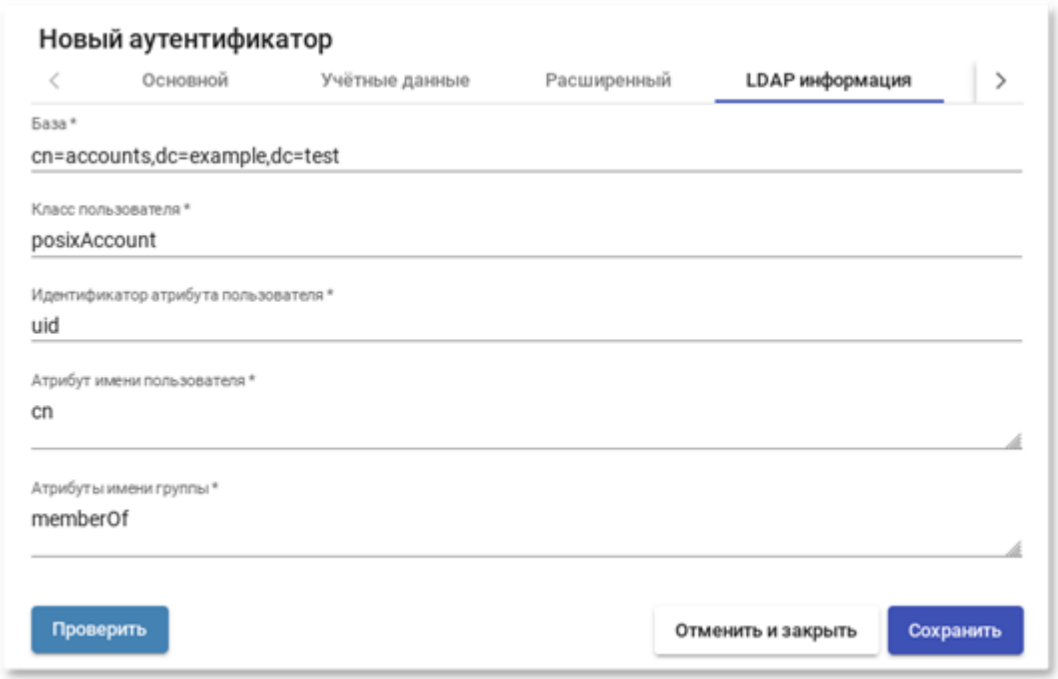

Рис. 83

Примечание. Используя кнопку «Проверить», можно проверить соединение с FreeIPA-сервером.

3) добавить группу LDAP, в которую входят пользователи. Для этого следует выбрать созданный аутентификатор, затем в открывшемся окне на вкладке «Группы» нажать «Новый» → «Группа».

Заполнить dn существующей группы (для FreeIPA по умолчанию это группа `cn=ipausers,cn=groups,cn=accounts,dc=ipa,dc=example,dc=test`), можно также указать разрешенные пулы (рис. 84);

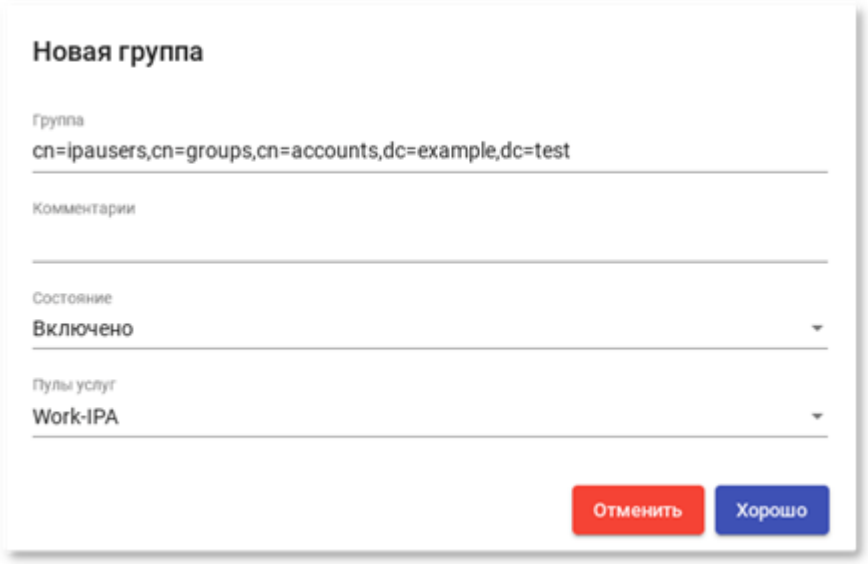

Рис. 84

7.3.2.2.2. Active Directory

Настройка аутентификации в Active Directory (домен test.alt):

1) в разделе «Аутентификаторы» (см. рис. 78) нажать на кнопку: «Новый» → «Аутентификатор Regex LDAP»;
2) заполнить поля первых трех вкладок:
   - вкладка «Основной»: имя аутентификатора, приоритет, метка, IP-адрес сервера AD, порт (обычно 389 без ssl, 636 с ssl) (рис. 85);

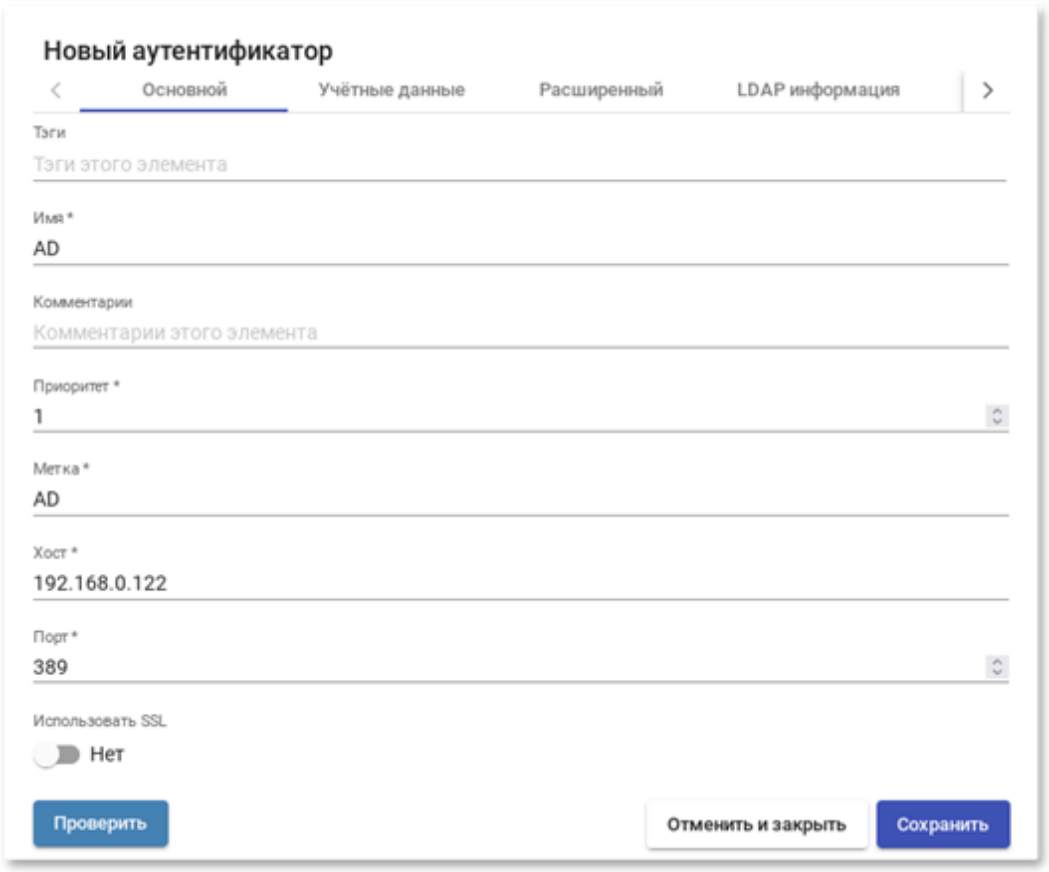

Рис. 85

- вкладка «Учетные данные»: имя пользователя (можно указать в виде имя@домен) и пароль (рис. 86);

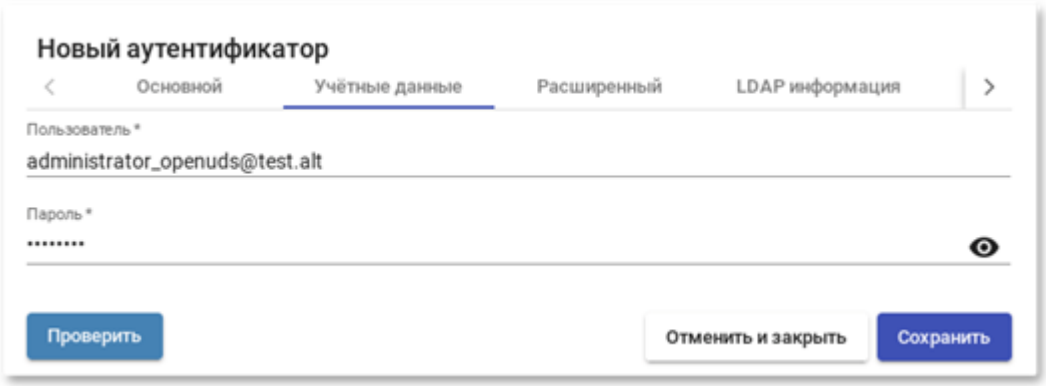

Рис. 86

- вкладка «LDAP информация»: общая база пользователей, класс пользователей LDAP, идентификатор атрибута пользователя, атрибут имени пользователя, атрибут имени группы (рис. 87);

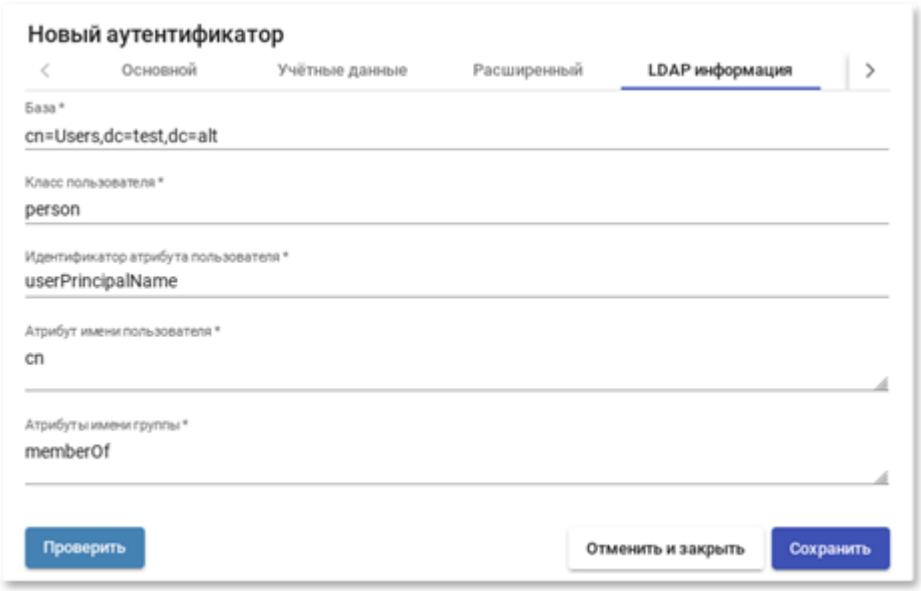

Рис. 87

Примечание. Используя кнопку «Проверить», можно проверить соединение с Active Directory.

4) добавить группу LDAP, в которую входят пользователи. Для этого следует выбрать созданный аутентификатор, затем в открывшемся окне на вкладке «Группы» нажать «Новый» → «Группа».

Заполнить dn существующей группы (например, `cn=UDS,cn=Users,dc=test,dc=alt`), можно также указать разрешенные пулы (рис. 88).

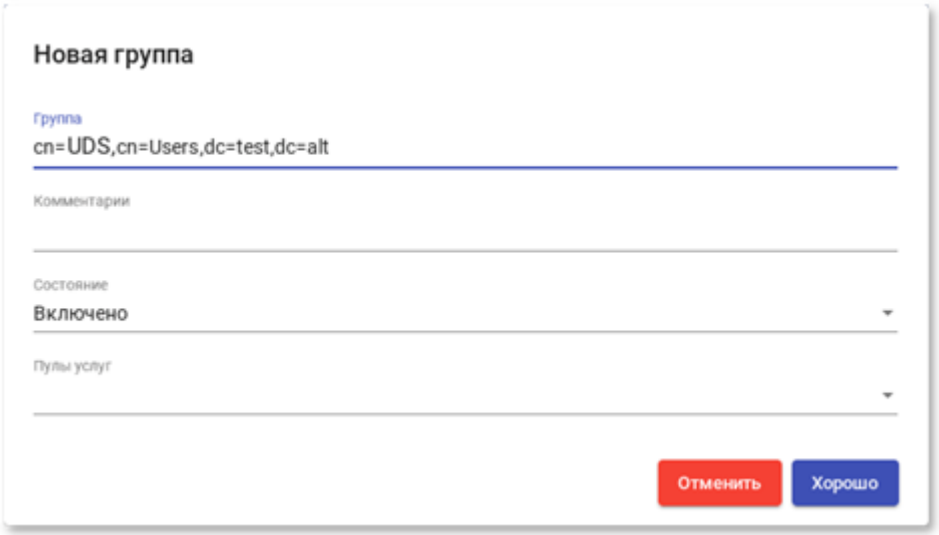

Рис. 88

Примечания:

1. Атрибут `memberOf` является многозначным атрибутом, который содержит группы, из которых пользователь является прямым членом, за исключением основной группы, которая представлена `primaryGroupId`. Поэтому в поле «Группы» не нужно указывать основную группу, например,

```
CN=Domain Users,CN=Users,DC=test,DC=alt
```

или

```
CN=Пользователи домена,CN=Users,DC=test,DC=alt
```

2. На вкладке «Пользователи» аутентификатора пользователи будут добавляться автоматически после первого входа в систему OpenUDS (пользователи должны входить в группы, указанные в аутентификаторе на вкладке «Группа») (рис. 89).

3. Можно зарегистрировать пользователя вручную, чтобы назначить ему специальные права перед первым подключением. Для этого необходимо нажать на кнопку «Новый» и указать пользователя, его статус (включен или отключен) и уровень доступа (поле «Роль»). Не рекомендуется заполнять поле «Группы», так как система должна автоматически добавить пользователя в группу участников (рис. 90).

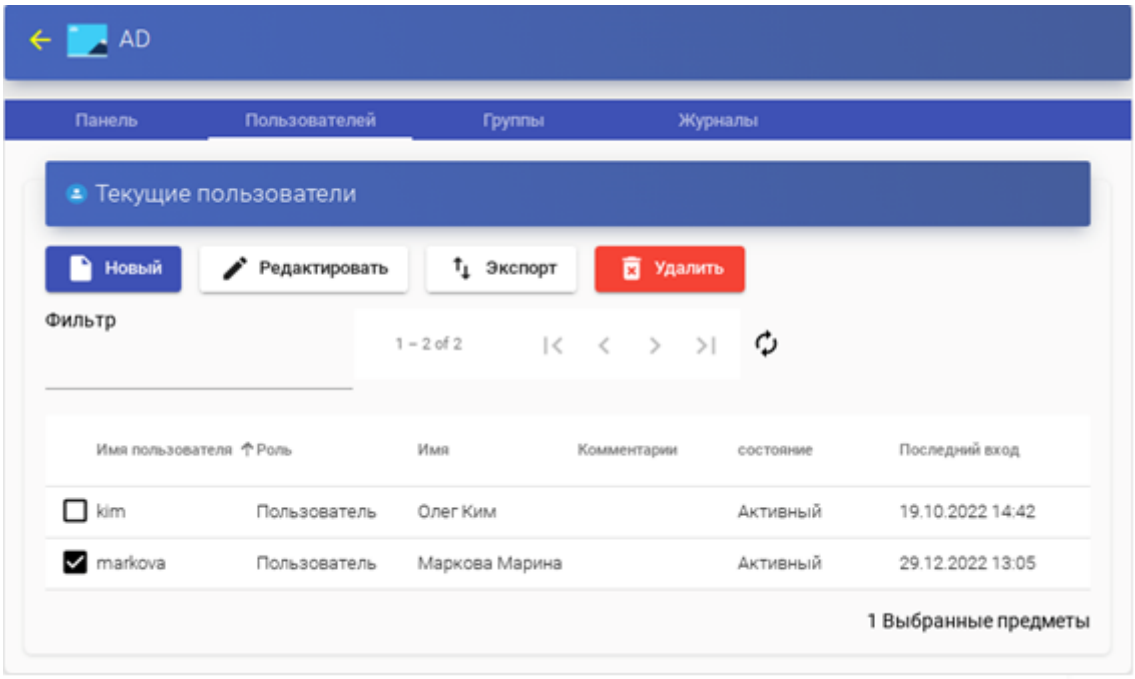

Рис. 89

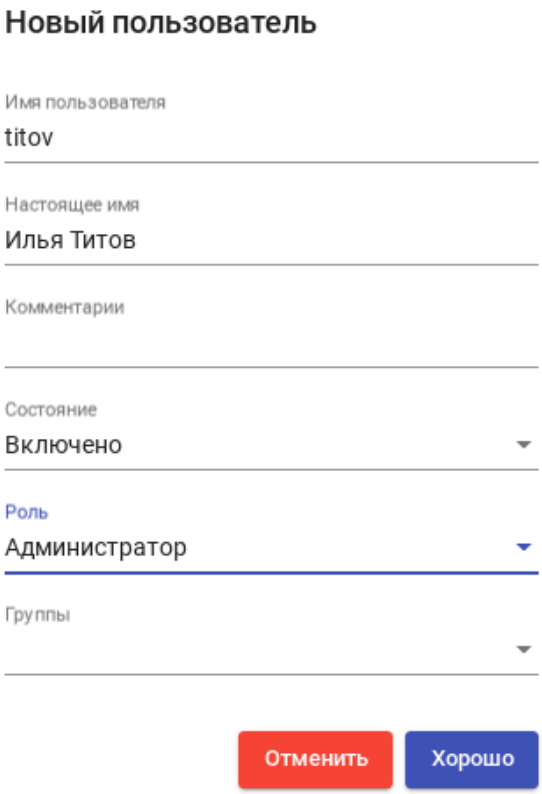

Рис. 90

7.3.2.3. IP аутентификатор

Этот тип аутентификации обеспечивает доступ клиентов к рабочим столам и виртуальным приложениям по их IP-адресу.

Для создания аутентификации типа «IP аутентификатор» в разделе «Аутентификаторы» следует нажать на кнопку: «Новый» → «IP аутентификатор».

Минимальные параметры конфигурации (вкладка «Основной»): имя аутентификатора, приоритет и метка (рис. 91).

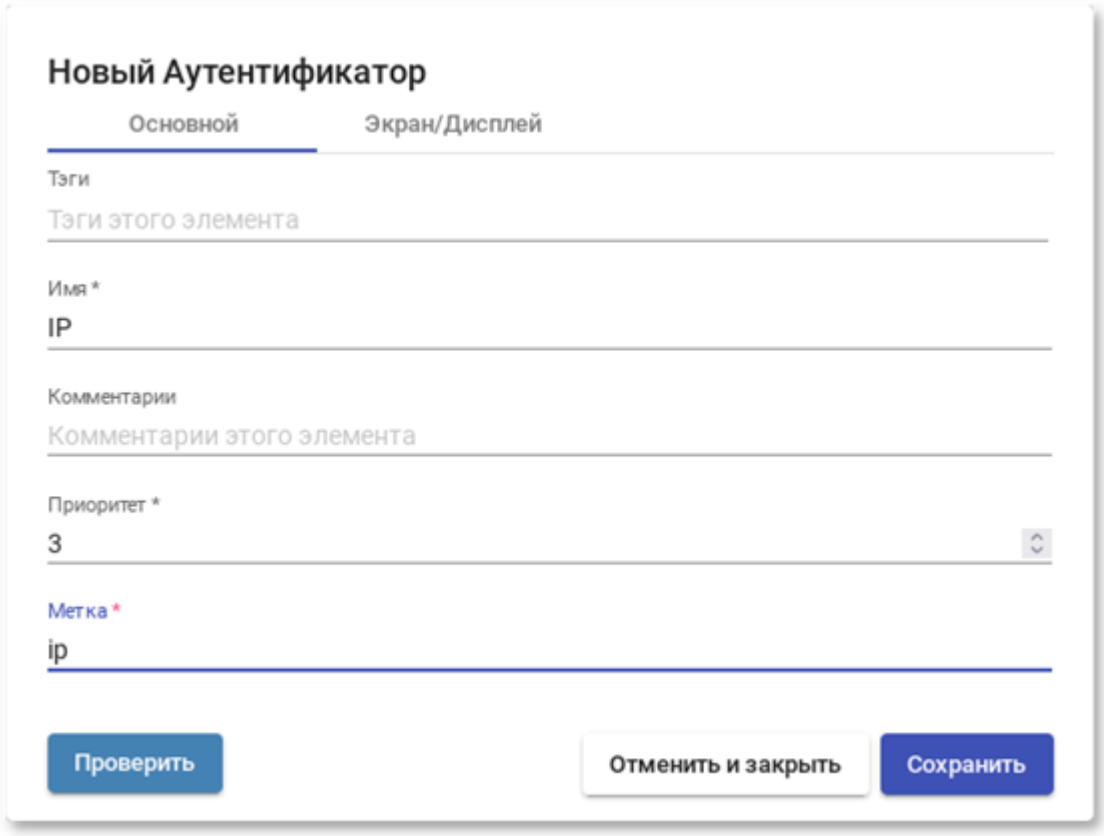

Рис. 91

Настройки на вкладке «Расширенный» (рис. 92):

- видно только из этих сетей – позволяет отфильтровать сети, из которых будет виден аутентификатор;
- разрешить прокси – позволяет корректно определять IP-адреса клиентов подключения, если есть промежуточный компонент для доступа к серверу OpenUDS, например, балансировщик нагрузки (OpenUDS автоматически определяет IP-адрес клиента подключения. В средах, где настроены балансировщики нагрузки, это обнаружение не удается, поскольку IP-адрес соответствует обнаруженным балансировщикам. Включение этой опции обеспечивает правильное определение IP-адреса клиента).

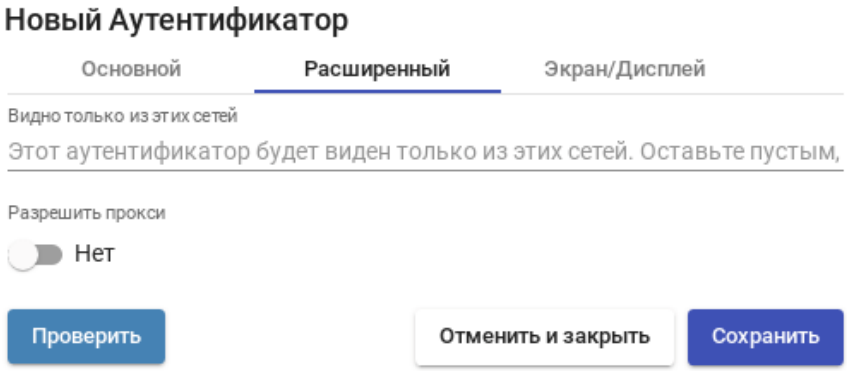

Рис. 92

После того, как аутентификатор типа «IP аутентификатор» создан, следует создать группы пользователей. Группа может представлять собой диапазон IP-адресов (192.168.0.1-192.168.0.55), подсеть (192.168.0.0/24) или отдельные IP-адреса (192.168.0.33, 192.168.0.110) (рис. 93).

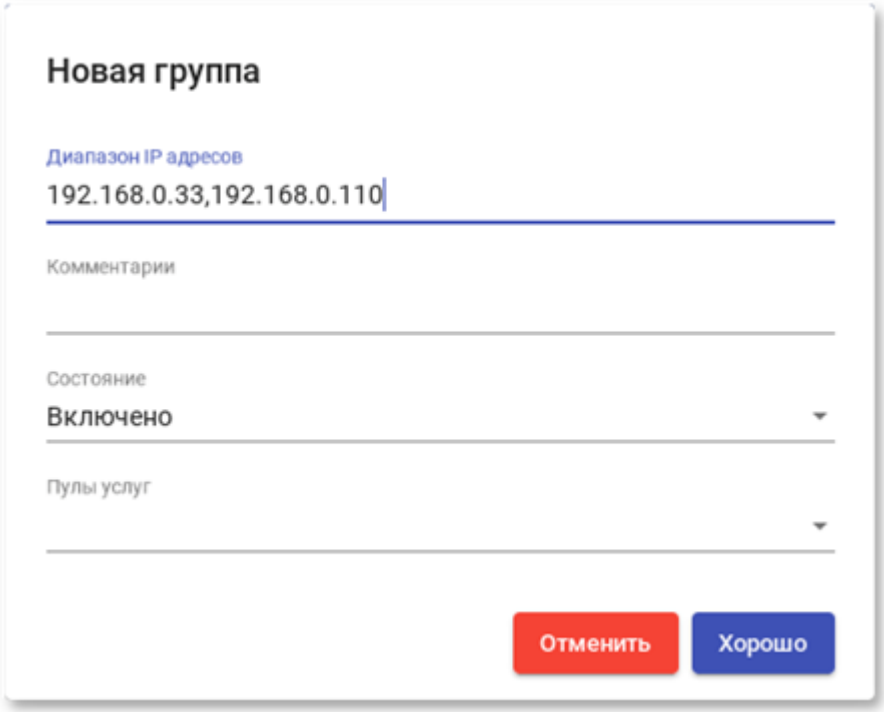

Рис. 93

7.3.3. Настройка менеджера ОС

OpenUDS Actor, размещенный на виртуальном рабочем столе, отвечает за взаимодействие между ОС и OpenUDS Server на основе конфигурации или выбранного типа «Менеджера ОС» (рис. 94).

Примечание. Для каждой службы, развернутой в OpenUDS, потребуется «Менеджер ОС», за исключением случаев, когда используется «Поставщик машин статических IP».

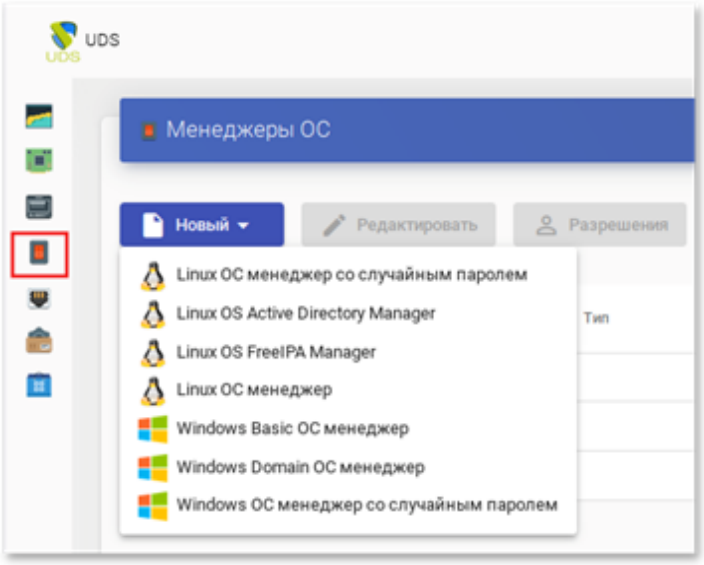

Рис. 94

Менеджер ОС запускает ранее настроенные службы:

- «Linux OS Active Directory Manager» используется для виртуальных рабочих столов на базе Linux, которые являются членами домена AD;
- «Linux OS FreeIPA Manager» используется для виртуальных рабочих столов на базе Linux, которые являются членами домена FreeIPA;
- «Linux ОС менеджер» используется для виртуальных рабочих столов на базе Linux. Он выполняет задачи переименования и управления сеансами виртуальных рабочих столов;
- «Windows Basic ОС менеджер» используется для виртуальных рабочих столов на базе Windows, которые не являются частью домена AD;
- «Windows Domain ОС менеджер» используется для виртуальных рабочих столов на базе Windows, которые являются членами домена AD.

Примечание. Для каждой службы, развернутой в OpenUDS, потребуется «Менеджер ОС», за исключением случаев, когда используется служба «Поставщик машин статических IP».

Минимальные настройки для «Linux OS Active Directory Manager»:

1) вкладка «Основной» (рис. 95):
   - «Имя» – название;
   - «Домен» – домен, к которому будут присоединены виртуальные рабочие столы. Необходимо использовать формат FQDN (например, test.alt);
   - «Аккаунт» – пользователь с правами на добавление машин в домен;
   - «Пароль» – пароль пользователя, указанного в поле «Аккаунт»;
   - «OU» – организационная единица, в которую будут добавлены виртуальные хосты (если не указано, хосты будут зарегистрированы в подразделении по умолчанию – «Computers»). Формат поддерживаемых OU: `OU = name_OU_last_level, … OU = name_OU_first_level, DC = name_domain, DC = extension_domain`. Во избежание ошибок, рекомендуется сверяться с полем `distinguishedName` в свойствах атрибута OU;
   - «Действие при выходе из системы» – действие, которое OpenUDS будет выполнять на виртуальном рабочем столе при закрытии сеанса пользователя. «Держать сервис привязанным – постоянный пул, при выходе пользователя (выключении ВМ), ВМ запускается заново, при повторном входе пользователю будет назначен тот же рабочий стол. «Удалить сервис» – непостоянный пул, при выходе пользователя из системы, ВМ удаляется и создается заново. «Держать сервис привязанным даже в новой публикации» – сохранение назначенной службы даже при создании новой публикации;
   - «Максимальное время простоя» – время простоя виртуального рабочего стола (в секундах). По истечении этого времени OpenUDS Actor автоматически закроет сеанс. Отрицательные значения и значения менее 300 секунд отключают эту опцию;
2) вкладка «Расширенный» (рис. 96):
   - «Client software» – позволяет указать, если это необходимо, способ подключения (SSSD или Winbind);

- «Membership software» – позволяет указать, если это необходимо, утилиту, используемую для подключения к домену (Samba или adcli);
- «Убрать машину» – если этот параметр установлен, OpenUDS удалит запись о виртуальном рабочем столе в указанном подразделении после удаления рабочего стола (необходимо, чтобы пользователь, указанный в поле «Аккаунт», имел права на выполнение данного действия в OU);
- «Использовать SSL» – если этот параметр установлен, будет использоваться SSL-соединение;
- «Automatic ID mapping» – автоматический маппинг ID;
- «Выход по календарю» – если этот параметр установлен, OpenUDS попытается завершить сессию пользователя, когда для текущего соединения истечет время доступа (если параметр не установлен, пользователю будет разрешено продолжить работу).

Примечание. Для возможности ввода компьютера в домен, на неем должен быть доступен сервер DNS, имеющий записи про контроллер домена Active Directory.

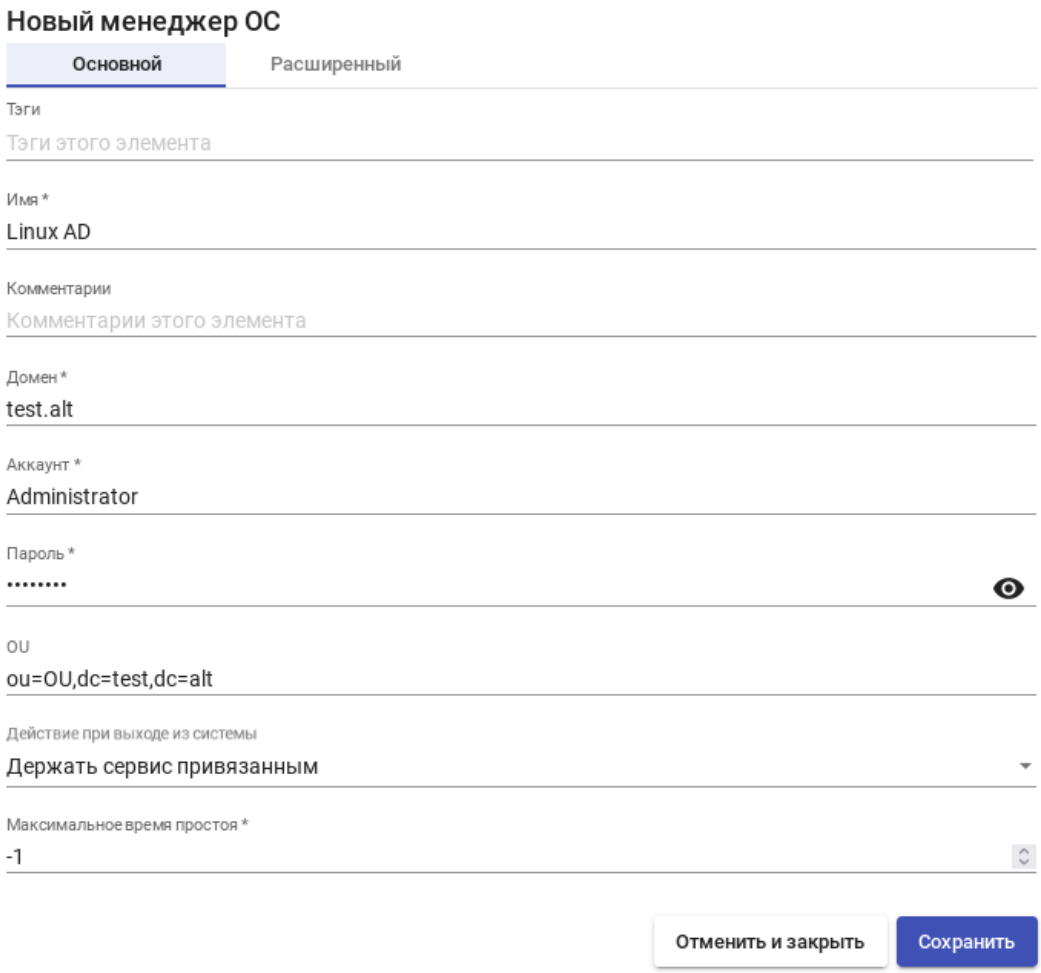

Рис. 95 – OpenUDS. Настройка «OS Linux OS Active Directory Manager»

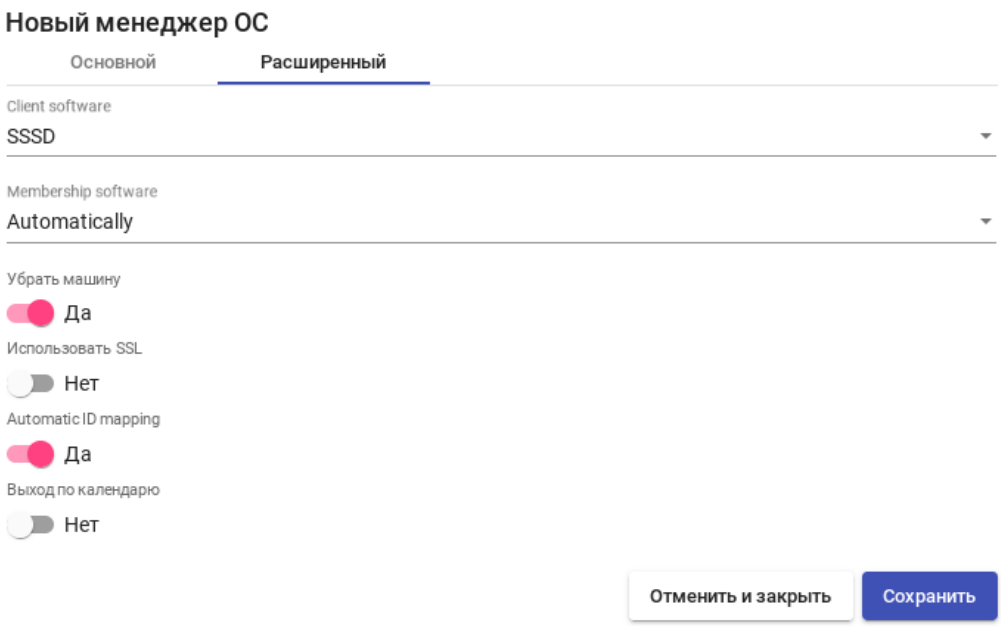

Рис. 96 – OpenUDS. Настройка «OS Linux OS Active Directory Manager»

Минимальные настройки для «Linux OS FreeIPA Manager»:

1) вкладка «Основной» (рис. 97):

- «Имя» – название;
- «Домен» – домен, к которому будут присоединены виртуальные рабочие столы. Необходимо использовать формат FQDN (например, example.test);
- «Аккаунт» – пользователь с правами на добавление машин в домен;
- «Пароль» – пароль пользователя, указанного в поле «Аккаунт»;
- «OU» – организационная единица, в которую будут добавлены виртуальные хосты (если не указано, хосты будут зарегистрированы в подразделении по умолчанию – «Computers»). Формат поддерживаемых OU: `OU = name_OU_last_level, … OU = name_OU_first_level, DC = name_domain, DC = extension_domain`. Во избежание ошибок, рекомендуется сверяться с полем `distinguishedName` в свойствах атрибута OU;
- «Действие при выходе из системы» – действие, которое OpenUDS будет выполнять на виртуальном рабочем столе при закрытии сеанса пользователя. «Держать сервис привязанным – постоянный пул, при выходе пользователя (выключении ВМ), ВМ запускается заново, при повторном входе пользователю будет назначен тот же рабочий стол. «Удалить сервис» – непостоянный пул, при выходе пользователя из системы, ВМ удаляется и создается заново. «Держать сервис привязанным даже в новой публикации» – сохранение назначенной службы даже при создании новой публикации;
- «Максимальное время простоя» – время простоя виртуального рабочего стола (в секундах). По истечении этого времени OpenUDS Actor автоматически закроет сеанс. Отрицательные значения и значения менее 300 секунд отключают эту опцию;

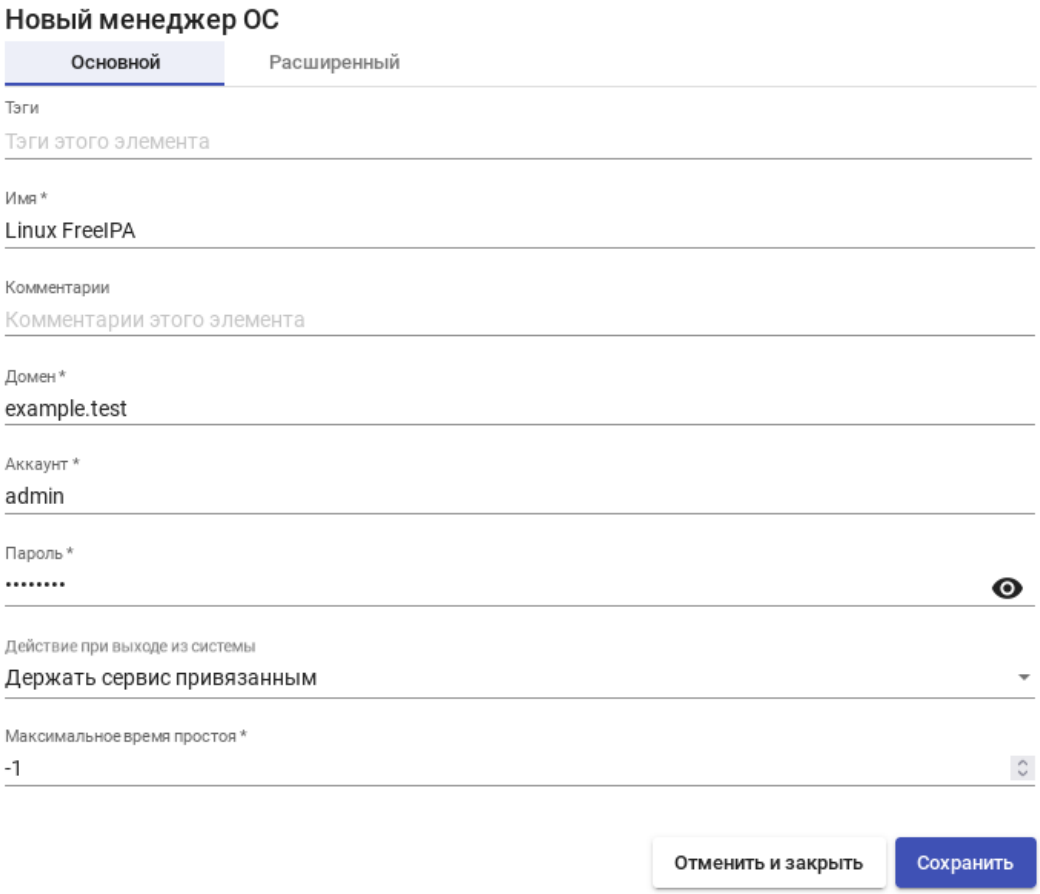

Рис. 97 – OpenUDS. Настройка «OS Linux OS FreeIPA Manager»

2) вкладка «Расширенный» (рис. 98):

- «Client software» – позволяет указать, если это необходимо, способ подключения (SSSD или Winbind);
- «Membership software» – позволяет указать, если это необходимо, утилиту, используемую для подключения к домену (Samba или adcli);
- «Убрать машину» – если этот параметр установлен, OpenUDS удалит запись о виртуальном рабочем столе в указанном подразделении после удаления рабочего стола (необходимо, чтобы пользователь, указанный в поле «Аккаунт», имел права на выполнение данного действия в OU);
- «Использовать SSL» – если этот параметр установлен, будет использоваться SSL-соединение;
- «Automatic ID mapping» – автоматический маппинг ID;
- «Выход по календарю» – если этот параметр установлен, OpenUDS попытается завершить сессию пользователя, когда для текущего

соединения истечет время доступа (если параметр не установлен, пользователю будет разрешено продолжить работу).

Примечание. Для возможности ввода компьютера в домен, на нем должен быть доступен сервер DNS, имеющий записи про сервер FreeIPA.

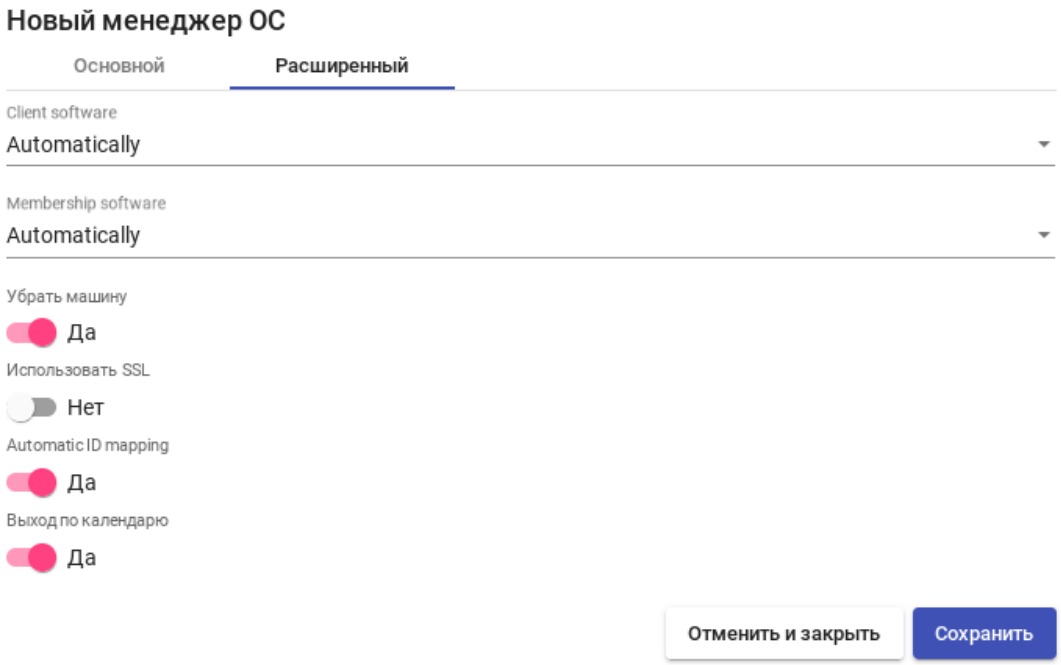

Рис. 98 – OpenUDS. Настройка «OS Linux OS FreeIPA Manager»

Минимальные настройки для Linux ОС менеджер и Windows Basic ОС менеджер:

1) вкладка «Основной» (рис. 99):

- «Имя» – название;
- «Действие при выходе из системы» – действие, которое OpenUDS будет выполнять на виртуальном рабочем столе при закрытии сеанса пользователя. Держать сервис привязанным – постоянный пул, при выходе пользователя (выключении ВМ), ВМ запускается заново, при повторном входе пользователю будет назначен тот же рабочий стол. Удалить сервис – непостоянный пул, при выходе пользователя из системы, ВМ удаляется и создается заново. Держать сервис привязанным даже в новой публикации – сохранение назначенной службы даже при создании новой публикации;

- «Максимальное время простоя» – время простоя виртуального рабочего стола (в секундах). По истечении этого времени OpenUDS Actor автоматически закроет сеанс. Отрицательные значения и значения менее 300 секунд отключают эту опцию.

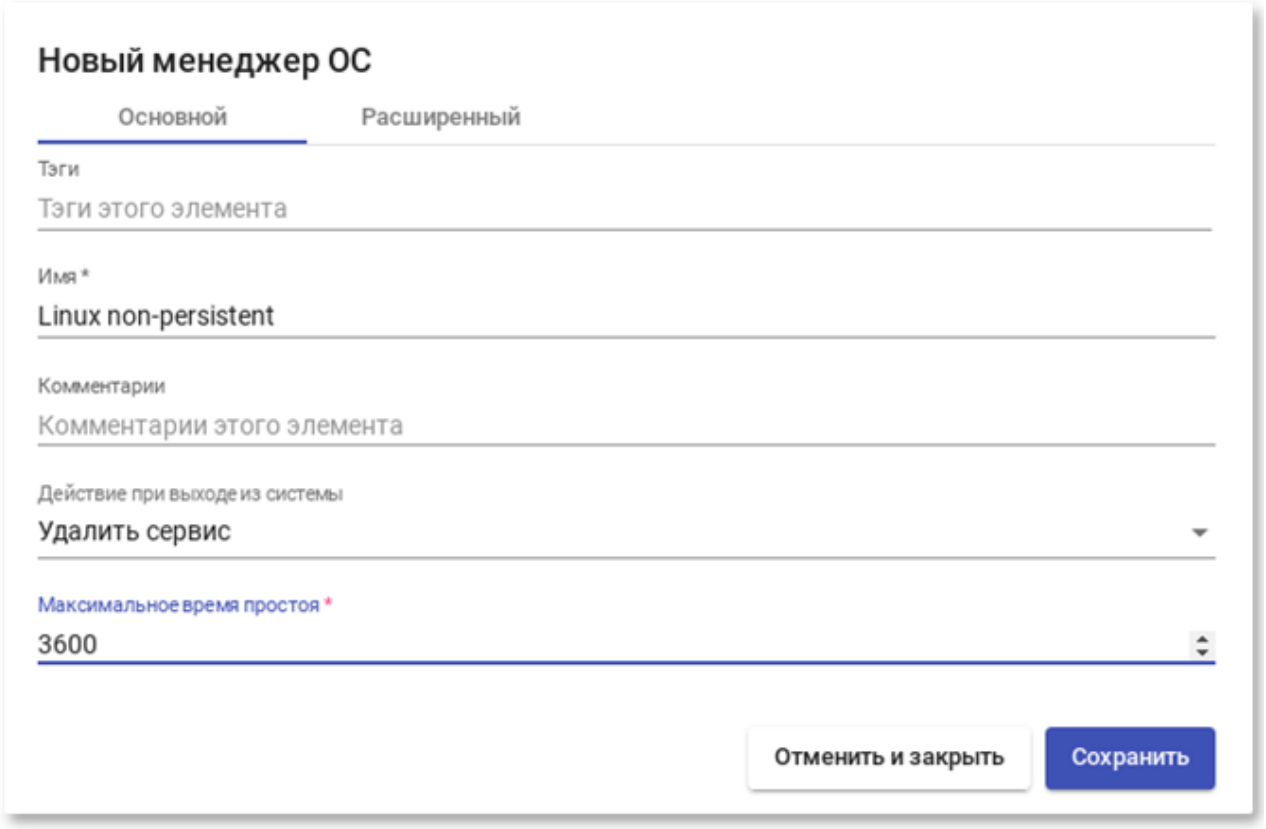

Рис. 99

2) вкладка «Расширенный»:

- «Выход из календаря» – если этот параметр установлен, OpenUDS попытается завершить сессию пользователя, когда для текущего соединения истечет время доступа (если параметр не установлен, пользователю будет разрешено продолжить работу).

Минимальные настройки для Windows Domain ОС менеджер:

1) вкладка «Основной» (рис. 100):

- «Имя» – название;
- «Домен» – домен, к которому будут присоединены виртуальные рабочие столы. Необходимо использовать формат FQDN (например, test.alt);

- «Аккаунт» – пользователь с правами на добавление машин в домен;
- «Пароль» – пароль пользователя, указанного в поле «Аккаунт»;
- «OU» – организационная единица, в которую будут добавлены виртуальные хосты (если не указано, хосты будут зарегистрированы в подразделении по умолчанию – Computers). Формат поддерживаемых OU: `OU = name_OU_last_level, … OU = name_OU_first_level, DC = name_domain, DC = extension_domain`. Во избежание ошибок, рекомендуется сверяться с полем «distinguishedName» в свойствах атрибута OU;
- «Действие при выходе из системы» – действие, которое OpenUDS будет выполнять на виртуальном рабочем столе при закрытии сеанса пользователя. Держать сервис привязанным (Keep service assigned) – постоянный пул, при выходе пользователя (выключении ВМ), ВМ запускается заново, при повторном входе пользователю будет назначен тот же рабочий стол. Удалить сервис (Remove service) – непостоянный пул, при выходе пользователя из системы, ВМ удаляется и создается заново. Держать сервис привязанным даже в новой публикации (Keep service assigned even on new publication) – сохранение назначенной службы даже при создании новой публикации;
- «Максимальное время простоя» – время простоя виртуального рабочего стола (в секундах). По истечении этого времени OpenUDS Actor автоматически закроет сеанс. Отрицательные значения и значения менее 300 секунд отключают эту опцию.

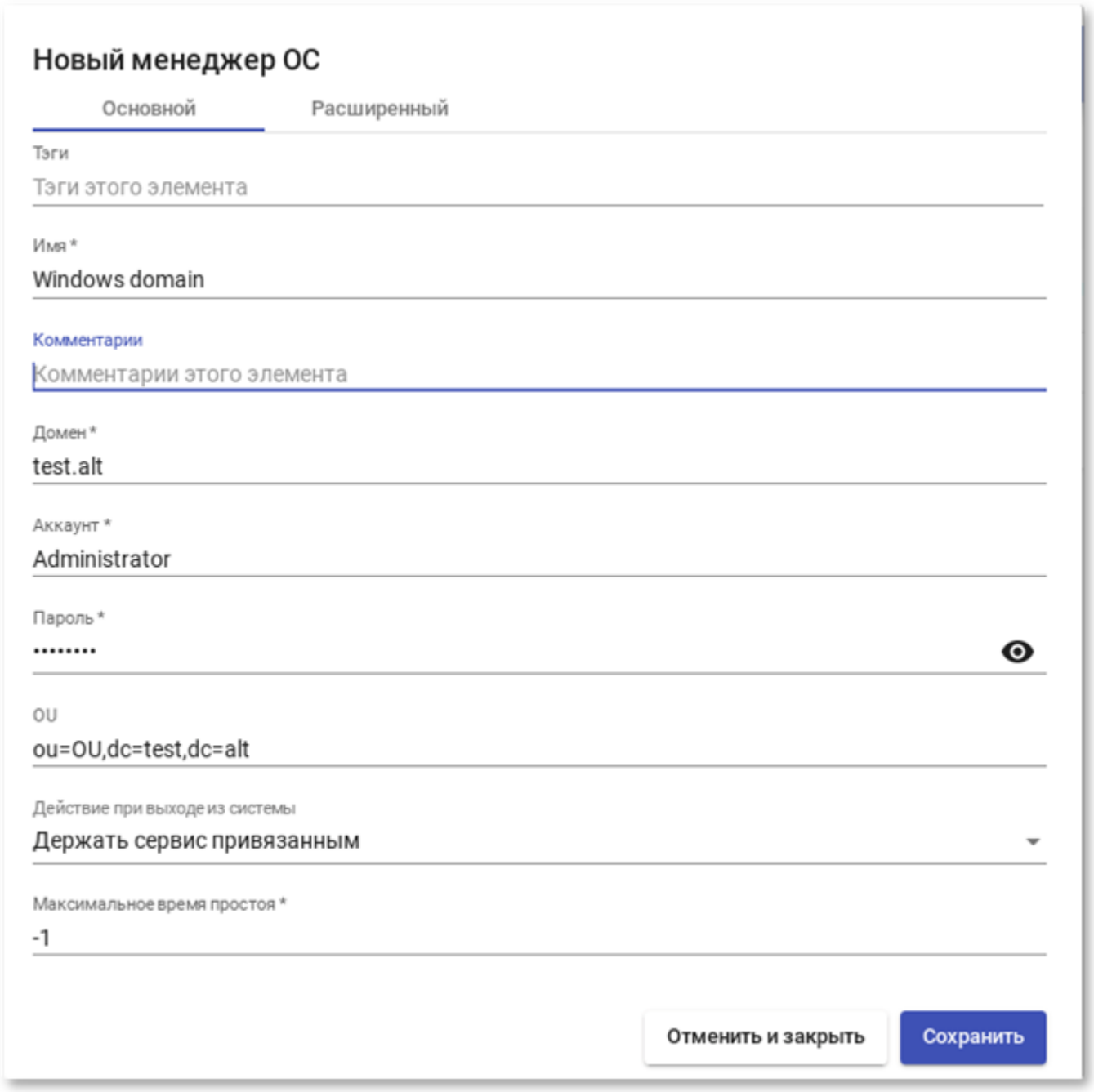

Рис. 100

П р и м е ч а н и е . Для возможности ввода компьютера в домен, на нем должен быть доступен сервер DNS, имеющий записи про контроллер домена Active Directory.

2) вкладка «Расширенный» (рис. 101):

- «Группа машин» – указывает, к какой группе машин AD будут добавлены виртуальные рабочие столы, созданные UDS;

- «Убрать машину» – если этот параметр установлен, OpenUDS удалит запись о виртуальном рабочем столе в указанном подразделении после удаления рабочего стола (необходимо, чтобы пользователь, указанный в поле «Аккаунт», имел права на выполнение данного действия в OU);
- «Предпочтения серверов» – если серверов AD несколько, можно указать, какой из них использовать предпочтительнее;
- «Использовать SSL» – если этот параметр установлен, будет использоваться SSL-соединение;
- «Выход из календаря» – если этот параметр установлен, OpenUDS попытается завершить сессию пользователя, когда для текущего соединения истечет время доступа (если параметр не установлен, пользователю будет разрешено продолжить работу).

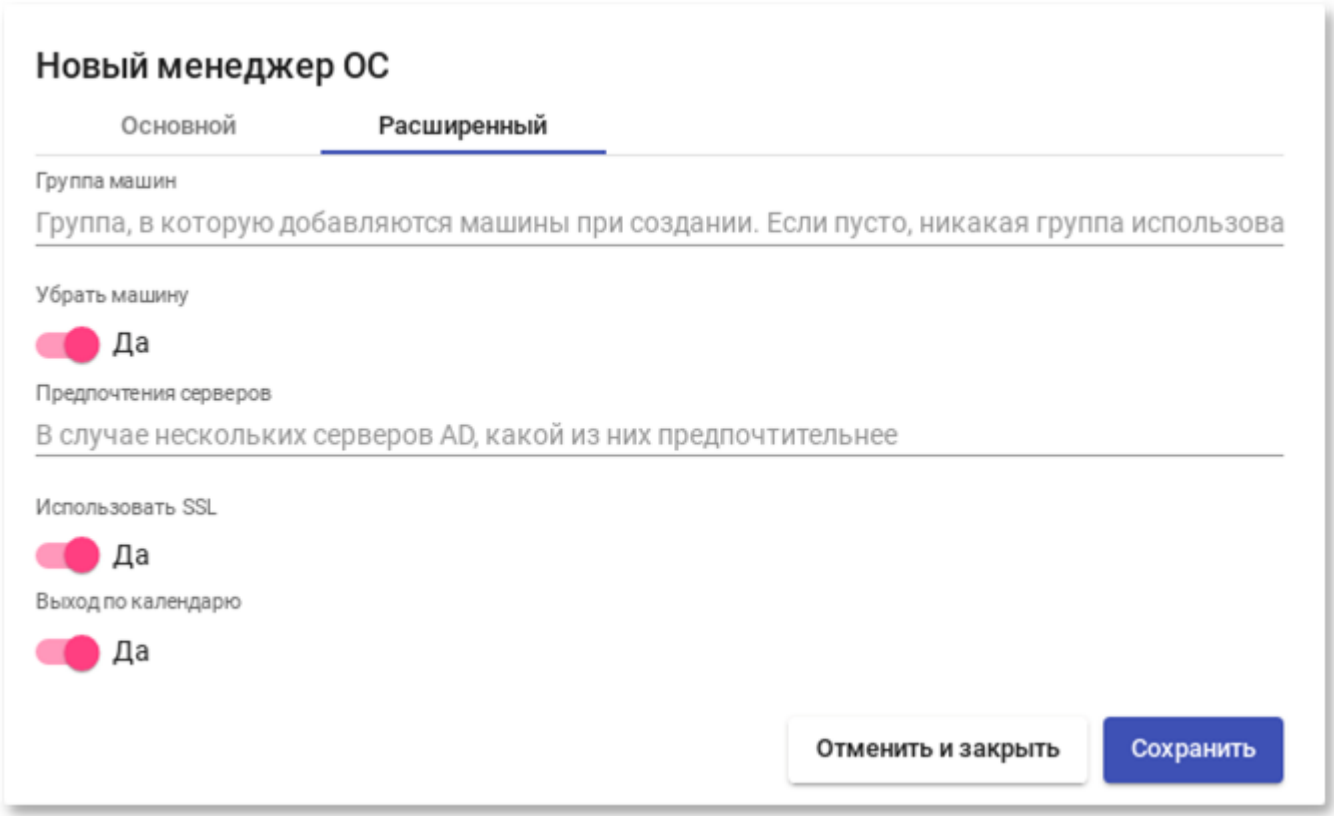

Рис. 101

7.3.4. Транспорт

Для подключения к виртуальным рабочим столам необходимо создать транспорт. Транспорт – это приложение, которое выполняется на клиенте и отвечает за предоставление доступа к реализованной службе.

Можно создать один транспорт для различных «пулов» или установить по одному транспорту для каждого «пула».

При создании транспорта необходимо выбрать его тип (рис. 102):

- «Прямой» – используется, если пользователь имеет доступ к виртуальным рабочим столам из внутренней сети (например, LAN, VPN и т. д.);
- «Туннельный» – используется, если у пользователя нет прямого подключения к рабочему столу.

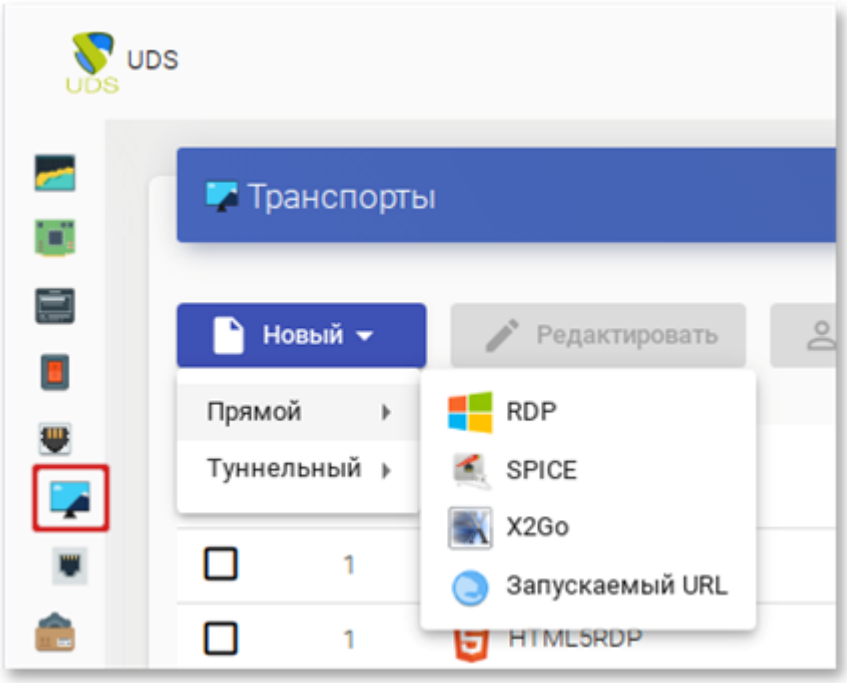

Рис. 102

7.3.4.1. RDP (прямой)

Данный транспорт позволяет пользователям получать доступ к виртуальным рабочим столам Windows/Linux. И на клиентах подключения, и на виртуальных рабочих столах должен быть установлен и включен протокол RDP (для виртуальных рабочих столов Linux необходимо использовать XRDP).

Параметры конфигурации для настройки транспорта RDP:

1) вкладка «Основной» (рис. 103):

- «Имя» – название транспорта;
- «Приоритет» – приоритет, чем меньше значение приоритета, тем выше данный транспорт будет указан в списке доступных транспортов для сервиса. Транспорт с самым низким приоритетом, будет транспортом по умолчанию;
- «Сетевой доступ» – разрешает или запрещает доступ пользователей к службе, в зависимости от сети из которой осуществляется доступ;
- «Сети» – сетевые диапазоны, подсети или IP-адреса. Пустое поле означает «все сети». Используется вместе с параметром «Сетевой доступ»;

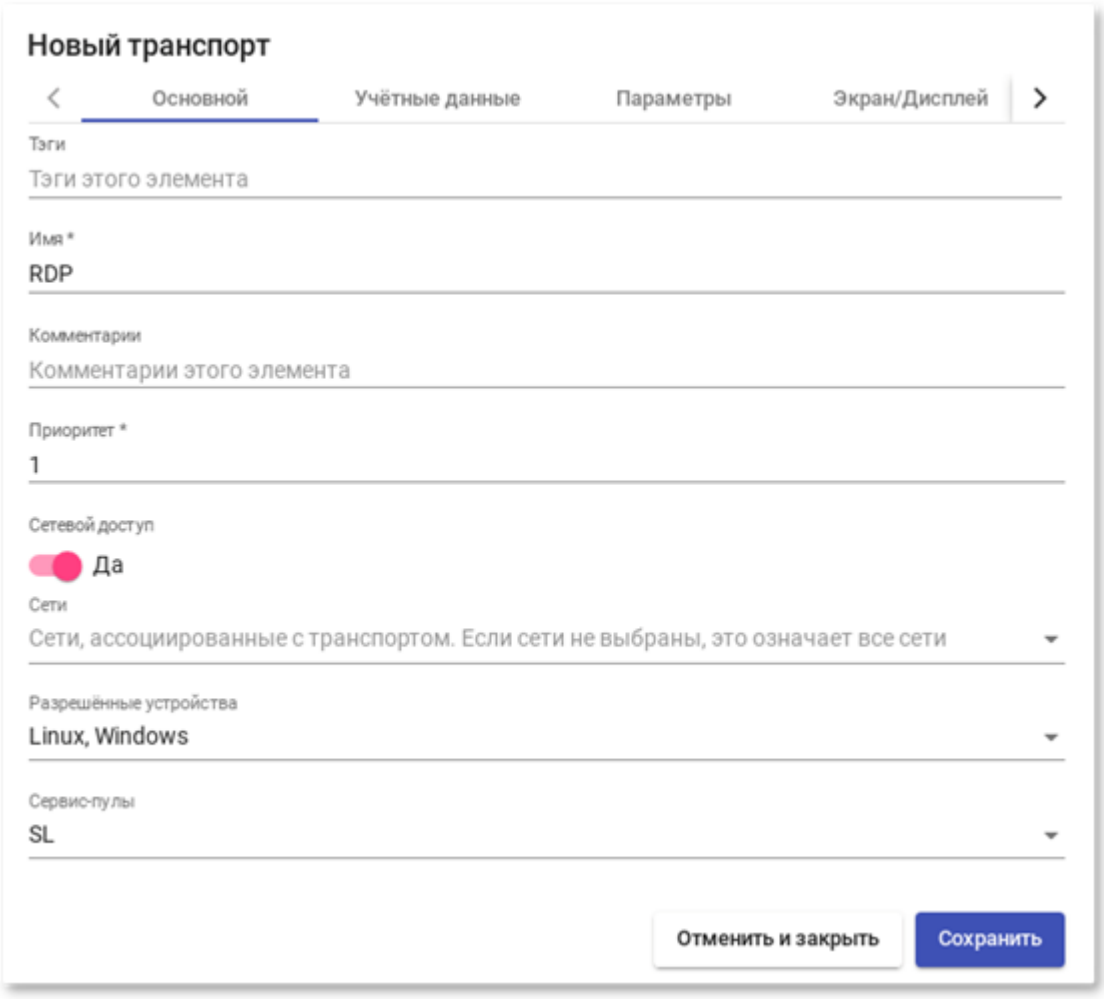

Рис. 103

- «Разрешенные устройства» – разрешает доступ к службе только с выбранных устройств. Пустое поле означает «все устройства»;
- «Сервис-пулы» – позволяет назначить транспорт одному или нескольким ранее созданным пулам услуг. Можно оставить это поле пустым и выбрать способы подключения при создании пула услуг;

2) вкладка «Учетные данные» (рис. 104):

- «Пропустить данные аккаунта» – если установлено значение «Да», учетные данные для доступа к виртуальному рабочему столу будут запрашиваться при подключении к серверу. Если установлено значение «Нет», будут использоваться данные OpenUDS (рис. 104);
- «Имя пользователя» – имя пользователя, которое будет использоваться для доступа к рабочему столу (пользователь должен существовать на ВМ). Если данное поле пустое, будет использован логин авторизовавшегося в веб-интерфейсе OpenUDS пользователя;
- «Пароль» – пароль пользователя, указанного в поле «Имя пользователя»;
- «Без домена» – указывает, перенаправляется ли доменное имя вместе с пользователем. Значение «Да» равносильно пустому полю «Домен»;
- «Домен» – домен. Если поле не пустое, то учетные данные будут использоваться в виде DOMAIN\user;

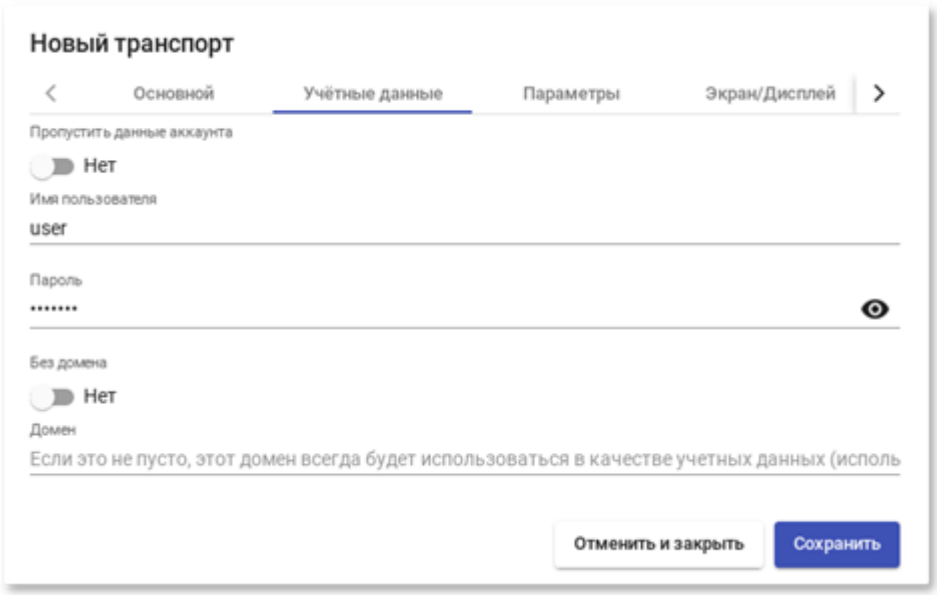

Рис. 104

3) на вкладке «Параметры» можно разрешить/запретить перенаправления дисков, принтеров и других устройств (рис. 105):

- «Разрешить смарткарты» – разрешить перенаправление смарт-карт;
- «Разрешить принтеры» – включить перенаправление принтеров;
- «Политика локальных дисков» – включить перенаправление дисков:
    а) «Allow none» – не перенаправлять диски;
    б) «Allow PnP drives» – во время активного сеанса перенаправлять только подключенные диски;
    в) «Allow any drive» – перенаправлять все диски;
- «Принудительное подключение дисков» – принудительное перенаправление определенных дисков;
- «Разрешить серийные порты» – включить перенаправление последовательного порта;
- «Включить буфер обмена» – разрешить общий буфер обмена;
- «Включить звук» – перенаправлять звук с рабочего стола на клиент подключения;
- «Включить веб-камеру» – перенаправлять веб-камеру;
- «USB redirection» – включить перенаправление USB;
- «Поддержка Credssp» – использовать «redential Security Support Provider»;
- «Порт RDP» – порт RDP (по умолчанию 3389);

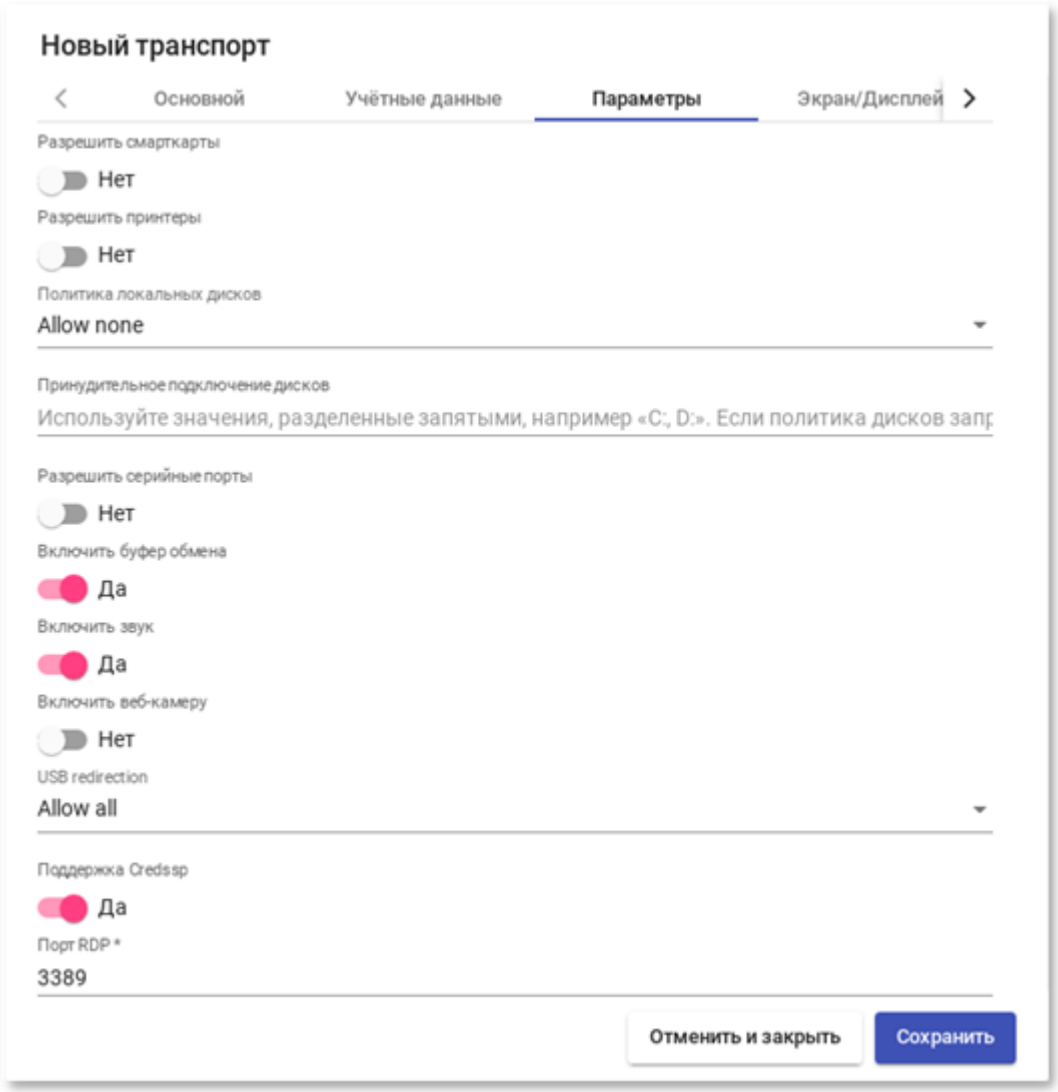

Рис. 105

4) на вкладке «Экран/Дисплей» настраиваются параметры окна рабочего стола (рис. 106):

- «Размер экрана» – размер окна рабочего стола;
- «Глубина цвета» – глубина цвета;
- «Обои/темы» – отображать фона рабочего стола;
- «Несколько мониторов» – использовать несколько мониторов (только для клиентов Windows);
- «Разрешить композицию рабочего стола» – включить Desktop Composition;
- «Сглаживание шрифтов» – активирует сглаживание шрифтов;

- «Окно подключения» – показывать панель подключения (только для клиентов Windows);

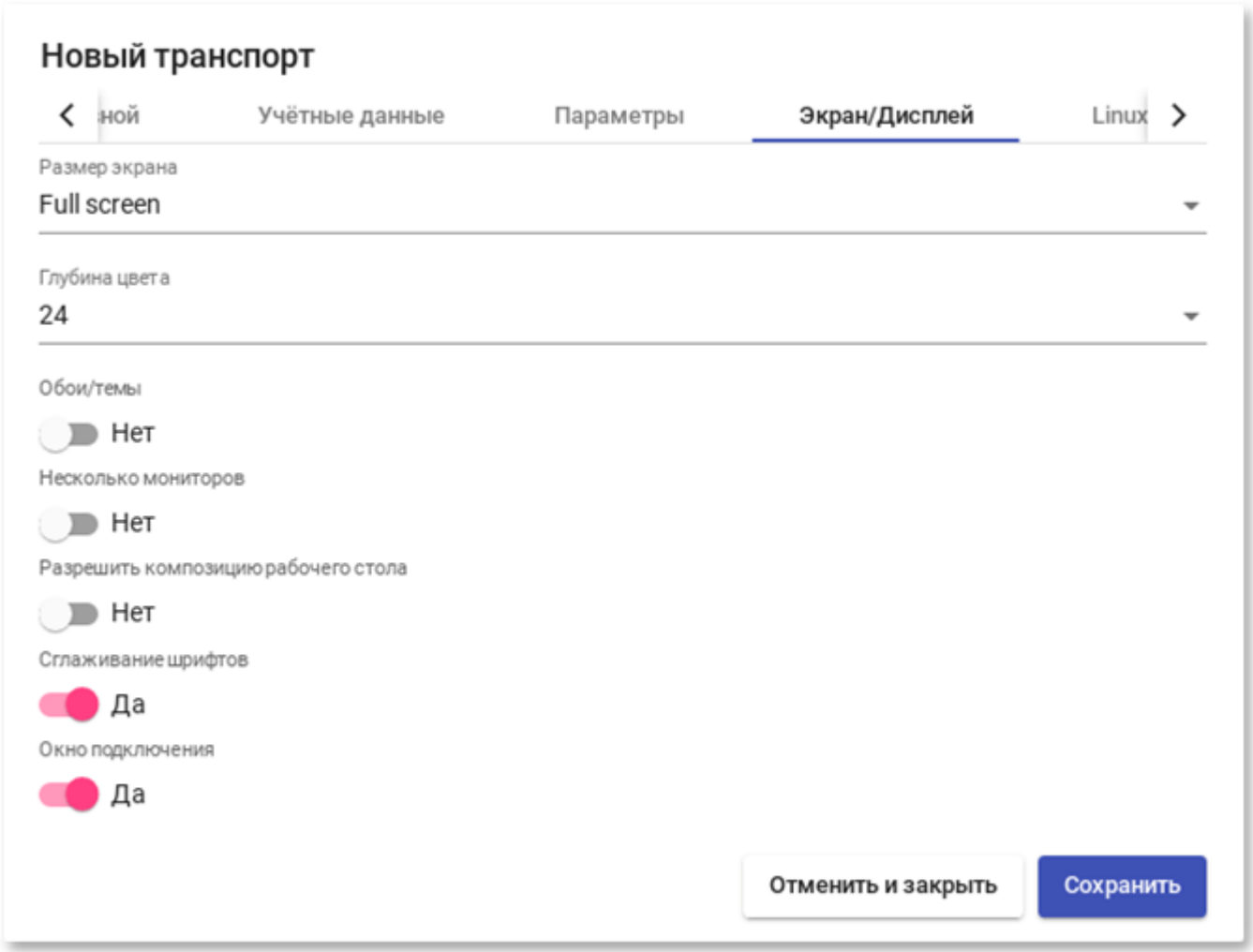

Рис. 106

5) вкладка «Linux Client» (рис. 107):

- «Мультимедийная синхронизация» – включает параметр мультимедиа на клиенте FreeRDP;
- «Использовать Alsa» – использовать звук через Alsa;
- «Строка принтера» – принтер, используемый клиентом FreeRDP (если включено перенаправление принтера). Пример: «HP_LaserJet_M1536dnf_MFP» (названия подключенных принтеров можно вывести командой `lpstat -a`);

- «Строка Smartcard» – токен, используемый клиентом FreeRDP (если включено перенаправление смарт-карт). Пример: «Aktiv Rutoken ECP 00 00»;
- «Пользовательские параметры» – здесь можно указать любой параметр, поддерживаемый клиентом FreeRDP;

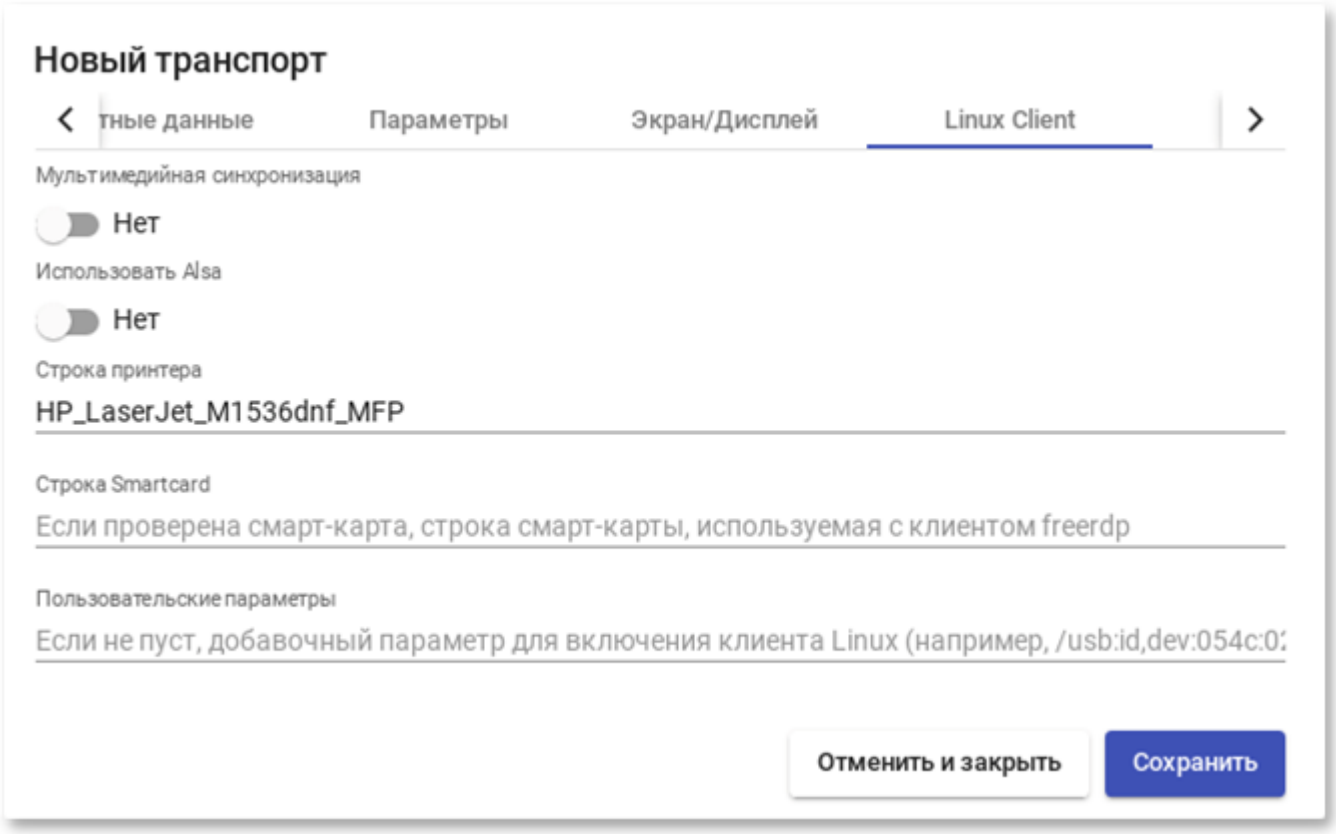

Рис. 107

6) вкладка «Расширенный»:

- «Метка» – метка транспорта метапула (используется для того чтобы назначить несколько транспортов метапулу).

7.3.4.2. RDP (туннельный)

Все настройки аналогичны настройке RDP, за исключением настроек на вкладке «Туннель».

Вкладка «Туннель» (рис. 108):

- «Туннельный сервер» – IP-адрес/имя OpenUDS Tunnel. Если доступ к рабочему столу осуществляется через глобальную сеть, необходимо ввести

общедоступный IP-адрес сервера OpenUDS Tunnel. Формат: IP_Tunneler:Port;

- «Время ожидания туннеля» – максимальное время ожидания туннеля;
- «Принудительная проверка SSL-сертификата» – принудительная проверка сертификата туннельного сервера.

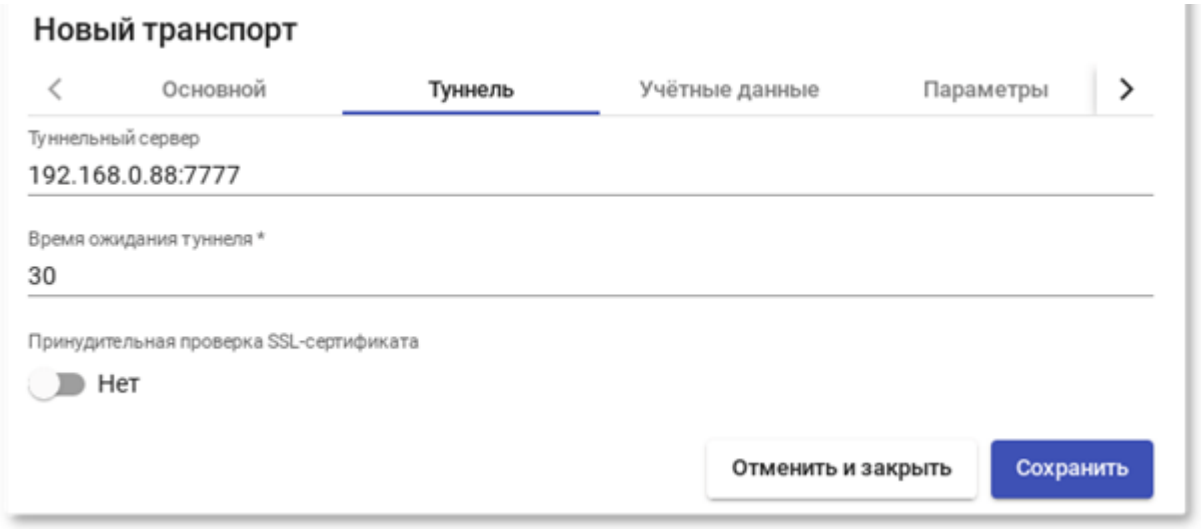

Рис. 108

7.3.4.3. X2Go (прямой)

X2Go позволяет пользователям получать доступ к виртуальным рабочим столам Linux. На клиентах подключения должен быть установлен клиент X2Go, и на виртуальных рабочих столах (сервере) должен быть установлен и включен сервер X2Go.

Параметры конфигурации для настройки транспорта X2Go:

1) вкладка «Основной» (рис. 109):
   - «Имя» – название транспорта;
   - «Приоритет» – приоритет, чем меньше значение приоритета, тем выше данный транспорт будет указан в списке доступных транспортов для сервиса. Транспорт с самым низким приоритетом, будет транспортом по умолчанию;
   - «Сетевой доступ» – разрешает или запрещает доступ пользователей к службе, в зависимости от сети из которой осуществляется доступ;

- «Сети» – сетевые диапазоны, подсети или IP-адреса (настраиваются в разделе «Сети»). Пустое поле означает «все сети». Используется вместе с параметром Сетевой доступ;
- «Разрешенные устройства» – разрешает доступ к службе только с выбранных устройств. Пустое поле означает «все устройства»;

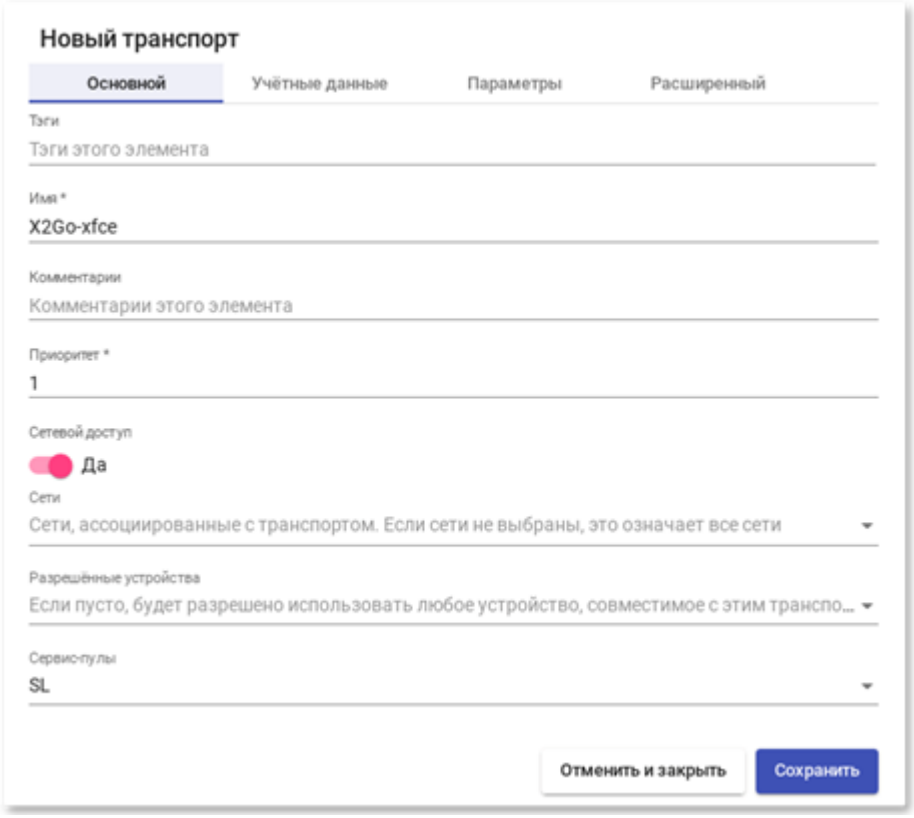

Рис. 109

- «Сервис-пулы» – позволяет назначить транспорт одному или нескольким ранее созданным пулам услуг. Можно оставить это поле пустым и выбрать способы подключения при создании пула услуг;

2) вкладка «Учетные данные» (рис. 110):

- «Имя пользователя» – имя пользователя, которое будет использоваться для доступа к рабочему столу (пользователь должен существовать на ВМ). Если данное поле пустое, будет использован логин авторизовавшегося в веб-интерфейсе OpenUDS пользователя;

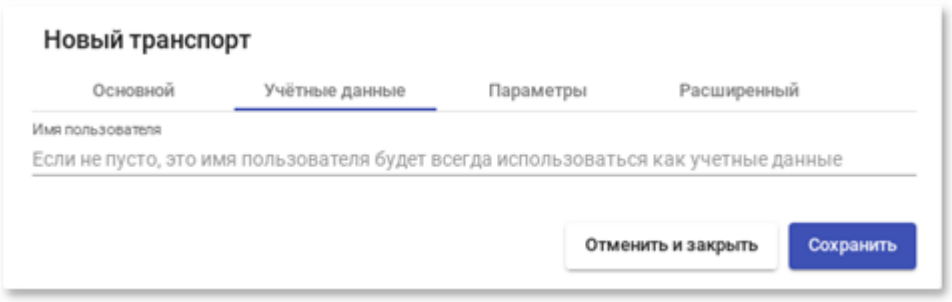

Рис. 110

3) вкладка «Параметры» (рис. 111):

- «Размер экрана» – размер окна рабочего стола;
- «Экран» – менеджер рабочего стола (Xfce, Mate и др.) или виртуализация приложений Linux (UDS vAPP);
- «vAPP» – полный путь до приложения (если в поле Экран выбрано значение UDS vAPP);
- «Включить звук»;
- «Перенаправить домашнюю папку» – перенаправить домашнюю папку клиента подключения на виртуальный рабочий стол (на Linux также перенаправлять /media);
- «Скорость» – скорость подключения.

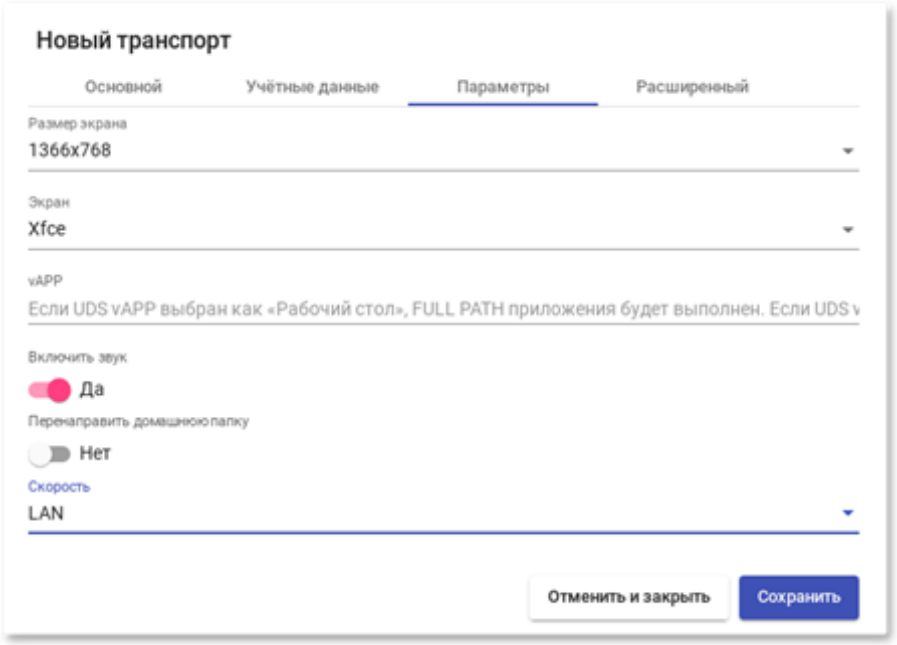

Рис. 111

4) вкладка «Расширенный» (рис. 112):

- «Звук» – тип звукового сервера;
- «Клавиатура» – раскладка клавиатуры;
- «Метка» – метка транспорта метапула (используется для того чтобы назначить несколько транспортов метапулу).

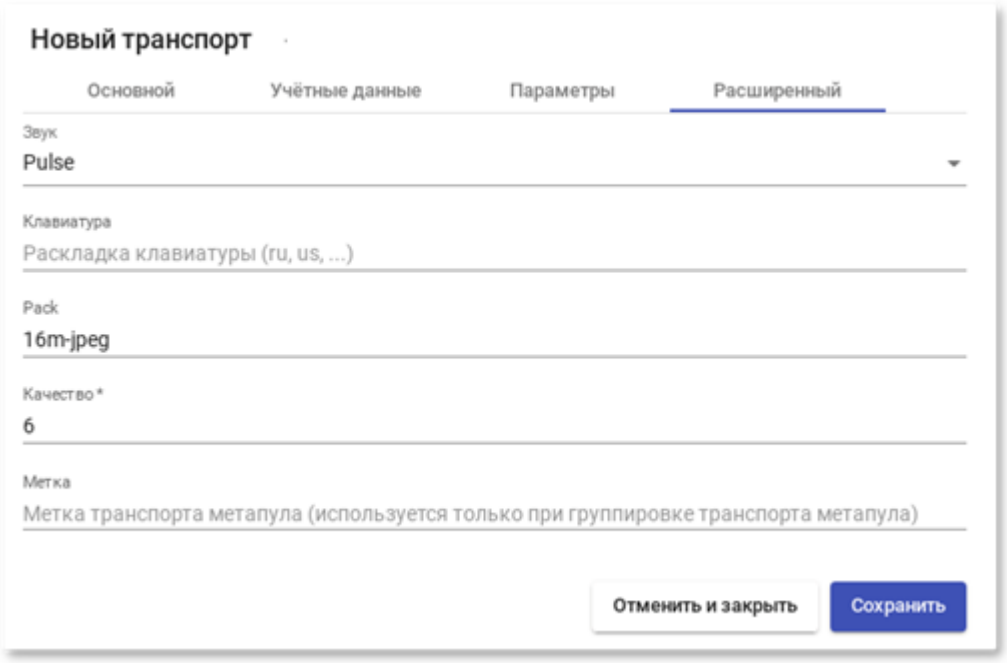

Рис. 112

7.3.4.4. X2Go (туннельный)

Все настройки аналогичны настройке X2Go, за исключением настроек на вкладке «Туннель».

Вкладка «Туннель» (рис. 113):

- «Туннельный сервер» – IP-адрес/имя OpenUDS Tunnel. Если доступ к рабочему столу осуществляется через глобальную сеть, необходимо ввести общедоступный IP-адрес сервера OpenUDS Tunnel. Формат: IP_Tunneler:Port;
- «Время ожидания туннеля» – максимальное время ожидания туннеля;
- «Принудительная проверка SSL-сертификата» – принудительная проверка сертификата туннельного сервера.

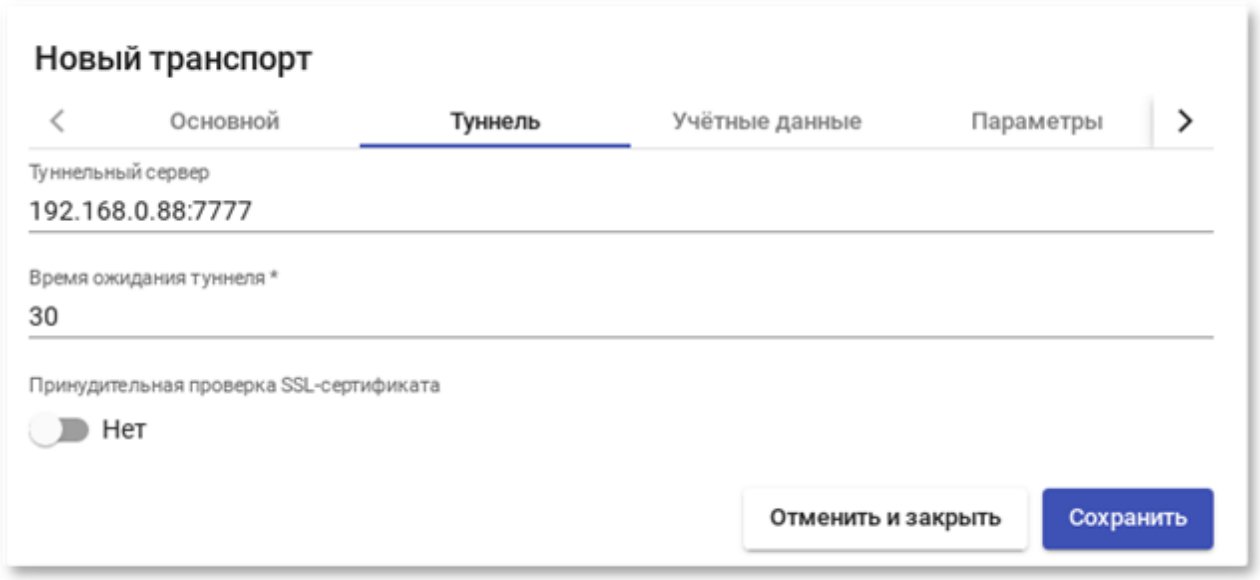

Рис. 113

7.3.4.5. SPICE (прямой)

Примечание. Транспортный протокол SPICE может использоваться только с oVirt/RHEV, OpenNebula и PVE.

SPICE позволяет пользователям получать доступ к виртуальным рабочим столам Windows/Linux. На клиентах подключения должен быть установлен клиент SPICE (virt-manager).

ВАЖНО
Для работы прямого подключения по протоколу SPICE на сервере OpenUDS и клиентах OpenUDS, откуда осуществляется подключение, имена узлов платформы виртуализации должны корректно разрешаться в IP-адреса этих узлов.

Параметры конфигурации для настройки транспорта SPICE:

1) вкладка «Основной» (рис. 114):

- «Имя» – название транспорта;
- «Приоритет» – приоритет, чем меньше значение приоритета, тем выше данный транспорт будет указан в списке доступных транспортов для сервиса. Транспорт с самым низким приоритетом, будет транспортом по умолчанию;

- «Сертификат» – сертификат, сгенерированный в ovirt-engine/RHV-manager или в OpenNebula. Требуется для подключения к виртуальным рабочим столам;
- «Сетевой доступ» – разрешает или запрещает доступ пользователей к службе, в зависимости от сети из которой осуществляется доступ;
- «Сети» – сетевые диапазоны, подсети или IP-адреса (настраиваются в разделе «Сети»). Пустое поле означает «все сети». Используется вместе с параметром «Сетевой доступ»;
- «Разрешенные устройства» – разрешает доступ к службе только с выбранных устройств. Пустое поле означает «все устройства»;
- «Сервис-пулы» – позволяет назначить транспорт одному или нескольким ранее созданным пулам услуг. Можно оставить это поле пустым и выбрать способы подключения при создании пула услуг;

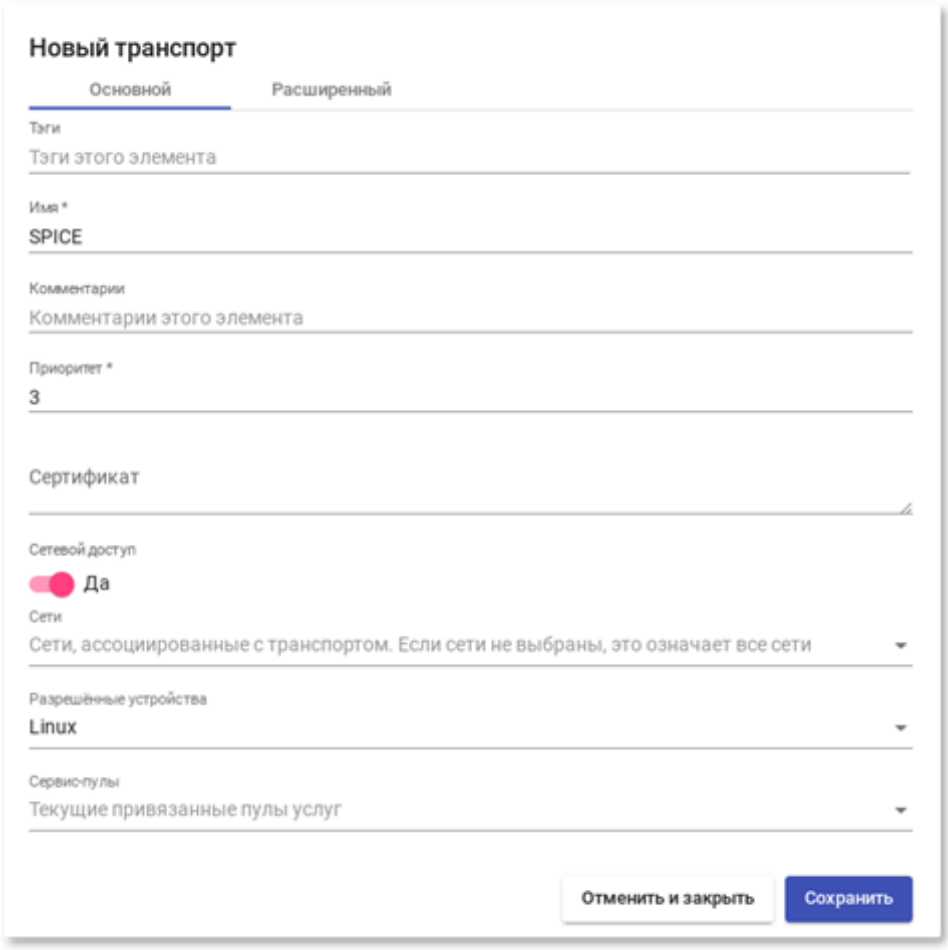

Рис. 114

2) вкладка «Расширенный» (рис. 115):

- «Полноэкранный режим» – включает полноэкранный режим виртуального рабочего стола;
- «Перенаправление смарткарты» – включает перенаправление смарт-карт;
- «Включить USB» – разрешает перенаправление устройств, подключенных к USB-порту;
- «Новый USB автообмен» – позволяет перенаправлять PnP-устройства, подключенные к USB-порту;
- «SSL Connection» – использовать SSL-соединение;
- «Метка» – метка транспорта метапула (используется для того чтобы назначить несколько транспортов метапулу).

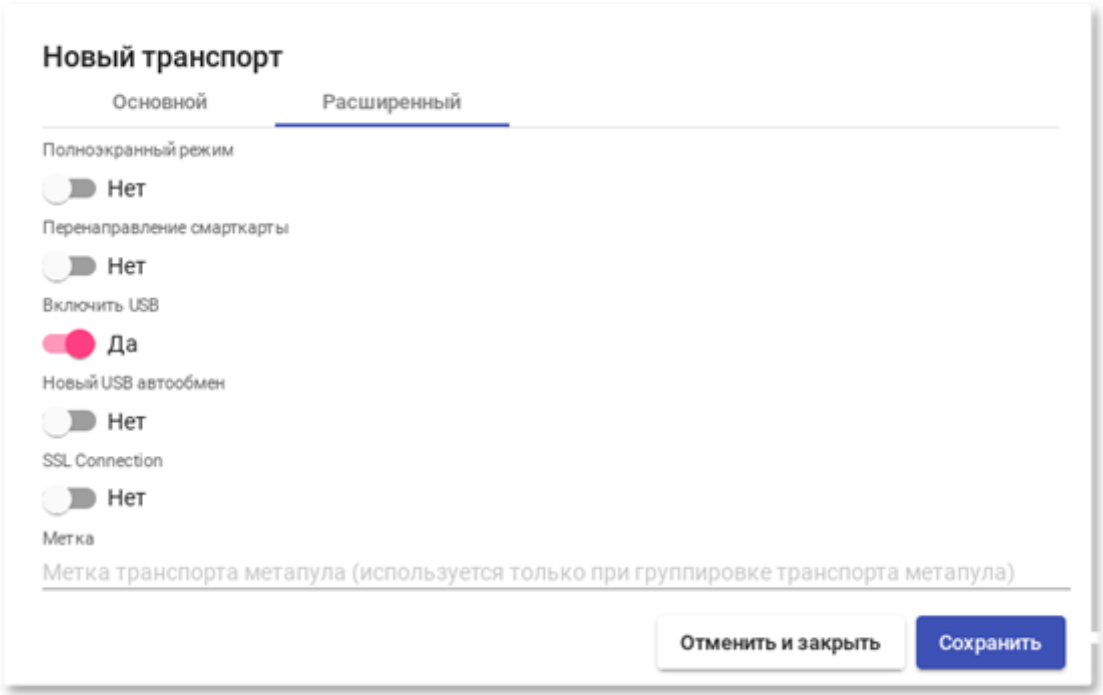

Рис. 115

7.3.4.6. HTML5 RDP (туннельный)

HTML5 RDP позволяет пользователям получать доступ к виртуальным рабочим столам Windows/Linux через протокол RDP с использованием веб-браузера, поддерживающего HTML5. На виртуальных рабочих столах должен быть установлен и включен протокол RDP (для виртуальных рабочих столов Linux необходимо использовать XRDP, для рабочих столов Windows необходимо настроить доступ HTML5 RDP).

Параметры конфигурации для настройки транспорта HTML5 RDP:

1) вкладка «Основной» (рис. 116):

- «Имя» – название транспорта;
- «Приоритет» – приоритет, чем меньше значение приоритета, тем выше данный транспорт будет указан в списке доступных транспортов для сервиса. Транспорт с самым низким приоритетом, будет транспортом по умолчанию;

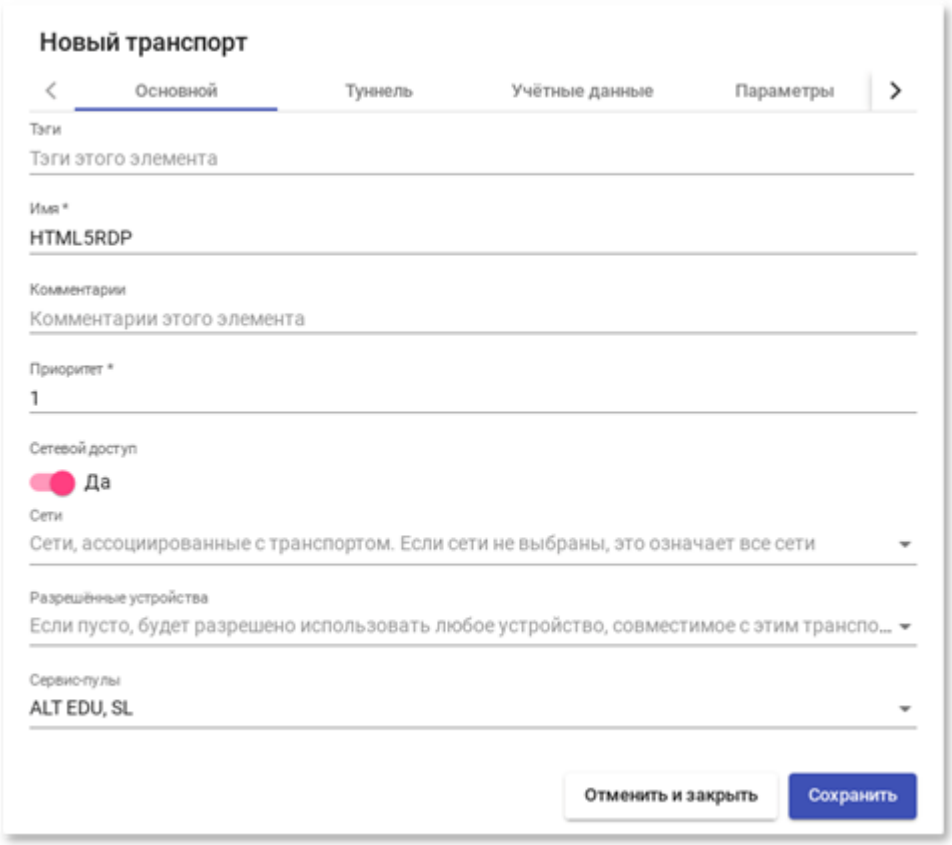

Рис. 116

- «Сетевой доступ» – разрешает или запрещает доступ пользователей к службе, в зависимости от сети из которой осуществляется доступ;
- «Сети» – сетевые диапазоны, подсети или IP-адреса (настраиваются в разделе «Сети»). Пустое поле означает «все сети». Используется вместе с параметром «Сетевой доступ»;
- «Разрешенные устройства» – разрешает доступ к службе только с выбранных устройств. Пустое поле означает «все устройства»;
- «Сервис-пулы» – позволяет назначить транспорт одному или нескольким ранее созданным пулам услуг. Можно оставить это поле пустым и выбрать способы подключения при создании пула услуг;

2) вкладка «Туннель» (рис. 117):

- «Туннельный сервер» – IP-адрес или имя OpenUDS Tunnel. Формат: `http(s)://IP_Tunneler:[Port]` (8080 – порт по умолчанию для http, 443 – для https);

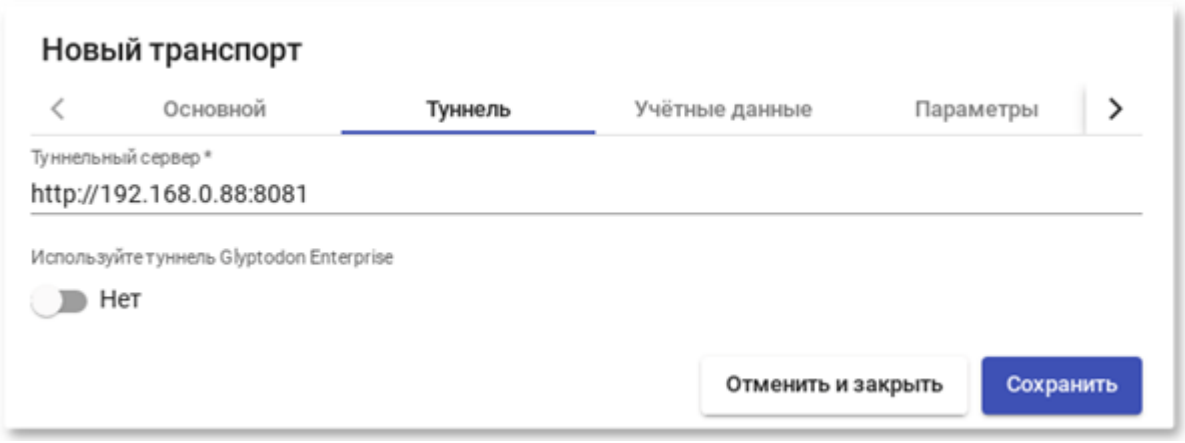

Рис. 117

3) вкладка «Учетные данные» (рис. 118):

- «Пропустить данные аккаунта» – если установлено значение «Да», учетные данные для доступа к виртуальному рабочему столу будут запрашиваться при подключении к серверу. Если установлено значение «Нет», будут использоваться данные OpenUDS;
- «Имя пользователя» – имя пользователя, которое будет использоваться для доступа к рабочему столу (пользователь должен существовать на

ВМ). Если данное поле пустое, будет использован логин авторизовавшегося в веб-интерфейсе OpenUDS пользователя;

- «Пароль» – пароль пользователя, указанного в поле «Имя пользователя»;
- «Без домена» – указывает, перенаправляется ли доменное имя вместе с пользователем. Значение «Да» равносильно пустому полю «Домен»;
- «Домен» – домен. Если поле не пустое, то учетные данные будут использоваться в виде DOMAIN\user;

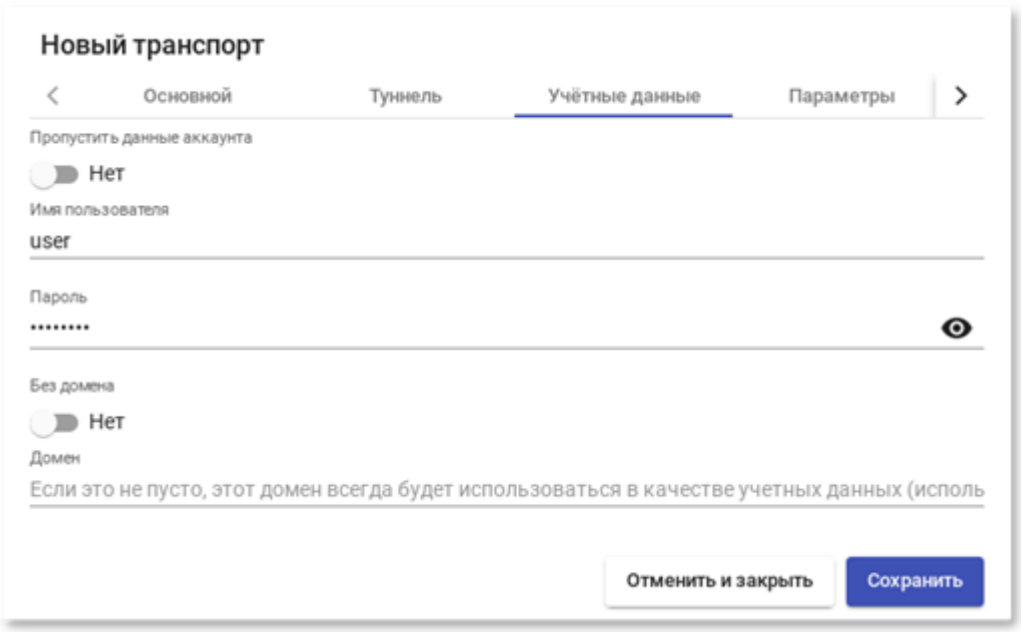

Рис. 118

4) вкладка «Параметры» (рис. 119):

- «Показать обои» – отображать обои рабочего стола;
- «Разрешить композицию рабочего стола» – включить «Desktop Composition»;
- «Сглаживание шрифтов» – активирует сглаживание шрифтов;
- «Включить аудио» – перенаправлять звук с рабочего стола на клиент подключения;
- «Включить микрофон» – включить микрофон на виртуальном рабочем столе;
- «Включить печать» – включить печать на виртуальном рабочем столе;

- «Обмен файлами» – политика обмена файлами между виртуальным рабочим столом и клиентом подключения. Позволяет создать временный каталог (расположенный на сервере OpenUDS Tunnel), для возможности обмена файлами между виртуальным рабочим столом и клиентом подключения;
- «Буфер обмена» – настройка общего буфера обмена;
- «Раскладка» – раскладка клавиатуры, которая будет включена на рабочем столе;

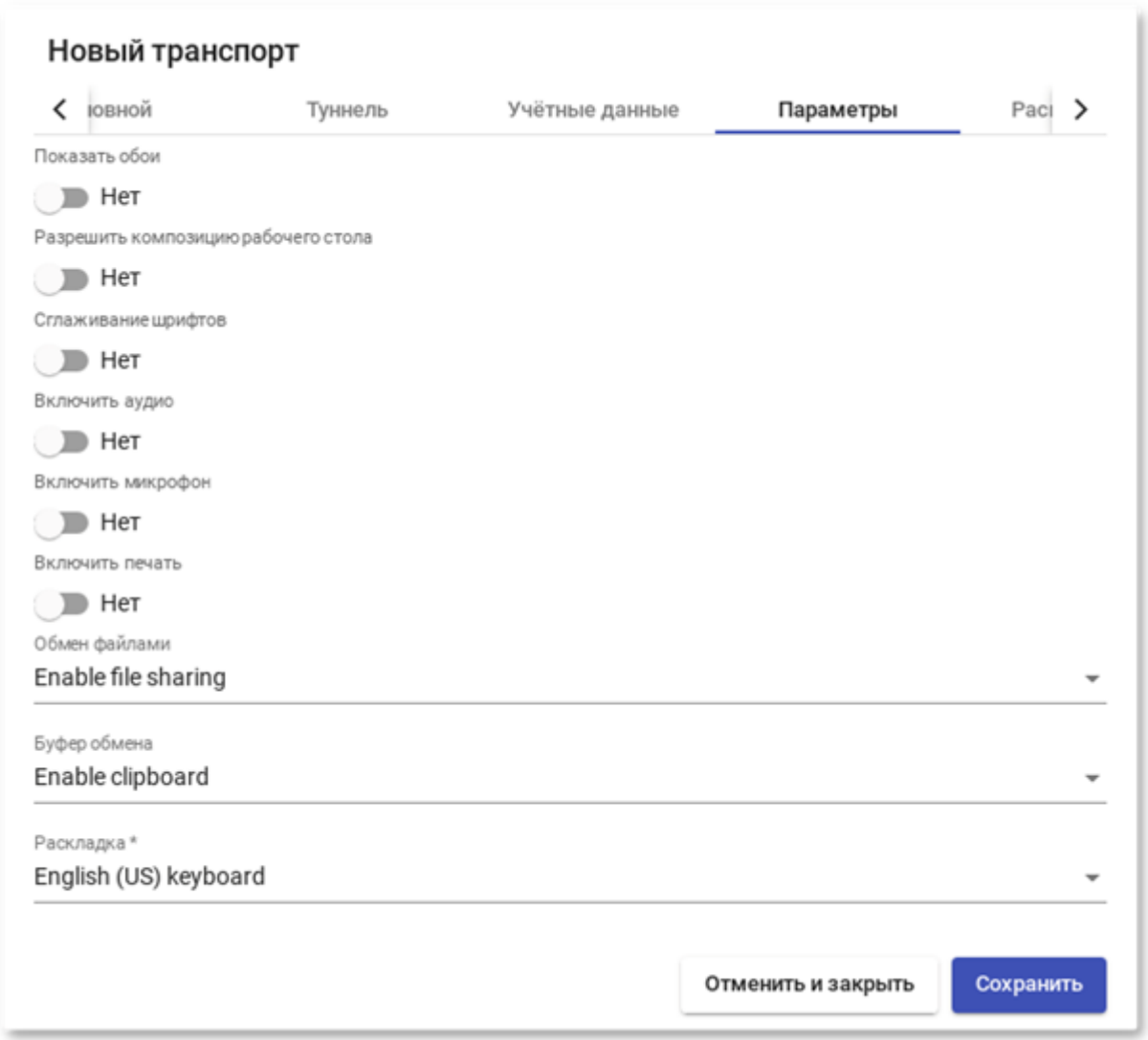

Рис. 119

5) вкладка «Расширенный» (рис. 120):

- «Срок действия билета» – допустимое время (в секундах) для клиента HTML5 для перезагрузки данных из OpenUDS Broker (рекомендуется использовать значение по умолчанию» – 60);
- «Открывать HTML в новом окне» – позволяет указать открывать ли подключение в новом окне;
- «Безопасность» – позволяет задать уровень безопасности соединения;
- «Порт RDP» – порт RDP (по умолчанию» – 3389);
- «Метка» – метка транспорта метапула (используется для того чтобы назначить несколько транспортов метапулу).

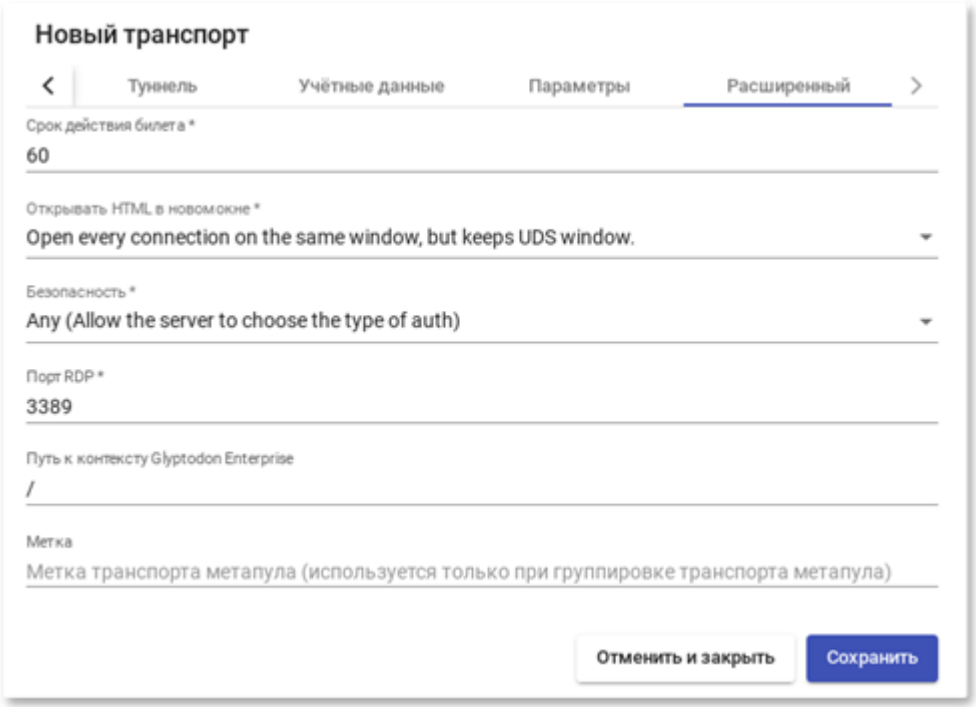

Рис. 120

7.3.4.7. HTML5 SSH (туннельный)

HTML5 SSH позволяет пользователям получать доступ к виртуальным рабочим столам Linux по протоколу SSH с использованием веб-браузера, поддерживающего HTML5 (на машинах должен быть запущен сервер SSH). Используя данный транспорт можно подключаться к серверам Linux, на которых не установлен оконный менеджер или среда рабочего стола.

Параметры для настройки транспорта HTML5 SSH:

1) вкладка «Основной» (рис. 121):

- «Имя» – название транспорта;
- «Приоритет» – приоритет, чем меньше значение приоритета, тем выше данный транспорт будет указан в списке доступных транспортов для сервиса. Транспорт с самым низким приоритетом, будет транспортом по умолчанию;
- «Сетевой доступ» – разрешает или запрещает доступ пользователей к службе в зависимости от сети, из которой осуществляется доступ;
- «Сети» – сетевые диапазоны, подсети или IP-адреса (настраиваются в разделе «Сети»). Пустое поле означает «все сети». Используется вместе с параметром «Сетевой доступ»;

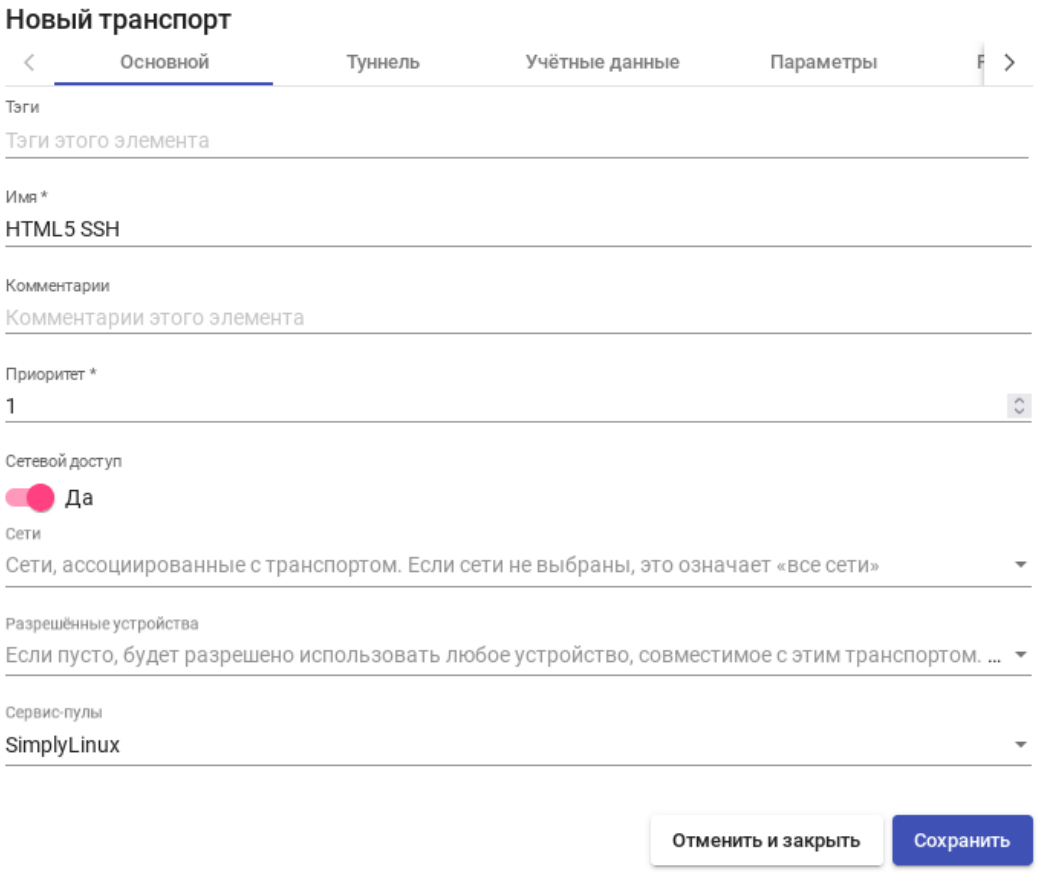

Рис. 121 – Настройка HTML5 SSH. Вкладка «Основной»

- «Разрешенные устройства» – разрешает доступ к службе только с выбранных устройств. Пустое поле означает «все устройства»;

- «Сервис-пулы» – позволяет назначить транспорт одному или нескольким ранее созданным пулам услуг. Можно оставить это поле пустым и выбрать способы подключения при создании пула услуг;

2) вкладка «Туннель» (рис. 122):
   - «Туннельный сервер» – IP-адрес или имя OpenUDS Tunnel. Формат: `http(s)://IP_Tunneler:[Port]` (8080 – порт по умолчанию для http, 443 – для https);
3) вкладка «Учетные данные» (рис. 123):
   - «Имя пользователя» – имя пользователя, которое будет использоваться для доступа к рабочему столу (пользователь должен существовать на ВМ). Если данное поле пустое, будет использован логин авторизовавшегося в веб-интерфейсе OpenUDS пользователя;
4) вкладка «Параметры» (рис. 124):
   - «SSH-команда» – команда, которая будет выполнена на удаленном сервере. Если команда не указана, будет запущена интерактивная оболочка (рис. 125);
   - «Обмен файлами» – политика обмена файлами между виртуальным рабочим столом и клиентом подключения;
   - «Корень общего доступа к файлам» – корневой каталог для доступа к файлам. Если не указан, будет использоваться корневой каталог (/);
   - «Порт SSH-сервера» – порт SSH-сервера (по умолчанию – 22);
   - «Ключ хоста SSH» – ключ хоста SSH. Если ключ не указан, проверка подлинности хоста выполняться не будет;
   - «Поддержка сервера в рабочем состоянии» – время (в секундах) между сообщениями проверки активности, отправляемых на сервер. Если не указано, сообщения проверки активности не отправляются;

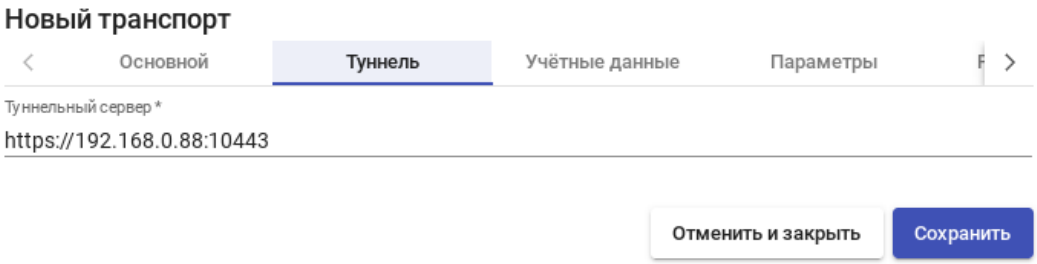

Рис. 122 – Настройка HTML5 SSH. Вкладка «Туннель»

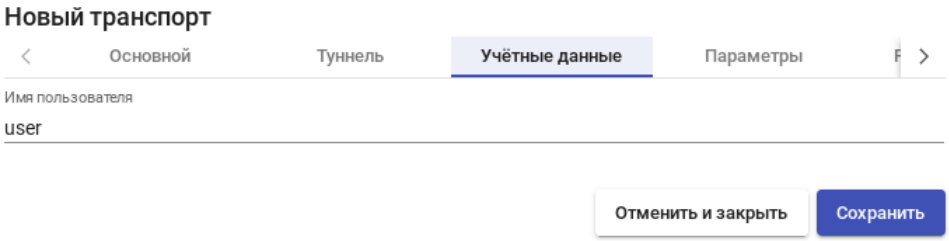

Рис. 123 – Настройка HTML5 SSH. Вкладка «Учетные данные»

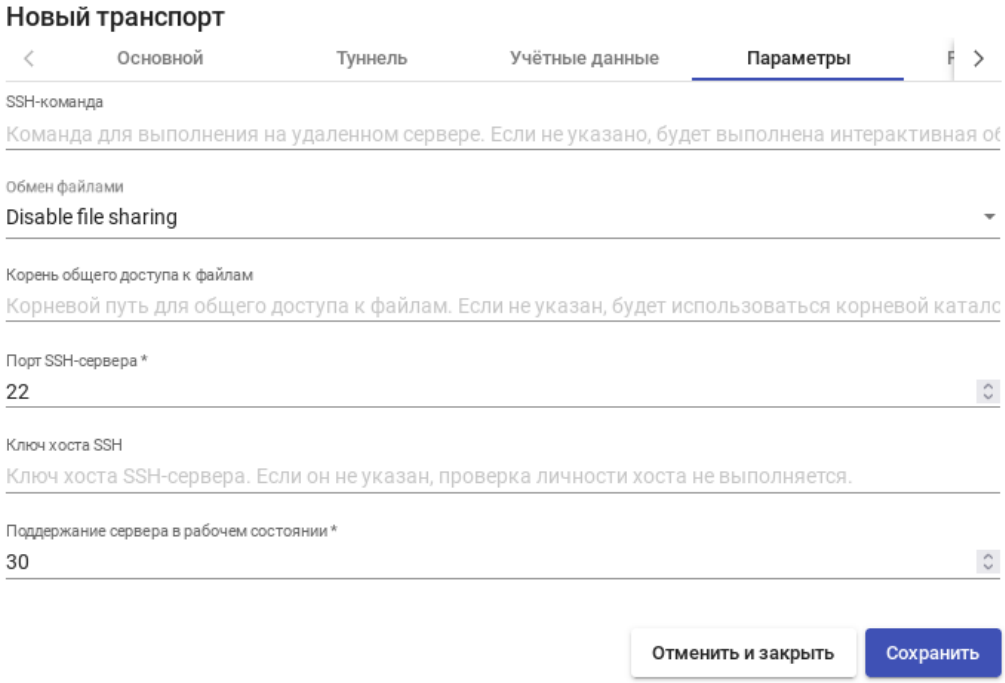

Рис. 124 – Настройка HTML5 SSH. Вкладка «Параметры»

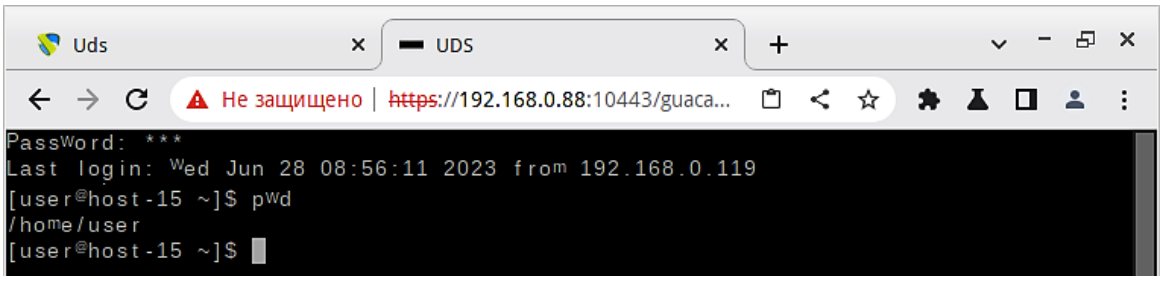

Рис. 125 – OpenUDS. Подключение по HTML5 SSH

5) вкладка «Расширенный» (рис. 126):

- «Срок действия билета» – допустимое время (в секундах) для клиента HTML5 для перезагрузки данных из OpenUDS Broker (рекомендуется использовать значение по умолчанию – 60);
- «Открывать HTML в новом окне» – позволяет указать открывать ли подключение в новом окне;
- «Метка» – метка транспорта метапула (используется для того чтобы назначить несколько транспортов метапулу).

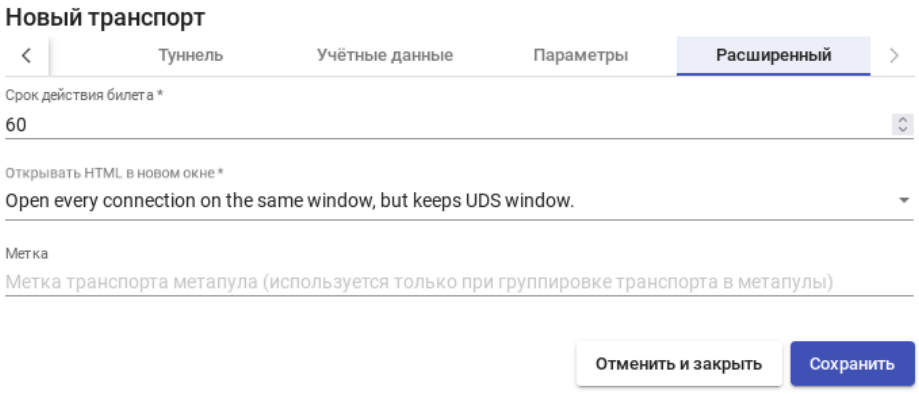

Рис. 126 – Настройка HTML5 SSH. Вкладка «Расширенный»

После входа на удаленный сервер, в зависимости от настроек политики обмена файлами, можно скачивать/загружать файлы. Для загрузки файлов можно открыть окно настроек (<Ctrl>+<Shift>+<Alt>), выбрать устройство в поле «Устройства», нажать на кнопку «Загрузка файлов» и выбрать файл. Ход передачи файла будет показан в левом нижнем углу окна (рис. 127).

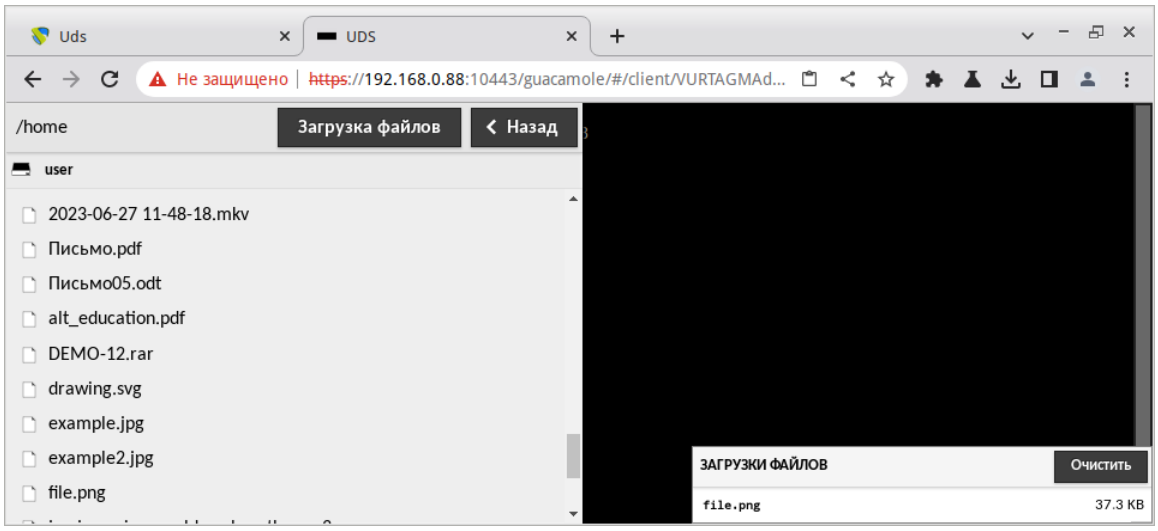

Рис. 127 – HTML5 SSH. Передача файлов

7.3.5. Сети

В OpenUDS можно зарегистрировать различные сети для управления доступом клиентов к виртуальным рабочим столам или приложениям (при доступе к OpenUDS определяется IP-адрес клиента подключения). Эти сети совместно с «Транспортом» будут определять, какой тип доступа будет доступен пользователи для подключения к виртуальным рабочим столам.

Чтобы добавить сеть, следует в разделе «Подключение» выбрать пункт «Сети» и нажать на кнопку «Новый» (рис. 128).

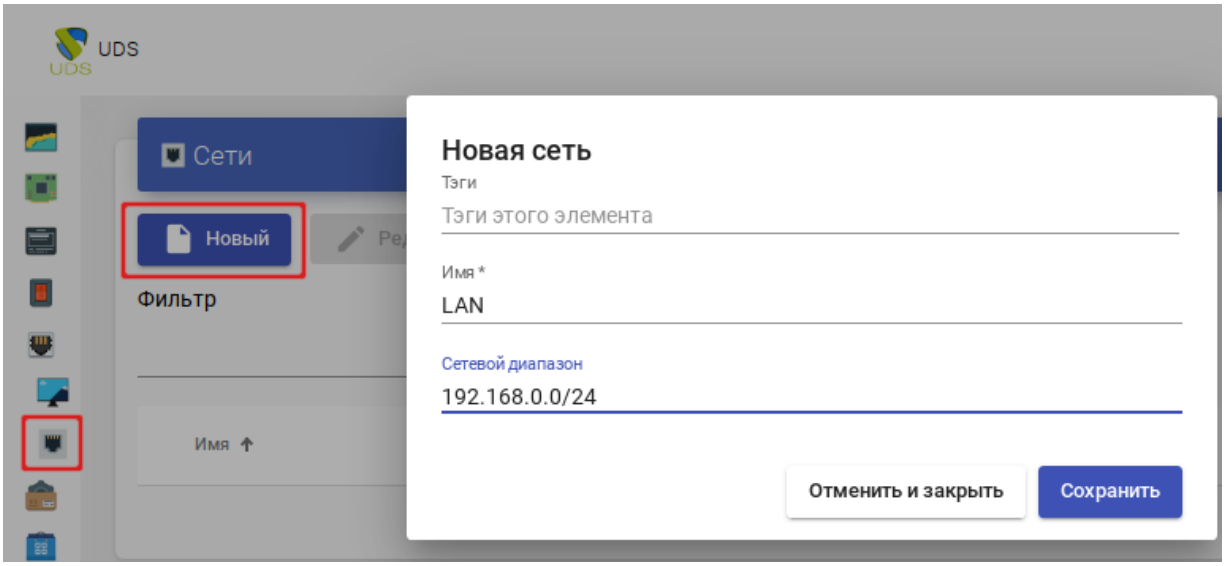

Рис. 128

В открывшемся окне следует указать название сети и сетевой диапазон. В качестве сетевого диапазона можно указать:

- одиночный IP-адрес: xxx.xxx.xxx.xxx (например, 192.168.0.33);
- подсеть: xxx.xxx.xxx.xxx/x (например, 192.168.0.0/24);
- диапазон IP-адресов: xxx.xxx.xxx.xxx-xxx.xxx.xxx.xxx (например, 192.168.0.1-192.168.0.50).

После создания сетей, появится возможность указать их при создании/редактировании транспорта. Можно настроить будет ли данный транспорт отображаться у клиента, в зависимости от сети, в которой находится клиент (рис. 129).

В данном примере транспорт «X2Go-xfce» будет доступен только клиентам из сети 192.168.0.0/24.

Если сети для транспорта не определены, доступ к службам рабочего стола и виртуальным приложениям будет возможен из любой сети.

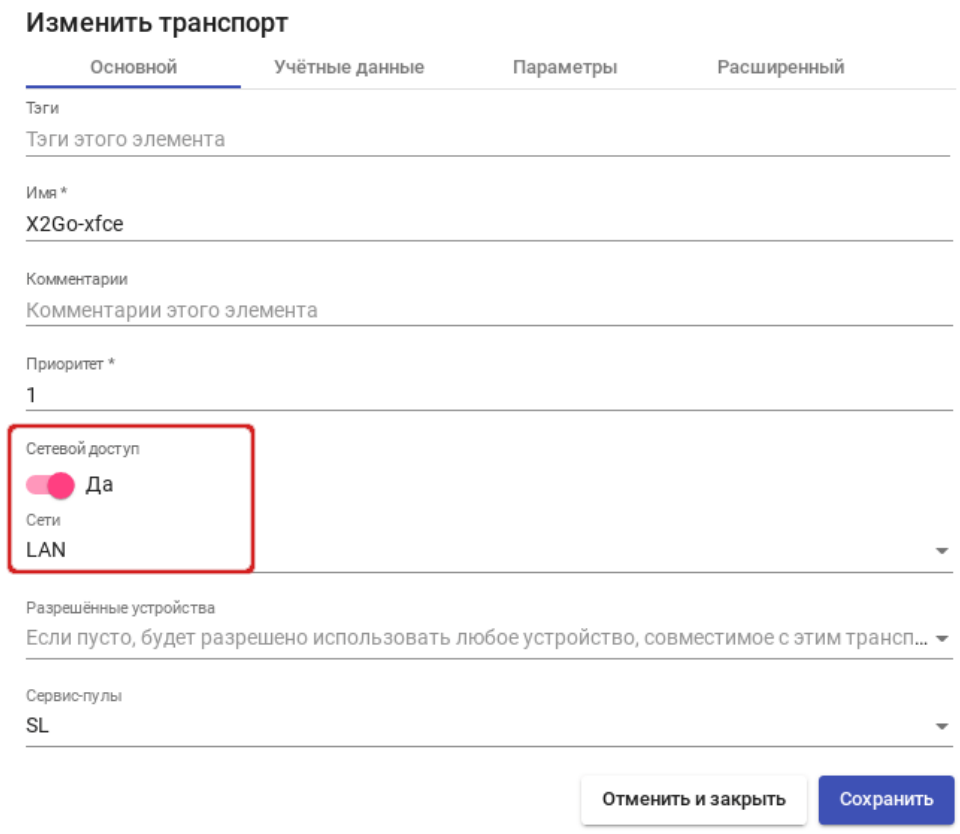

Рис. 129

7.3.6. Пулы услуг

После того, как был создан и настроен хотя бы один поставщик услуг с соответствующей службой/услугой, аутентификатор (с пользователем и группой), менеджер ОС и транспорт, можно создать пул услуг (Сервис-пул) для публикации виртуальных рабочих столов.

Для создания пула услуг необходимо в разделе «Сервис-пулы» нажать на кнопку «Новый» (рис. 130).

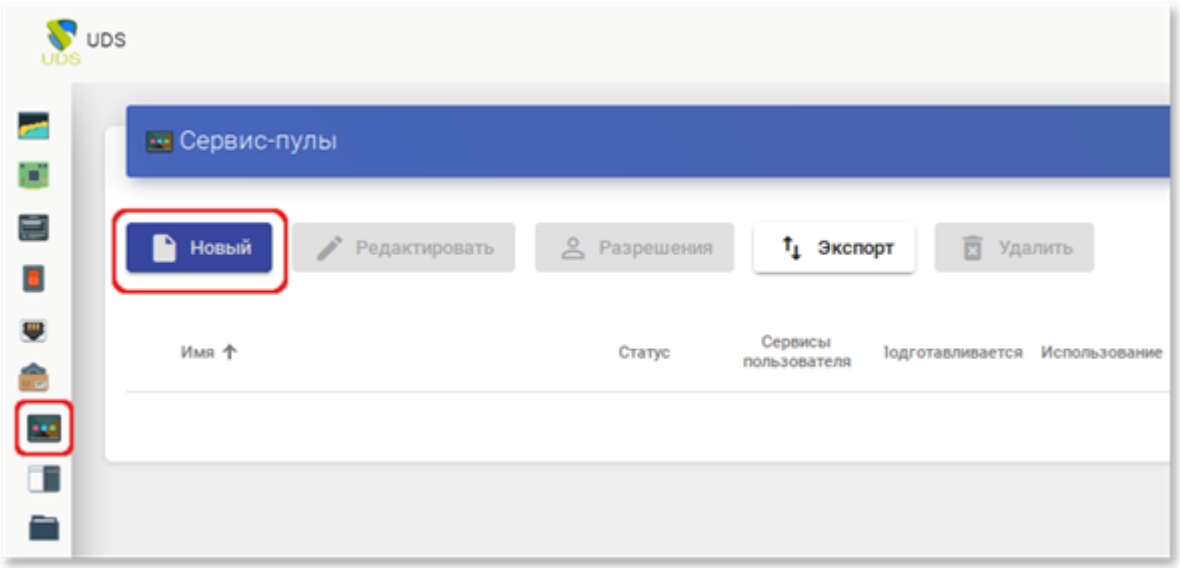

Рис. 130

Заполнить параметры конфигурации:

1) вкладка «Основной» (рис. 131):

- «Имя» – название службы (это имя будет показано пользователю для доступа к рабочему столу или виртуальному приложению). В этом поле можно использовать переменные для отображения информации об услугах:
- {use} – указывает процент использования пула (рассчитывается на основе поля «Максимальное количество предоставляемых сервисов» и назначенных услуг);
- {total} – общее количество машин (данные извлечены из поля «Максимальное количество предоставляемых сервисов»);
- {usec} – количество машин, используемых пользователями в пуле;

- {left} – количество машин, доступных в пуле для подключения пользователей;
- «Базовый сервис» – служба, созданная ранее в поставщике услуг (состоит из поставщика услуг и базовой услуги);
- «ОС Менеджер» – ранее созданный менеджер ОС, конфигурация которого будет применяться к каждому из созданных виртуальных рабочих столов или приложений. Если выбрана услуга типа «Статический IP», это поле не используется;
- «Публиковать при создании» – если этот параметр включен, при сохранении пула услуг система автоматически запустит первую публикацию. Если установлено значение «Нет», будет необходимо запустить публикацию сервиса вручную (из вкладки «Публикации»);

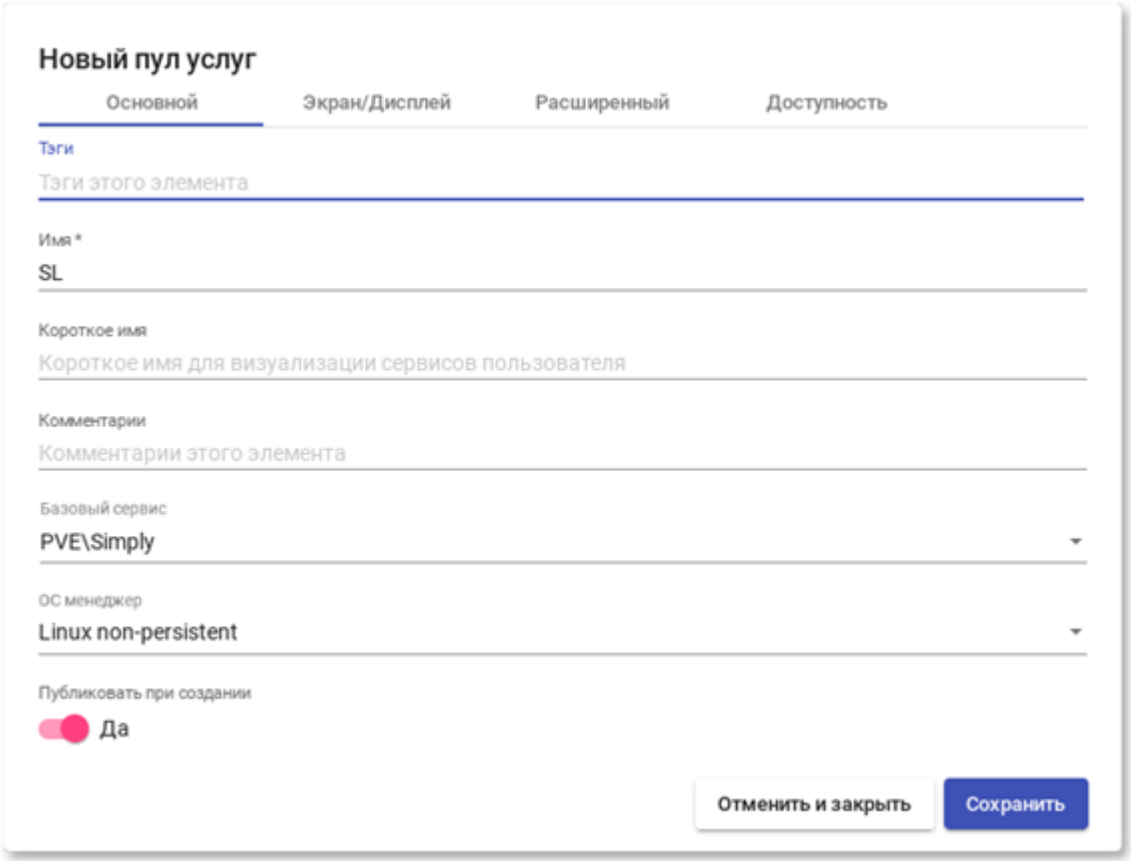

Рис. 131

2) вкладка «Экран/Дисплей» (рис. 132):

- «Видимый» – если этот параметр отключен, пул не будет отображаться у пользователей;
- «Привязанный образ» – изображение, связанное с услугой. Изображение должно быть предварительно добавлено в репозиторий изображений (раздел «Инструменты» → «Галерея»);
- «Пул-группа» – позволяет группировать различные службы. Группа должна быть предварительно создана в разделе «Пулы» → «Группа»;
- «Доступ к календарю запрещен» – позволяет указать сообщение, которое будет показано пользователю, если доступ к сервису ограничен правилами календаря;

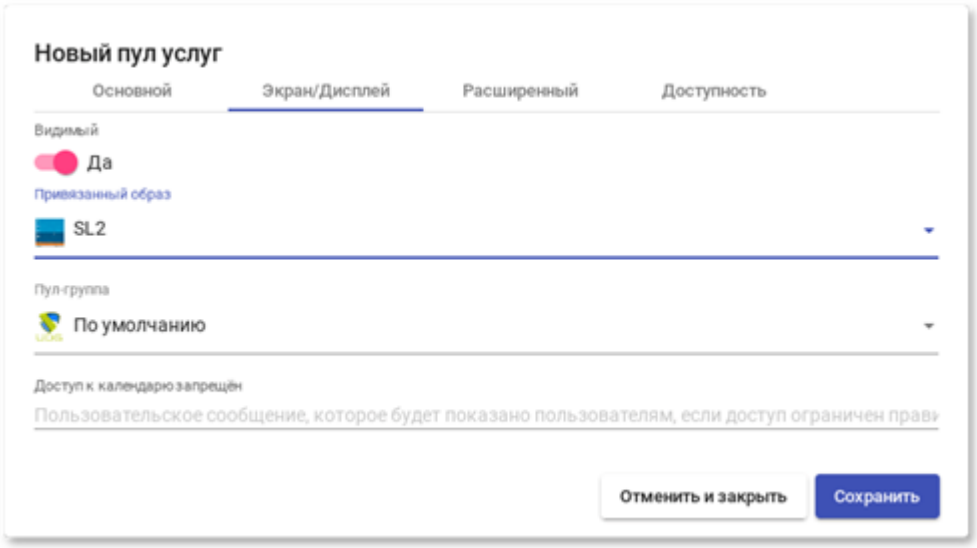

Рис. 132

3) вкладка «Расширенный» (рис. 133):

- «Разрешить удаление пользователями» – если этот параметр включен, пользователи могут удалять назначенные им службы. Если сервис представляет собой виртуальный рабочий стол, автоматически сгенерированный OpenUDS, он будет удален, и при следующем подключении ему будет назначен новый. Если это другой тип сервиса (vAPP/статический IP), будет удалено только назначение, а новое будет назначено на следующее подключение;

- «Разрешить сброс пользователям» – если этот параметр включен, пользователь сможет перезапускать или сбрасывать назначенные ему службы (относится только к виртуальным рабочим столам, автоматически созданным OpenUDS);
- «Игнорирует неиспользуемые» – если этот параметр включен, непостоянные пользовательские службы, которые не используются, не будут удаляться;
- «Показать транспорты» – если этот параметр включен, будут отображаться все транспорты, назначенные услуге. Если параметр не активирован, будет отображаться только транспорт по умолчанию (с наивысшим приоритетом);
- «Учетные записи» – назначение услуги ранее созданным «Аккаунтам» («Пулы» → «Аккаунты»);

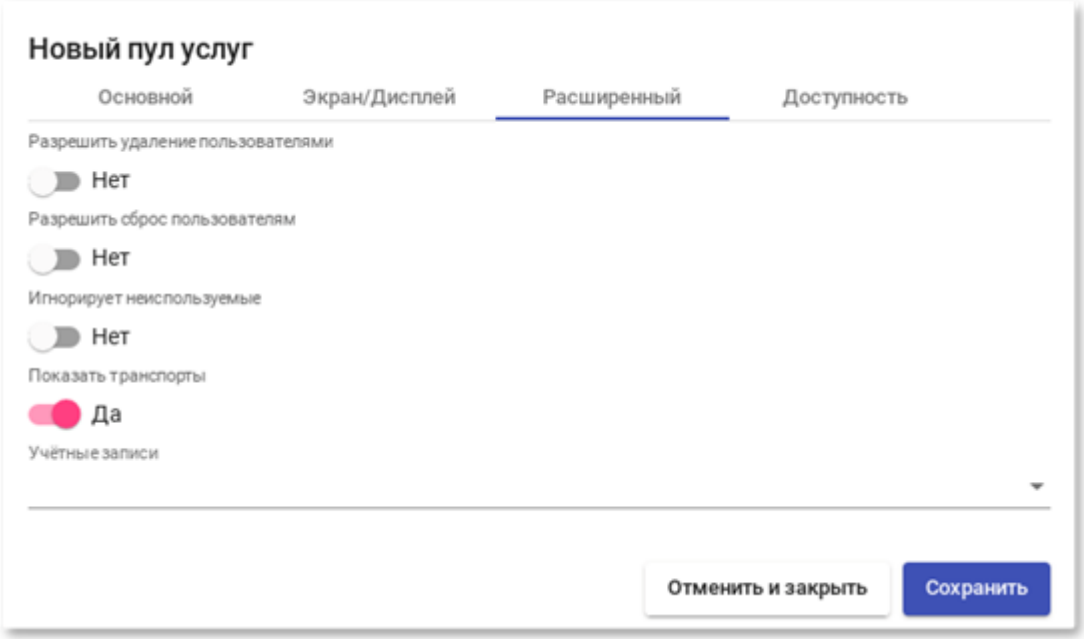

Рис. 133

4) вкладка «Доступность» (рис. 134):

- «Первоначально доступные сервисы» – минимальное количество виртуальных рабочих столов, созданных, настроенных и назначенных/доступных для службы;

- «Сервисы для удержания в кэше» – количество доступных виртуальных рабочих мест. Эти ВМ всегда будут настроены и готовы к назначению пользователю (они будут автоматически создаваться до тех пор, пока не будет достигнуто максимальное количество машин, указанное в поле «Максимальное количество предоставляемых сервисов»);
- «Сервисы, хранящиеся в L2 кэше» – количество виртуальных рабочих столов в спящем или выключенном состоянии. Виртуальные рабочие столы, сгенерированные на уровне кэша L2, будут помещены в кэш, как только система потребует их (они никогда не будут напрямую назначены пользователям);
- «Максимальное количество предоставляемых сервисов» – максимальное количество виртуальных рабочих столов, созданных системой в данном пуле (рабочие столы, созданные в кэше L2, не учитываются).

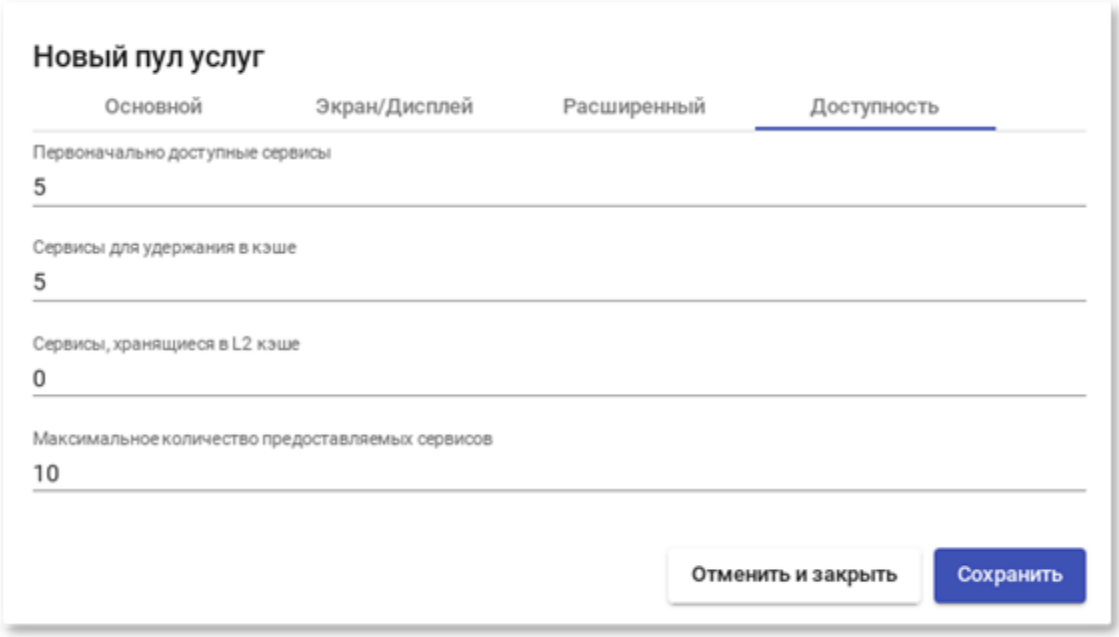

Рис. 134

После нажатия кнопки «Сохранить» система начнет создавать виртуальные рабочие столы на основе настроенного кэша.

После создания пула, в настройках (дважды щелкнуть мышью по строке созданного пула или в контекстном меню пула выбрать пункт «Подробность») необходимо:

- на вкладке «Группы» назначить группы доступа (выбрать аутентификатор и группу, которая будет иметь доступ к этому пулу служб) (рис. 135);

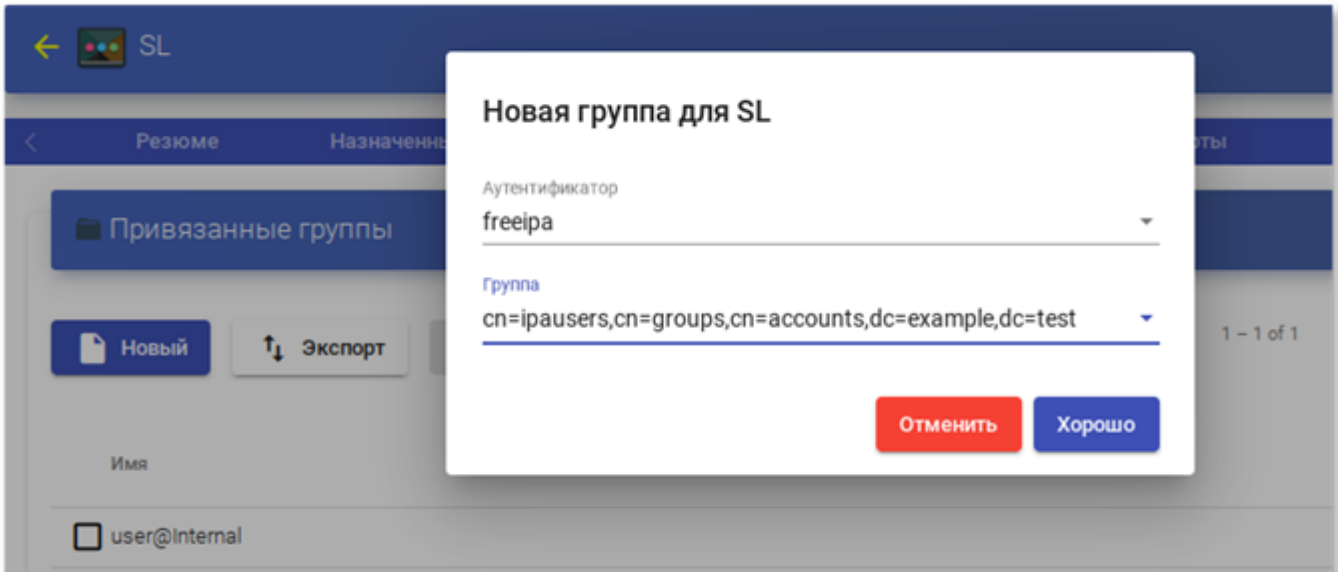

Рис. 135

- на вкладке «Транспорты» выбрать способы подключения пользователей к рабочему столу (рис. 136);

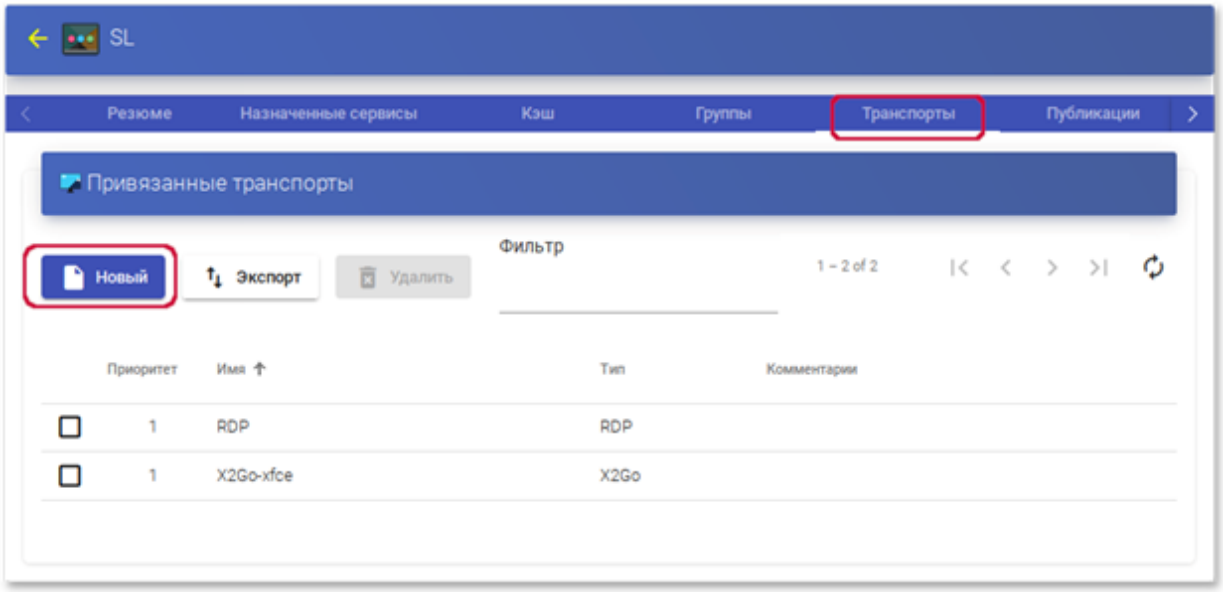

Рис. 136

7.3.7. «Мета-пулы»

Виртуальные рабочие столы можно сгруппировать в пулы рабочих столов («Мета-пулы»), что упрощает управление и организацию. Создание «Мета-пула» позволит получить доступ к виртуальным рабочим стола или приложениям из разных «Service Pools». Эти пулы будут работать вместе, предоставляя различные услуги абсолютно прозрачным для пользователей способом.

«Пулы услуг», образующие «Мета-пул», будут работать в соответствии с политикой, которая позволит предоставлять услуги в соответствии с потребностями пула. Чтобы создать «Мета-пул», следует в разделе «Пулы» выбрать пункт «Мета-пул» и нажать на кнопку «Новый» (рис. 137).

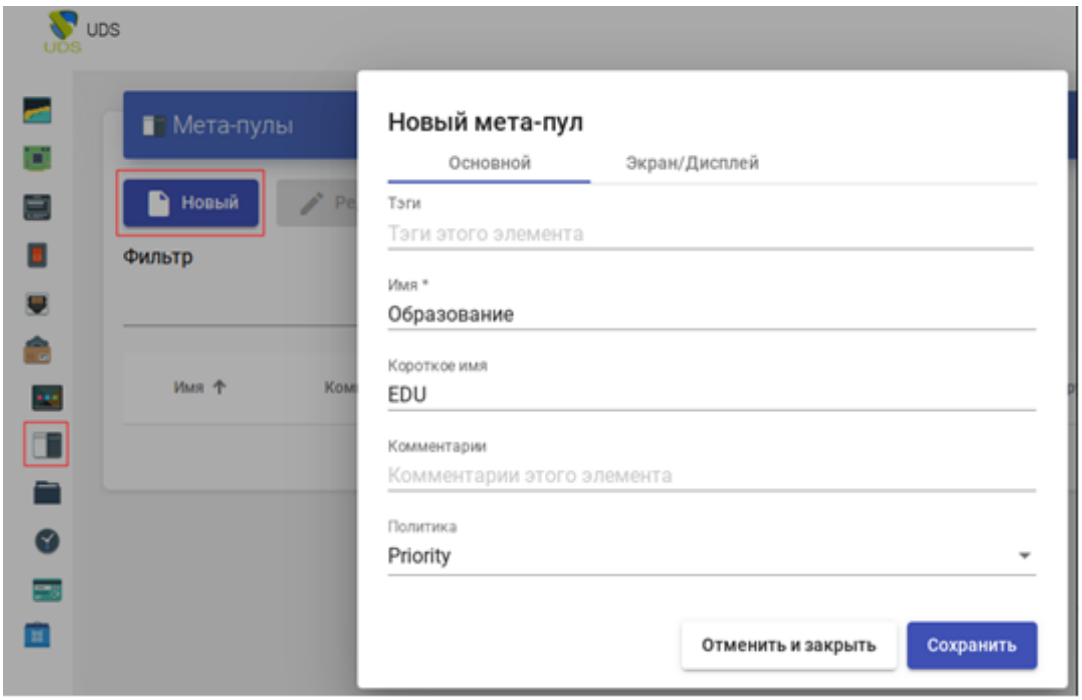

Рис. 137

Для настройки «Мета-пула» необходимо указать:

1) вкладка «Основной»:
   - «Имя» – название «Мета-пула» (это будет видеть пользователь для доступа к службе);
   - «Короткое имя» – если указано, то это будет видеть пользователь для доступа к службе (при наведении на него указателя появится содержимое поля «Имя»);

- «Политика» – политика, которая будет применяться при создании сервисов в «Пулах услуг», являющихся частью «Мета-пула»:
    а) «Eventy distributed» – услуги будут создаваться и использоваться равномерно во всех «пулах услуг», составляющих «Мета-пул»;
    б) «Priority» – услуги будут создаваться и использоваться из «пула услуг» с наибольшим приоритетом (приоритет определяется полем «priority», чем ниже значение этого поля, тем выше приоритет у элемента). Когда будет достигнуто максимального количество сервисов данного «пула услуг», будут использоваться сервисы следующего;
    в) «Greater % available» – службы будут создаваться и использоваться из «пула услуг», который имеет самый высокий процент свободных услуг;

2) вкладка «Экран/Дисплей» (рис. 138):
- «Привязанный образ» – изображение, связанное с «Мета-пулом». Изображение должно быть предварительно добавлено в репозиторий изображений (раздел «Инструменты» → «Галерея»);
- «Пул-группа» – позволяет группировать различные «Мета-пулы». Группа должна быть предварительно создана в разделе «Пулы» → «Группы»;
- «Видимый» – если этот параметр отключен, «Мета-пул» не будет отображаться у пользователей;
- «Доступ к календарю запрещен» – текст, который будет отображаться, когда доступ к «Мета-пулу» запрещен приложением календаря доступа;
- «Выбор транспорта» – указывает как на «Мета-пул» будет назначен транспорт:
    а) «Automatic selection» – будет доступен в транспорт с самым низким приоритетом, назначенным «пулу услуг». Выбор транспорта не допускается;

б) «Use only common transports» – транспорт, который является общими для всего «пула услуг», будет доступен в «Мета-пуле»;

в) «Group Transports by label» – транспорт, которым назначены «метки», будут доступны в «метапуле» (это поле находится внутри каждого «Транспорта» на вкладке «Дополнительно»).

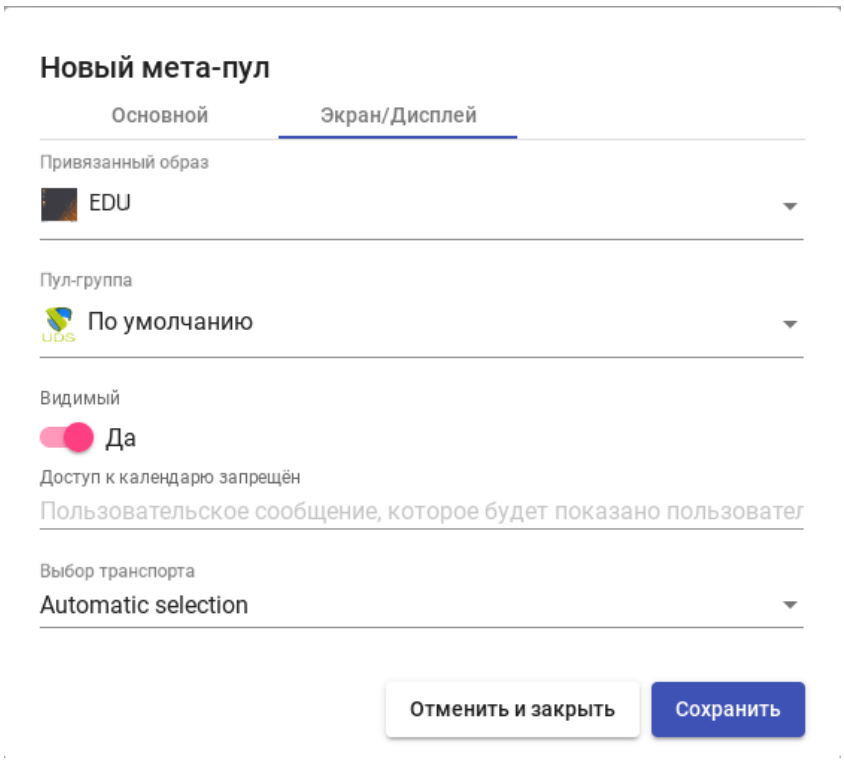

Рис. 138

Сохранив конфигурацию «Мета-пула», можно начать регистрацию «Пулов услуг». Для этого следует дважды щелкнуть мышью по строке созданного «Мета-пула» или в контекстном меню «Мета-пула» выбрать пункт «Подробность» (рис. 139).

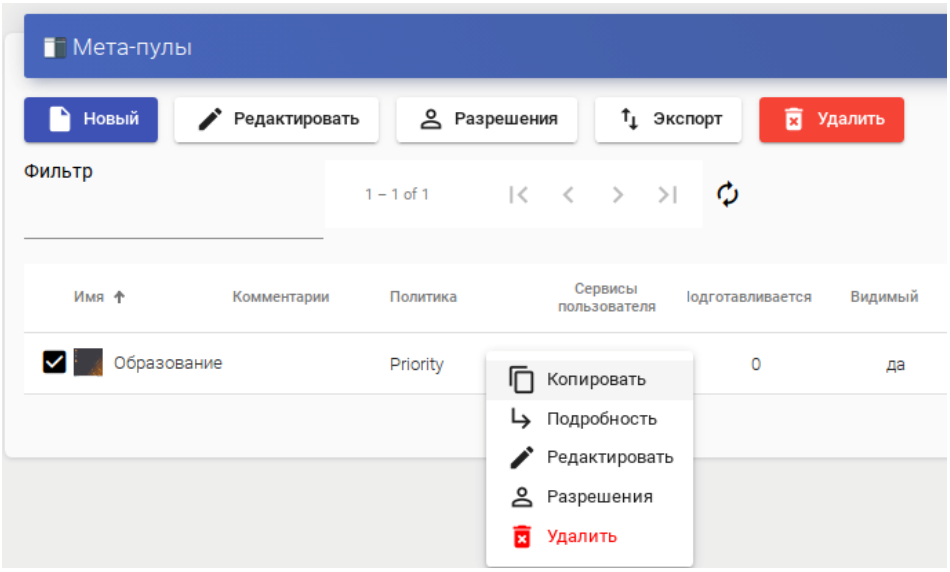

Рис. 139

Чтобы добавить «Пул услуг» в «Мета-пул», следует нажать на кнопку «Новый» (рис. 140).

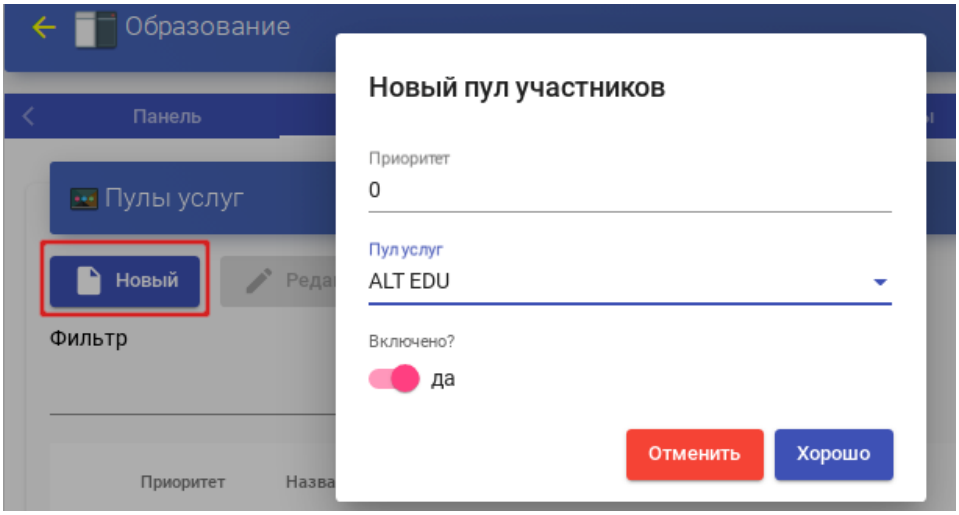

Рис. 140

Для добавления «Пула услуг» необходимо указать:

- «Приоритет» – приоритет, который будет иметь «Пул услуг» в «Мета-пуле» (чем ниже значение, тем больше приоритет);
- «Пул услуг» – «Пул услуг», который будет добавлен в «Мета-пул» («Пул услуг» должен быть предварительно создан);
- «Включено» – включает или отключает видимость «Пула услуг» в «Мета-пуле».

Можно добавить столько «Пулов услуг», сколько нужно, комбинируя службы, размещенные на разных платформах виртуализации (PVE, KVM, OpenNebula и т. д.), серверах приложений и статических устройствах (рис. 141).

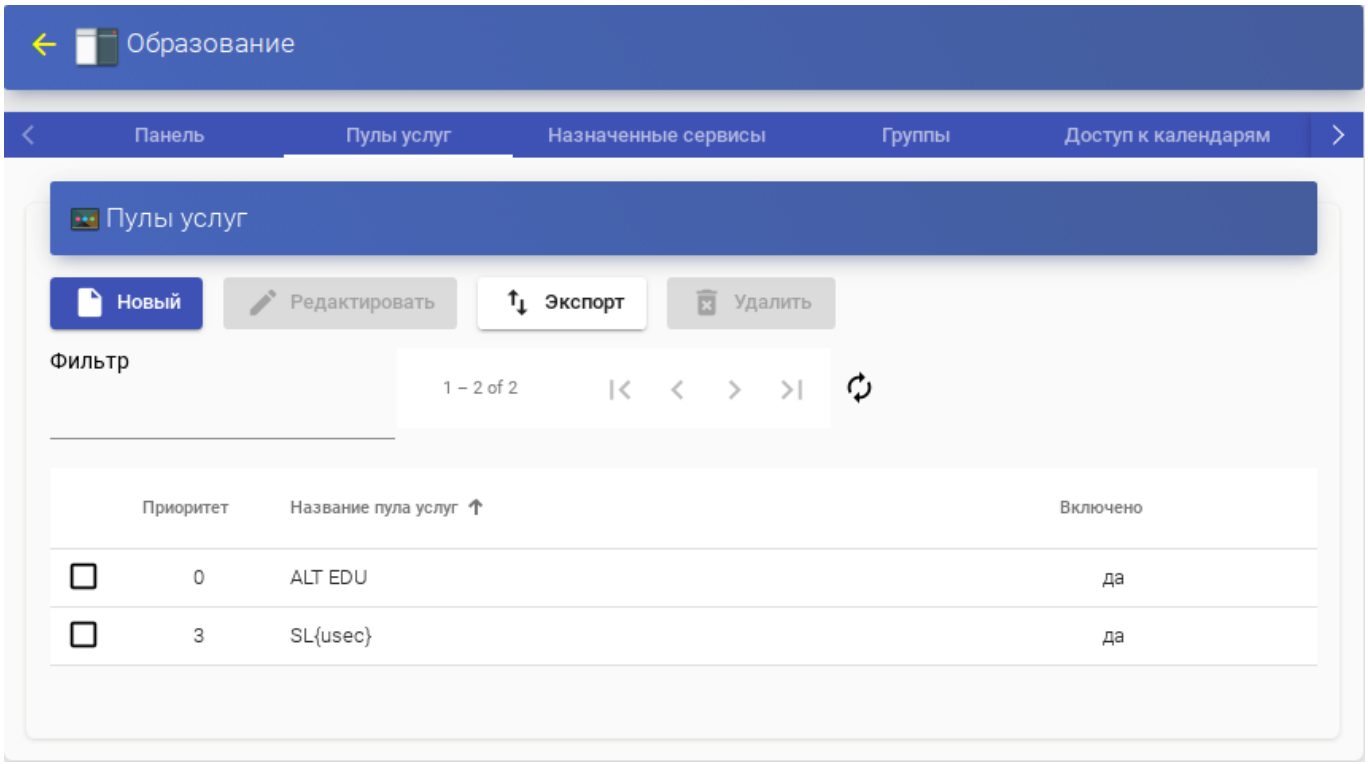

Рис. 141

Как и при создании «Пула услуг», здесь есть следующие вкладки с информацией и конфигурацией:

- «Назначенные сервисы» – показывает службы, назначенные пользователям (можно вручную удалить назначение и переназначить другому пользователю);
- «Группы» – указывает, какие группы пользователей будут иметь доступ к услуге;
- «Доступ к календарям» – позволяет применить ранее созданный календарь доступа;
- «Журналы» – журналы «Мета-пула».

7.3.8. Управление доступом по календарю

В OpenUDS можно настроить ограничение доступа пользователей к удаленным рабочим столам и виртуальным приложениям по дате и времени.

С помощью календаря также можно автоматизировать определенные задачи в «Пуле услуг», такие, как создание новых публикаций, настройка значений системного кэша, добавление/удаление групп и транспорта, изменение максимального количества услуг.

Для создания календаря необходимо в разделе «Календари» нажать на кнопку «Новый». В открывшемся окне ввести описательное название в поле «Имя» и нажать на кнопку «Сохранить» (рис. 142).

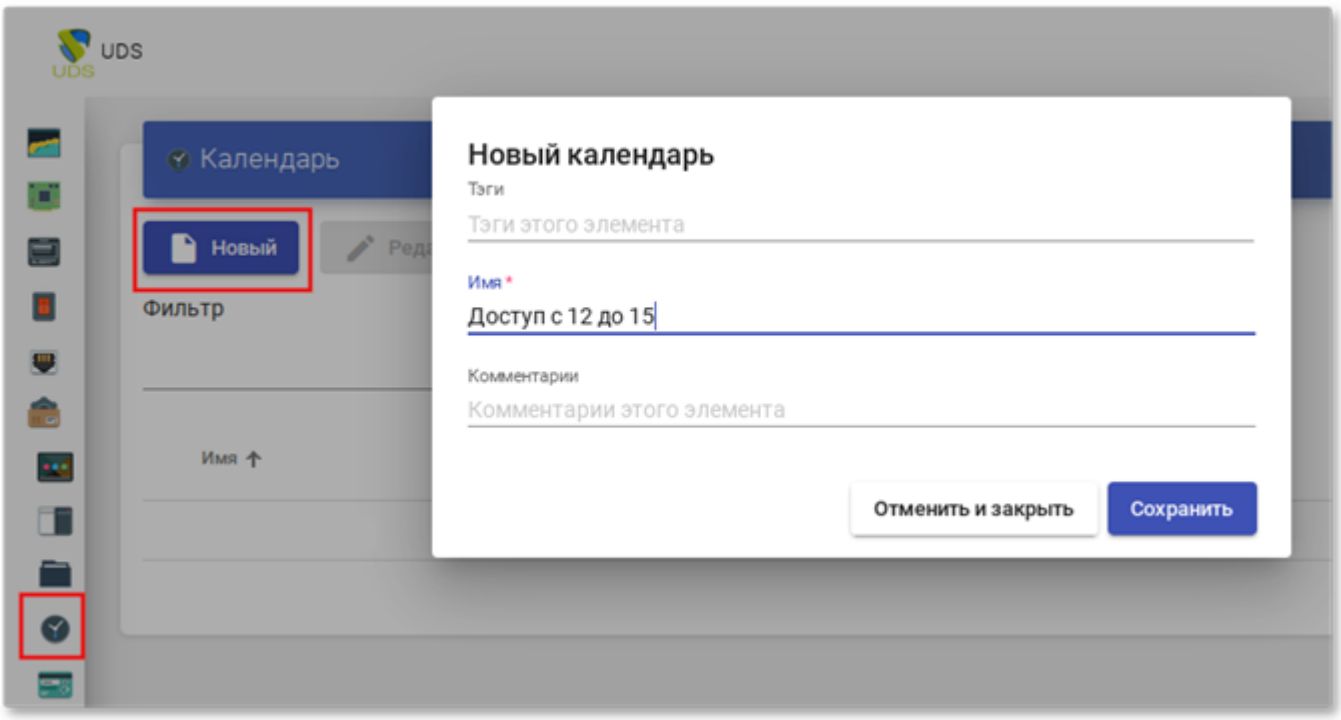

Рис. 142

В «Календаре» можно зарегистрировать правила, чтобы запланировать доступность услуги в определенное время. Для создания правила следует выбрать календарь (дважды щелкнуть мышью по строке созданного календаря или в контекстном меню календаря выбрать пункт «Подробность») и нажать на кнопку «Новый» (рис. 143).

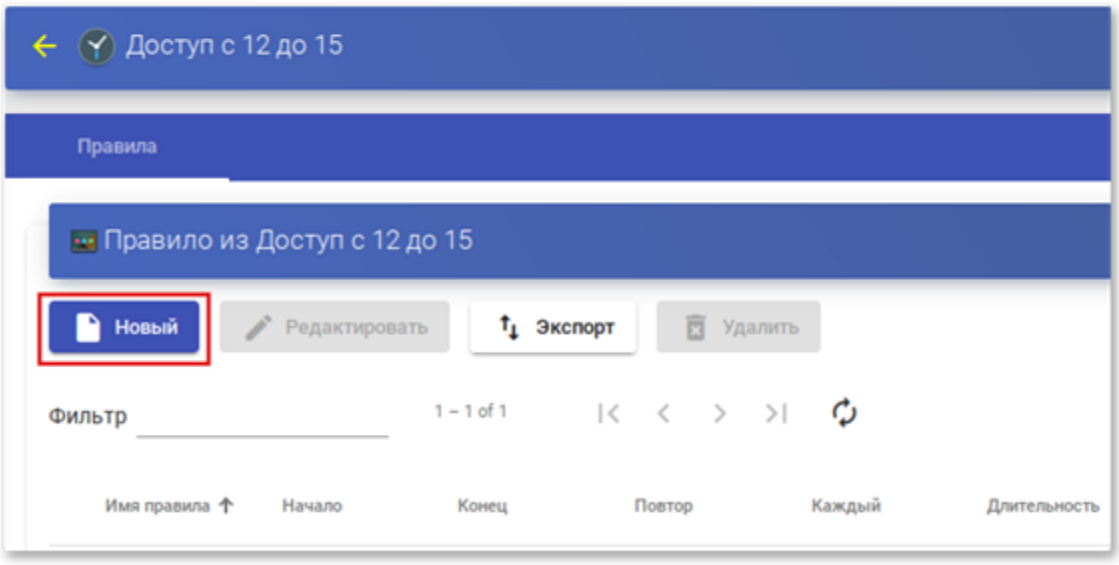

Рис. 143

Минимальные параметры для настройки правила (рис. 144):

- «Имя» – название правила;
- «Событие» – настройка времени выполнения. Необходимо указать время начала и продолжительность события (в минутах/часах/днях/неделях);

Рис. 144

- «Repetition» (Периодичность) – настройка периодичности выполнения. Необходимо указать дату начала, частоту повторения правила (ежедневно/еженедельно/ежемесячно/ежегодно/по будням) и интервал повторения (в днях);
- «Панель» – показывает сводные данные (резюме) всех ранее указанных настроек.

После нажатия кнопки «Хорошо» будет создано правило, которое будет назначено «Пулу услуг» (виртуальному рабочему столу и/или приложению) (рис. 145).

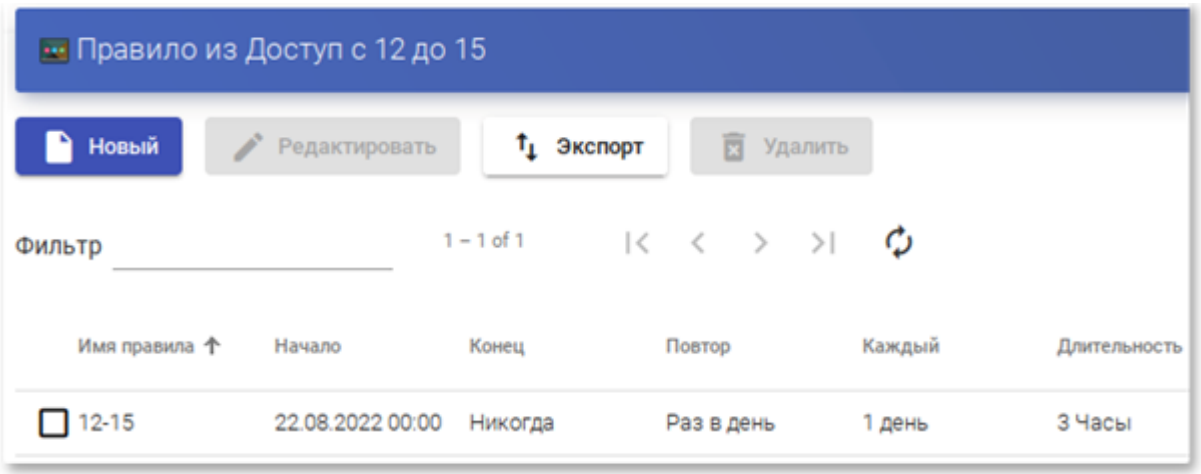

Рис. 145

7.3.8.1. Разрешение/запрет доступа

После настройки правил в календарях их можно использовать для управления доступом пользователей к службам рабочего стола или приложениям. Для этого следует выбрать «Пул услуг», перейти на вкладку «Доступ к календарям» и нажать на кнопку «Новый» (рис. 146).

В открывшемся окне необходимо указать приоритет доступа, выбрать календарь и указать действие, которое будет применяться при доступе к сервису (рис. 147).

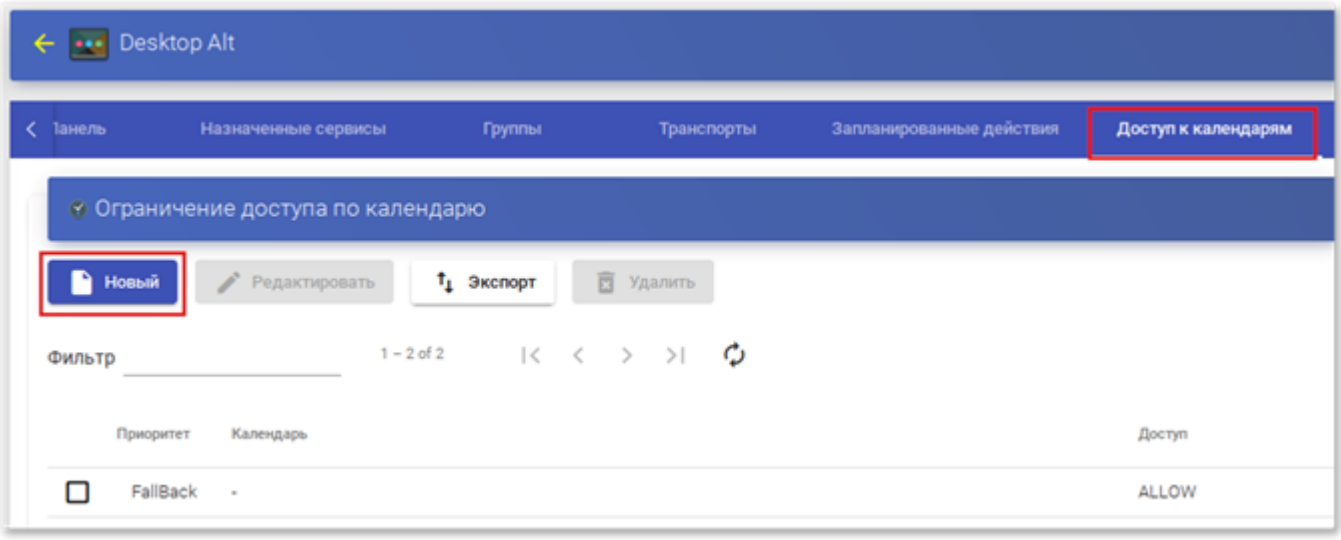

Рис. 146

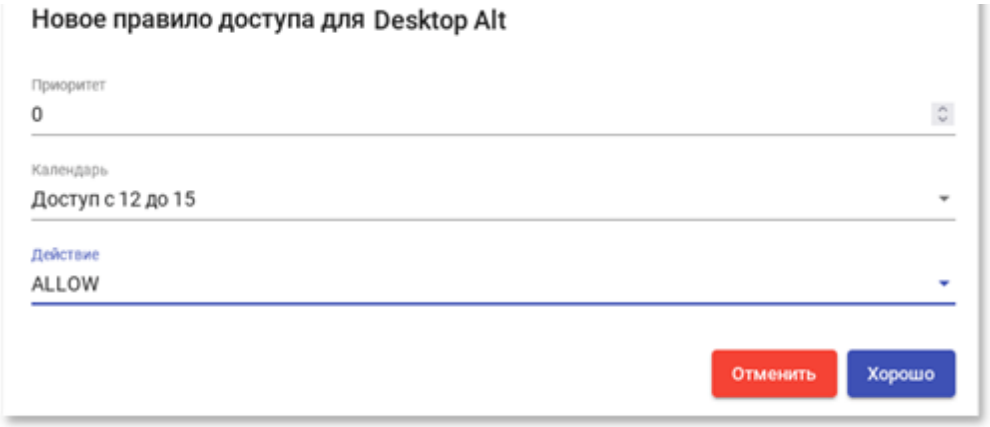

Рис. 147

П р и м е ч а н и е . Правило по умолчанию (FallBack) должно разрешать или запрещать доступ к сервису, когда календарь не применяется (рис. 148).

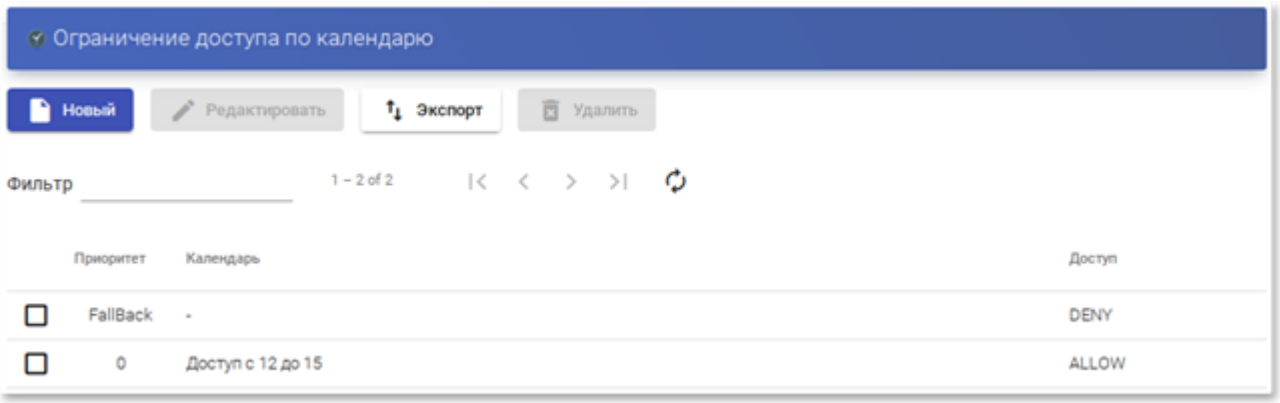

Рис. 148

Доступ к сервису «SL» запрещен (рис. 149).

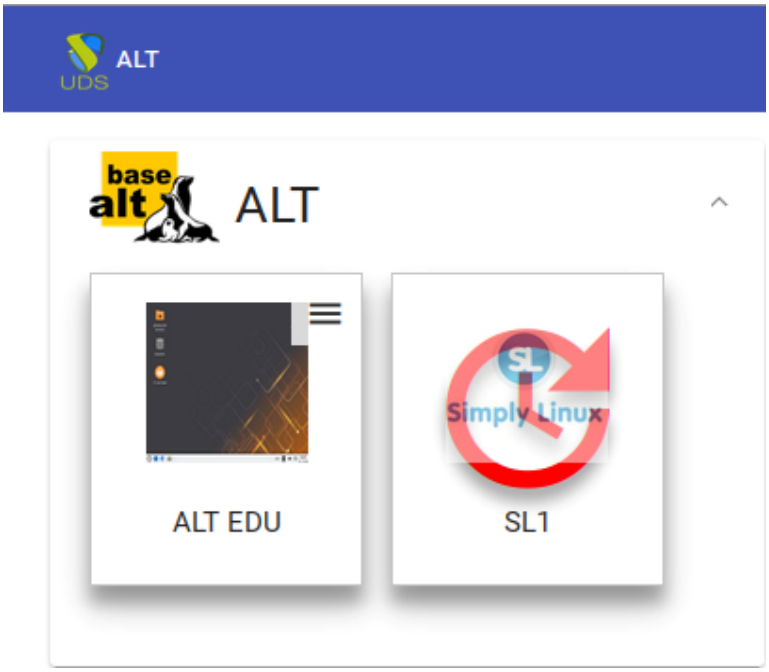

Рис. 149

7.3.8.2. Запланированные действия

После настройки правил в календарях их можно использовать для планирования определенных задач в «Пуле услуг». Для этого следует выбрать «Пул услуг», перейти на вкладку «Запланированные действия» и нажать на кнопку «Новый» (рис. 150).

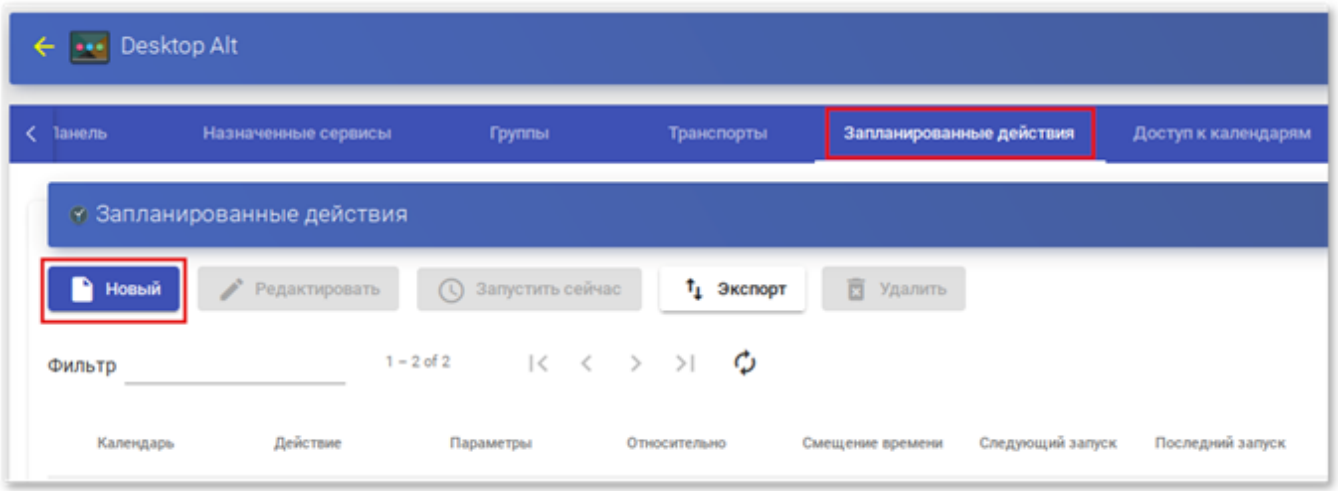

Рис. 150

В открывшемся окне необходимо указать календарь, время, в течение которого будет выполняться действие, выбрать действие, которое необходимо выполнить (рис. 151).

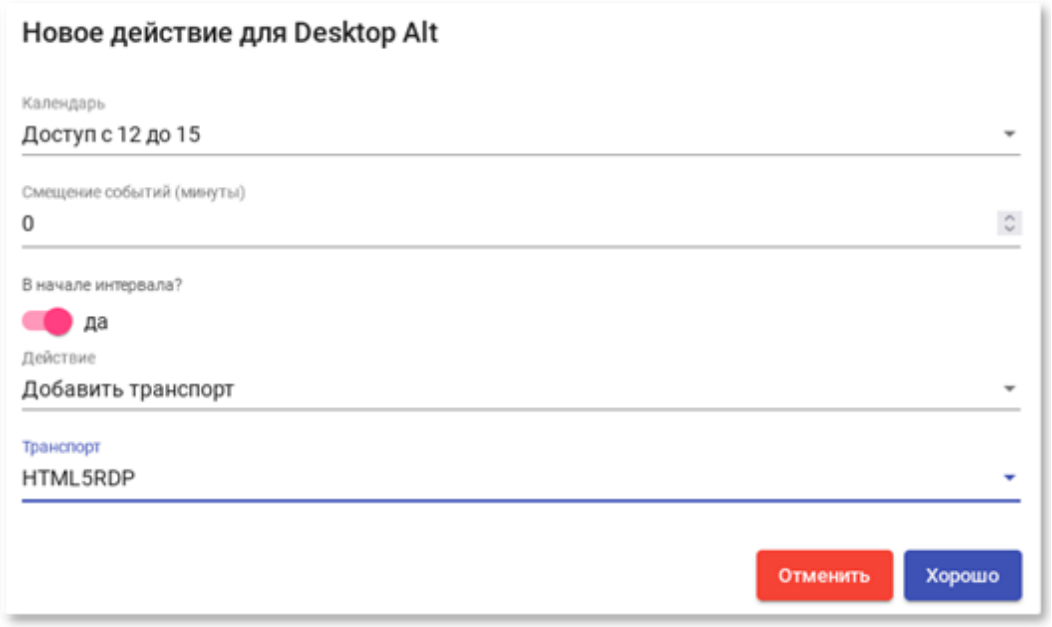

Рис. 151

Список возможных действий зависит от поставщика услуг данного пула:

- «Установить начальные сервисы» – сбрасывает минимальное количество созданных и настроенных виртуальных рабочих столов;
- «Установить размер кеша» – сбрасывает виртуальные рабочие столы, доступные в системном кеше. Эти рабочие столы будут настроены и готовы к назначению пользователю;
- «Установить максимальное количество сервисов» – изменяет максимальное количество виртуальных рабочих столов в «Пуле услуг»;
- «Установить размер L2 кэша» – сбрасывает виртуальные рабочие столы, доступные в кэше L2;
- «Публикация» – создание новой публикации в «Пуле услуг»;
- «Добавить транспорт» – добавляет существующий транспорт в «Пул услуг»;
- «Удалить транспорт» – удаляет транспорт из «Пула услуг»;
- «Удалить все транспорты» – удаляет весь транспорт из «Пула услуг»;
- «Добавить группу» – добавляет существующую группу в «Пул услуг»;

- «Удалить группу» – удаляет группу из «Пула услуг»;
- «Удалить все группы» – удаляет все группы из «Пула услуг»;
- «Устанавливает игнорирование неиспользуемых» – устанавливает параметр «Игнорировать неиспользуемые»;
- «Удалить ВСЕ назначенные пользовательские сервисы» – удаляет все службы, назначенные пользователям;
- «Удалить СТАРЫЕ назначенные пользовательские сервисы» – удаляет службы, назначенные пользователям, которые не использовались заданное время.

После сохранения появится запланированная задача, выполняющая конкретное действие в данном «Пуле услуг» (рис. 152).

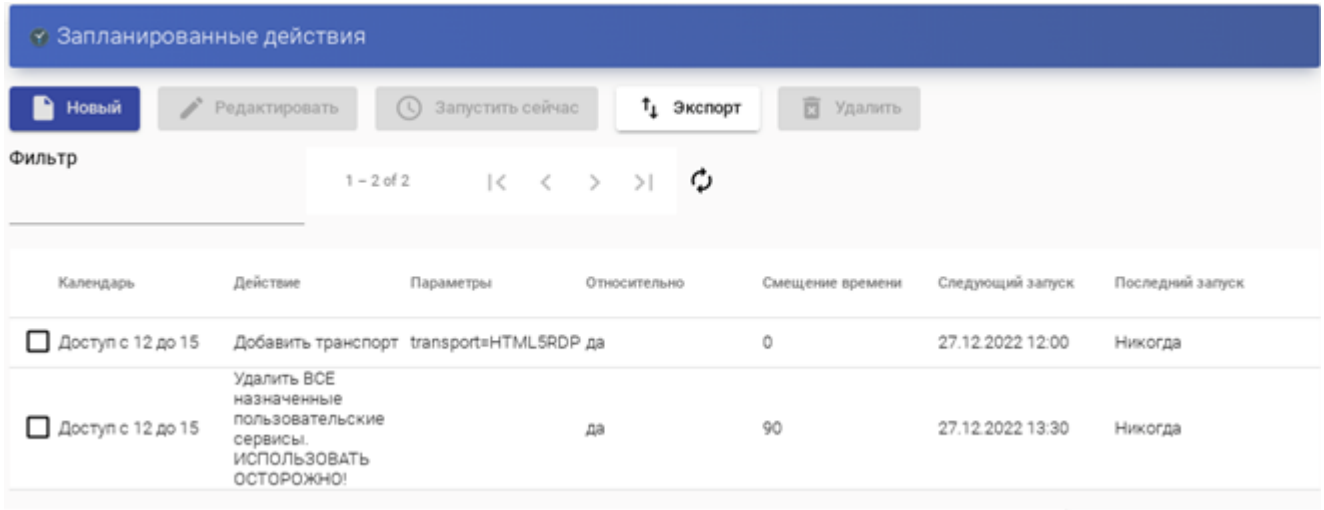

Рис. 152

7.3.9. Настройка разрешений

В OpenUDS можно назначать пользователям и группам пользователей права доступа к различным элементам администрирования. Разрешения будут назначены непосредственно для каждого элемента, а также будут применяться к его подэлементам.

Примечание. Чтобы пользователь мог получить доступ к администрированию, ему должна быть назначена роль «Штатный сотрудник» (рис. 153).

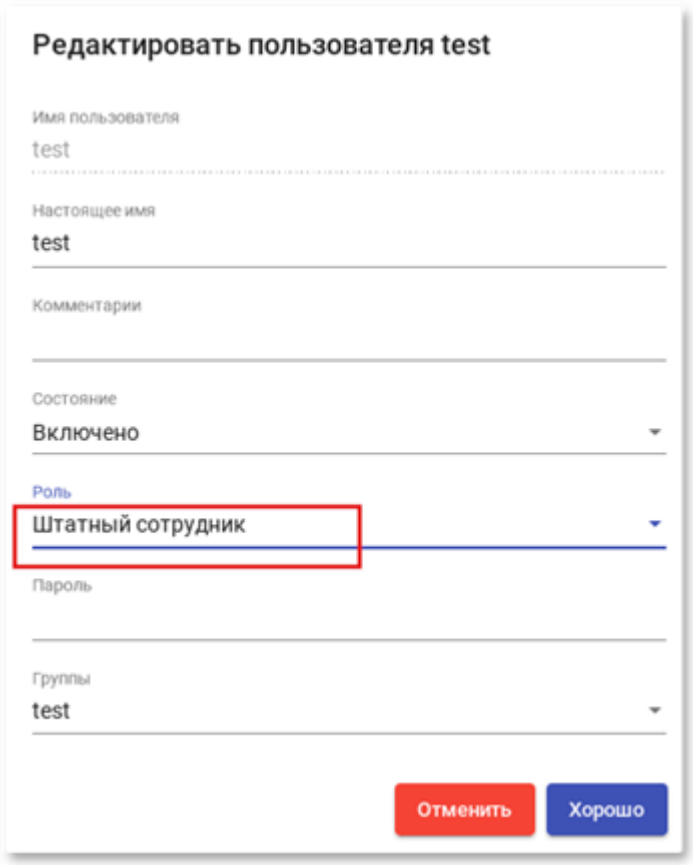

Рис. 153

Для предоставления разрешения к элементу администрирования следует выбрать элемент и нажать на кнопку «Разрешения» (рис. 154).

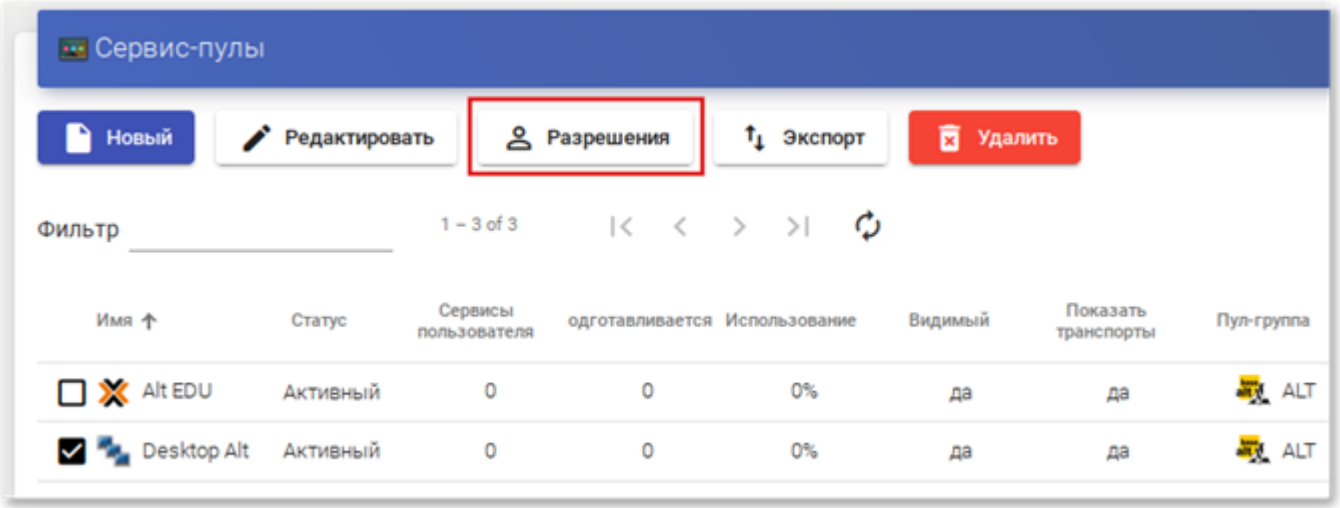

Рис. 154

В окне разрешений следует нажать ссылку «Новое разрешение…» для групп или пользователей, выбрать аутентификатор и группу/пользователя, к которым

будет применяться разрешение. Также нужно указать, будет ли пользователь/группа иметь доступ для чтения к элементу (Только чтение) или полный доступ (Полный доступ) (рис. 155).

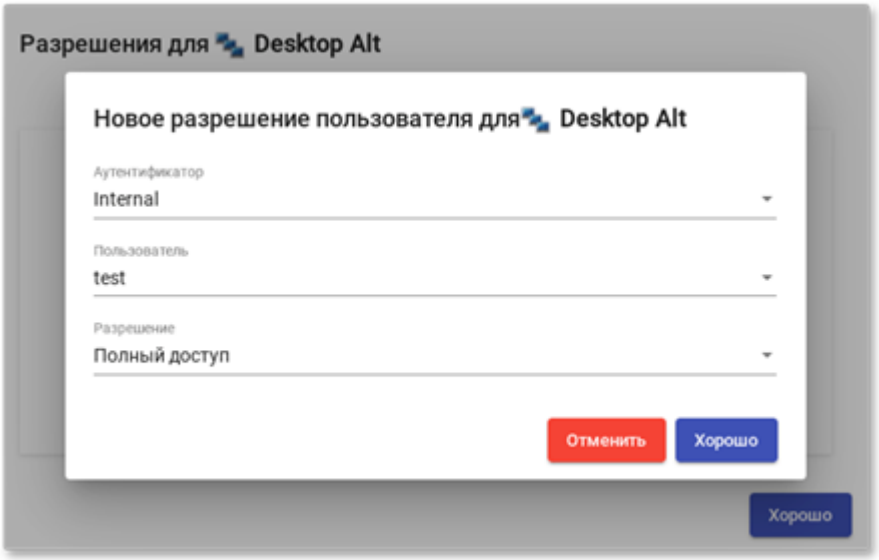

Рис. 155

После сохранения настроек, пользователи, которым назначена роль «Штатный сотрудник», смогут получить доступ к этому элементу администрирования с назначенными разрешениями (рис. 156).

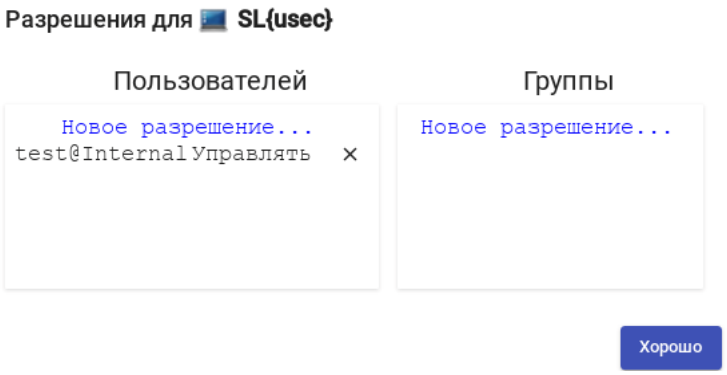

Рис. 156

Примечание. Разрешения типа «Полный доступ» (Управление) могут применяться только к элементам второго уровня («Календари», «Пулы услуг» и т. д.).

7.3.10. Конфигурация OpenUDS

В разделе «Конфигурация» можно настроить ряд параметров, которые будут определять работу системы. Эти параметры отвечают за определение таких аспектов, как безопасность, режим работы, подключение и т. д. как самой системы OpenUDS, так и ее связи с виртуальными платформами, зарегистрированными в OpenUDS.

ВАЖНО
Ниже описаны некоторые системные переменные, которые считаются наиболее полезными для управления виртуальными рабочими столами. Не рекомендуется изменять значения других переменных, так как некоторые из них указывают системе, как она должна работать (количество одновременных задач, время выполнения задач, плановые проверки и т. д.). Изменение этих параметров может привести неправильной работе или к полной остановке системы.

П р и м е ч а н и е . Для применения изменений, после редактирования значений любой из переменных конфигурации OpenUDS, необходимо перезапустить сервер OpenUDS.

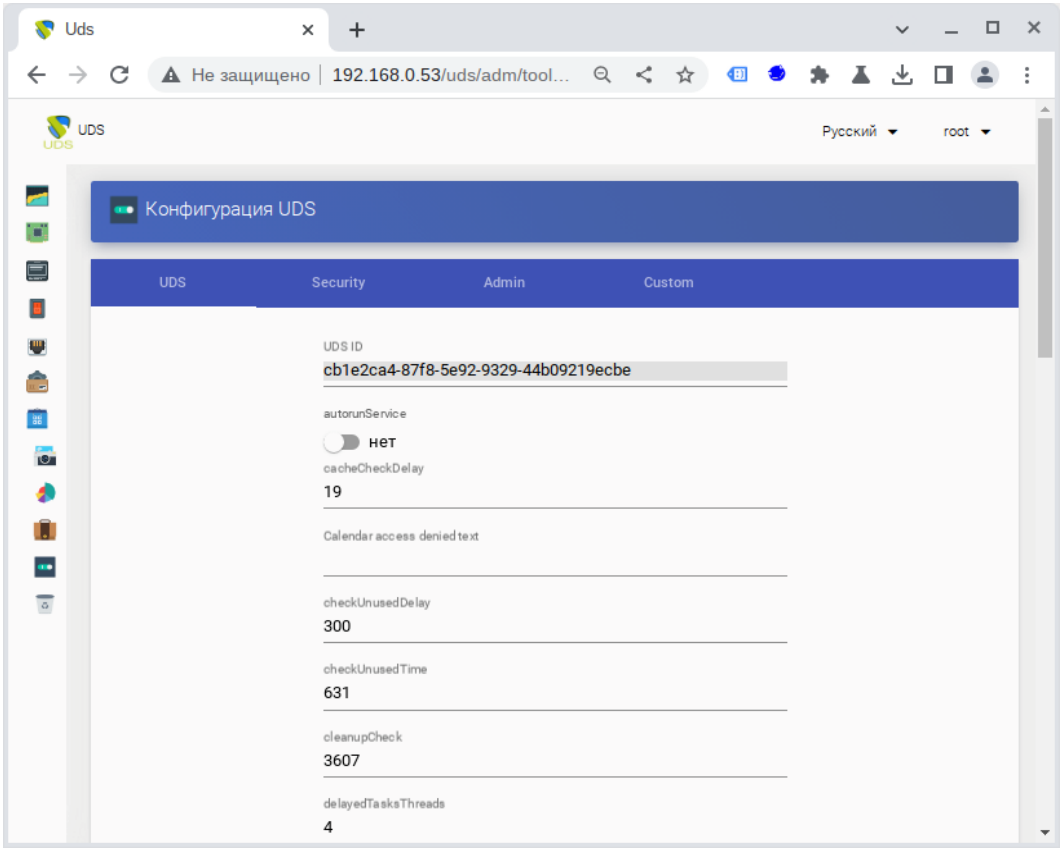

Рис. 157

1) Вкладка «UDS»:

- «AutorunService» выполнять прямой доступ к службе пользователя, если пользователю назначена только одна служба. Если этот параметр активирован, пользователи, которым назначен один сервис, будут подключаться к нему напрямую, минуя экран выбора сервиса и используя предварительно настроенный «Транспорт». По умолчанию: нет;
- «DisallowGlobalLogin» – если включено, на странице входа не будет отображаться список аутентификаторов (поле «Аутентификатор»). В этом случае, будет использоваться аутентификатор по умолчанию. Для предоставления пользователю доступа к системе с помощью других аутентификаторов необходимо будет использовать «Метку» («Label») (определенную в аутентификаторе) в URL-адресе доступа. По умолчанию: нет;
- «KeepInfoTime» – время (в секундах), в течение которого завершенные события «пула услуг» остаются видимыми. По умолчанию: 14401 секунд (4 часа);
- «RedirectToHttps» – автоматически перенаправлять доступ к OpenUDS с http на https (по умолчанию: нет);
- «SessionExpireTime» – максимальное время, в течение которого сеанс пользователя будет открыт после создания новой публикации. По истечении этого времени система закроет сеанс пользователя и продолжит удаление службы. Если у службы есть «Менеджер ОС» с параметром «Держать сервис привязанным даже в новой публикации», этот параметр не будет применяться. По умолчанию: 24 часа;
- «StatsDuration» – время, в течение которого система хранит статистику (по умолчанию: 365 дней).

2) Вкладка «Security»:
   - «AllowRootWebAccess» – разрешить суперпользователю входить в панель управления OpenUDS (пользователю, созданному при разворачивании OpenUDS-сервера). По умолчанию: да;
   - «Behind a proxy» – указывает системе, что серверы OpenUDS находятся «за» прокси-сервером (например, среда OpenUDS с HA Proxy). По умолчанию: нет;
   - «Block ip on login failure» – заблокировать пользователя, при неправильном вводе пароля (также блокируется IP-адрес). Количество попыток, указывается в переменной «`maxLoginTries`». По умолчанию: нет;
   - «Enforce Zero-Trust Mode» – включение режима нулевого доверия (запретить системе хранить пароли). По умолчанию: нет;
   - «LoginBlockTime» – время (в секундах), в течение которого после неправильного ввода пароля пользователь будет заблокирован. Количество попыток, указывается в переменной «`MaxLoginTries`». По умолчанию: 300 секунд (5 минут);
   - «MaxLoginTries» – количество попыток, за которые пользователь должен будет ввести свой пароль, прежде чем система заблокирует его;
   - «Session timeout for Admin» – время бездействия (в секундах) для администраторов платформы;
   - «Session timeout for User» – время бездействия (в секундах) для пользователей;
   - «Trusted Hosts» – узлы, которые OpenUDS считает безопасными. Эти узлы могут делать «sensitive» запросы к OpenUDS, такие как туннели. Допустимые значения: подсеть, диапазон IP-адресов, конкретные IP-адреса. По умолчанию: "*" (все разрешено).
3) Вкладка «Администрирование» («Admin»):
   - «Trusted Hosts for Admin» – узлы, с которых можно управлять OpenUDS (как с помощью веб-доступа, так и администрирование с

помощью API). Допустимые значения: подсеть, диапазон IP-адресов, конкретные

IP-адреса. По умолчанию: "*" (все разрешено).

4) На вкладке «Custom» задаются параметры, связанные с графической настройкой OpenUDS:

- «CSS» – CSS код для изменения стиля страниц OpenUDS;
- «Logo name» – текст, который отображается рядом с логотипом;
- «Min. Services to show filter» – минимальное количество служб, которые должны существовать у пользователя (в режиме пользователя), чтобы отображался фильтр;
- «Show Filter on Top» – расположение панели поиска на странице пользовательских служб;
- «Site copyright info» – текст копирайт;
- «Site copyright link» – веб-адрес, на который будет вести ссылка с копирайта;
- «Site information» – HTML-код для частичной настройки страницы входа в OpenUDS. Введенный код появится под полем входа пользователя;
- «Site name» – текст, который будет отображаться в верхней части поля входа пользователя на странице входа OpenUDS.

7.4. Подготовка шаблона виртуальной машины

Для возможности использования ВМ в качестве шаблона OpenUDS, на машине необходимо включить и настроить удаленный рабочий стол, установить OpenUDS Actor и зарегистрировать его на сервере OpenUDS.

7.4.1. Шаблон ВМ с ОС Альт

Подготовить шаблон ВМ (все действия выполняются на ВМ):

1) Установить openuds-actor:

```
# apt-get install openuds-actor
```

2) Включить автозапуск сервиса `udsactor.service`:

```
# systemctl enable udsactor.service
```

3) Зарегистрировать OpenUDS Actor на сервере OpenUDS:

- запустить OpenUDS Actor из меню «Настройки» → «UDS Actor Configuration» или командой:

  ```
  $ /usr/sbin/UDSActorConfig-pkexec
  ```

  Потребуется ввести пароль пользователя, входящего в группу wheel.

- на вкладке «UDS Server» указать имя или IP-адрес сервера OpenUDS, аутентификатор (значение Administration соответствует суперпользователю), имя и пароль пользователя, имеющего права администратора в среде OpenUDS и нажать на кнопку «Register with UDS» (Зарегистрироваться в UDS) (рис. 158);

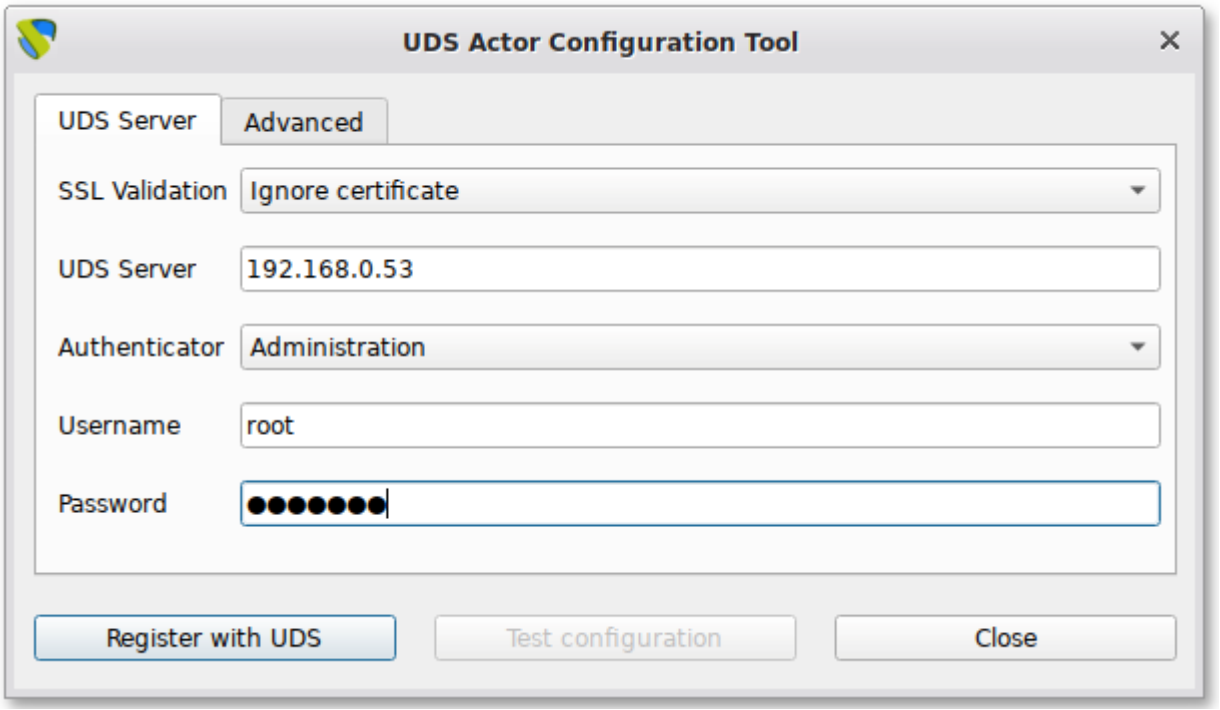

Рис. 158

- на вкладке «Advanced» можно указать дополнительные параметры, в том числе уровень журналирования:

  а) «Preconnect» – сценарий, который будет запущен непосредственно перед тем, как пользователь подключится к виртуальному рабочему столу. Скрипту могут быть переданы

следующие параметры: имя пользователя, протокол, IP-адрес, имя хоста;

б) «Runonce» – сценарий, который будет запущен только один раз перед настройкой UDS Actor. После выполнения скрипт удаляется из конфигурации. Параметры можно передать непосредственно скрипту. Необходимо, чтобы выполняемый скрипт завершился перезапуском виртуального рабочего стола;

в) «Postconfig» – сценарий, который будет запущен после того, как UDS Actor завершит настройку. Параметры можно передать непосредственно скрипту. Скрипт запускается только один раз, но в отличие от режима «Runonce» перезапускать виртуальный рабочий стол не нужно;

г) «Log Level» – уровень журналирования (файл журнала: `/var/log/udsactor.log`).

Для применения настроек, указанных на этой вкладке необходимо выполнить перерегистрацию UDS Actor.

4) Настроить один из вариантов удаленного доступа:

- XRDP:

а) установить пакет xrdp:

```
# apt-get install xrdp
```

б) включить сервисы xrdp и xrdp-sesman:

```
# systemctl enable --now xrdp
# systemctl enable --now xrdp-sesman
```

в) для доступа к терминальному сеансу включить пользователя в группу tsusers:

```
# gpasswd -a user tsusers
```

- X2Go:

а) установить пакет x2goserver:

```
# apt-get install x2goserver
```

б) включить сервис x2goserver:

```
# systemctl enable --now x2goserver
```

7.4.2. Шаблон ВМ с ОС Windows

Примечание. В данном разделе рассмотрен процесс настройки ВМ с ОС Windows x64 10 Pro для использования в качестве шаблона OpenUDS.

Требования к шаблону ВМ с ОС Windows:

- рекомендуется отключить автоматические обновления, чтобы предотвратить выполнение этого процесса на создаваемых виртуальных рабочих столах;
- машина должна получать IP-адрес по DHCP;
- шаблон не нужно добавлять в домен Active Directory. Если нужны виртуальные рабочие столы, включенные в домен AD, настройка должна быть выполнена в панели управления OpenUDS;
- автоматический вход пользователя должен быть отключен (учетные данные всегда должны запрашиваться у пользователя).

Примечание. Для возможности ввода ВМ в домен, в шаблоне ВМ должен быть доступен сервер DNS, имеющий записи про контроллер домена Active Directory.

Для настройки удаленного рабочего стола, необходимо выполнить следующие действия в шаблоне ВМ:

1) открыть окно «Параметры» (Win+I);
2) выбрать раздел «Система», а затем слева в списке – «Удаленный рабочий стол»;
3) ползунок «Включить удаленный рабочий стол» установить в положение «Вкл.» (рис. 159);
4) выбрать учетные записи, которым разрешено удаленное подключение. Для этого нажать ссылку «Выберите пользователей, которые могут получить доступ к этому компьютеру» и добавить пользователей (рис. 160);
5) проверить возможность подключения к машине удаленно.

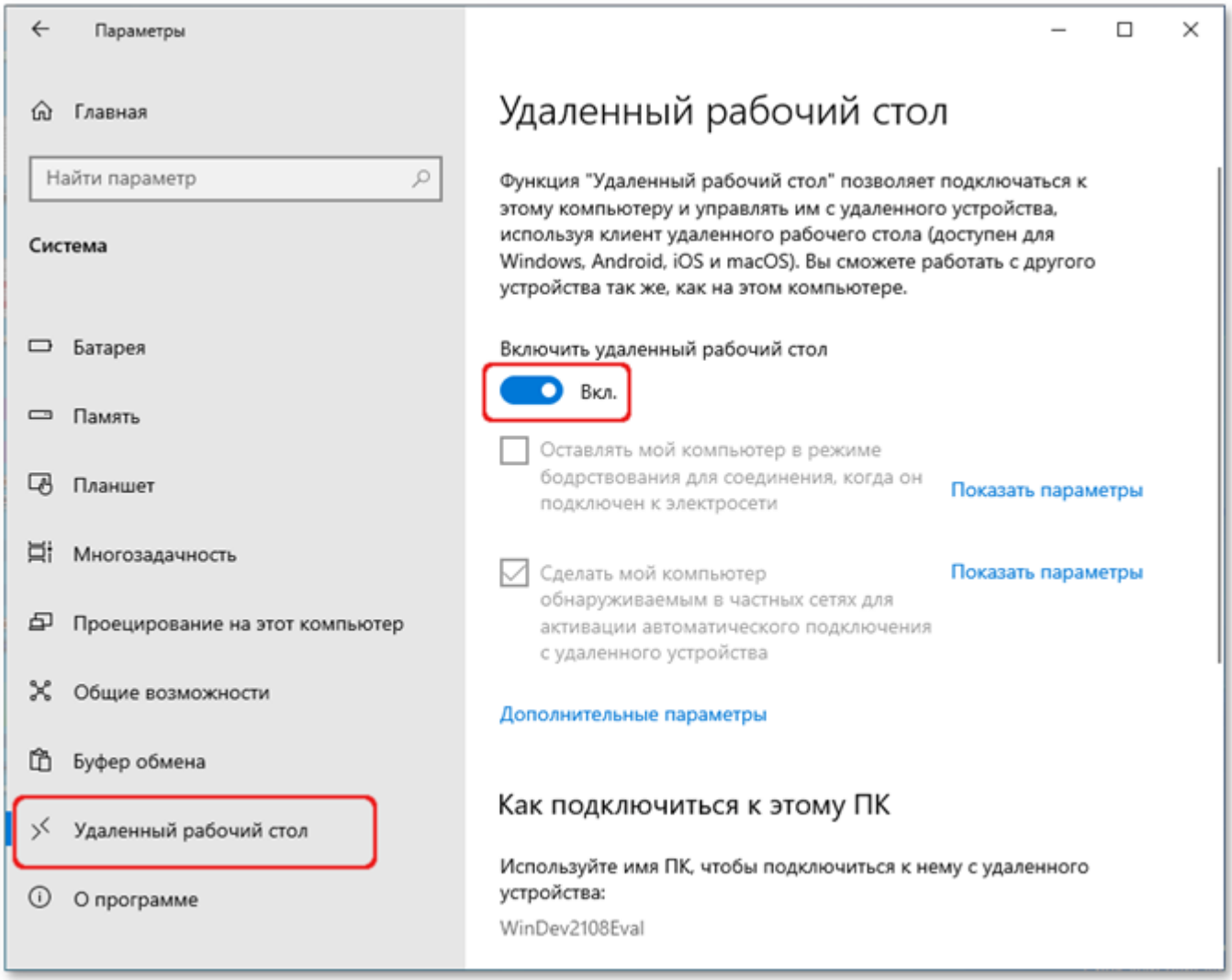
Параметры
Главная
Найти параметр
Система
Батарея
Память
Планшет
Многозадачность
Проецирование на этот компьютер
Общие возможности
Буфер обмена
Удаленный рабочий стол
О программе
Удаленный рабочий стол
Функция "Удаленный рабочий стол" позволяет подключаться к этому компьютеру и управлять им с удаленного устройства, используя клиент удаленного рабочего стола (доступен для Windows, Android, iOS и macOS). Вы сможете работать с другого устройства так же, как на этом компьютере.
Включить удаленный рабочий стол
Вкл.
Оставлять мой компьютер в режиме бодрствования для соединения, когда он подключен к электросети
Показать параметры
Сделать мой компьютер обнаруживаемым в частных сетях для активации автоматического подключения с удаленного устройства
Показать параметры
Дополнительные параметры
Как подключиться к этому ПК
Используйте имя ПК, чтобы подключиться к нему с удаленного устройства:
WinDev2108Eval

Рис. 159

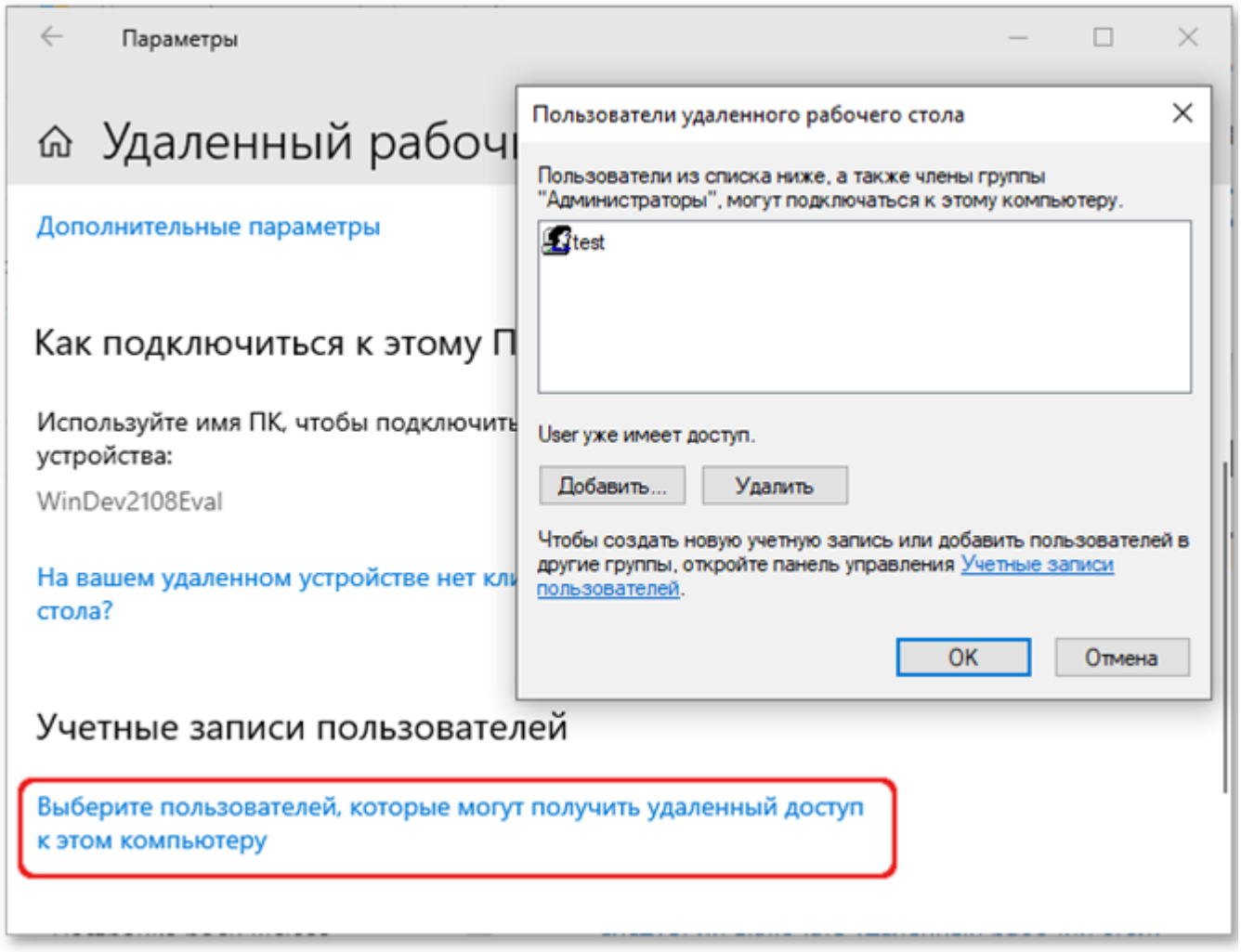

Рис. 160

Примечание. Для возможности подключения клиентов Linux может потребоваться снять отметку с пункта «Требовать использование компьютерами аутентификации на уровне сети для подключения» в дополнительных параметрах (рис. 161).

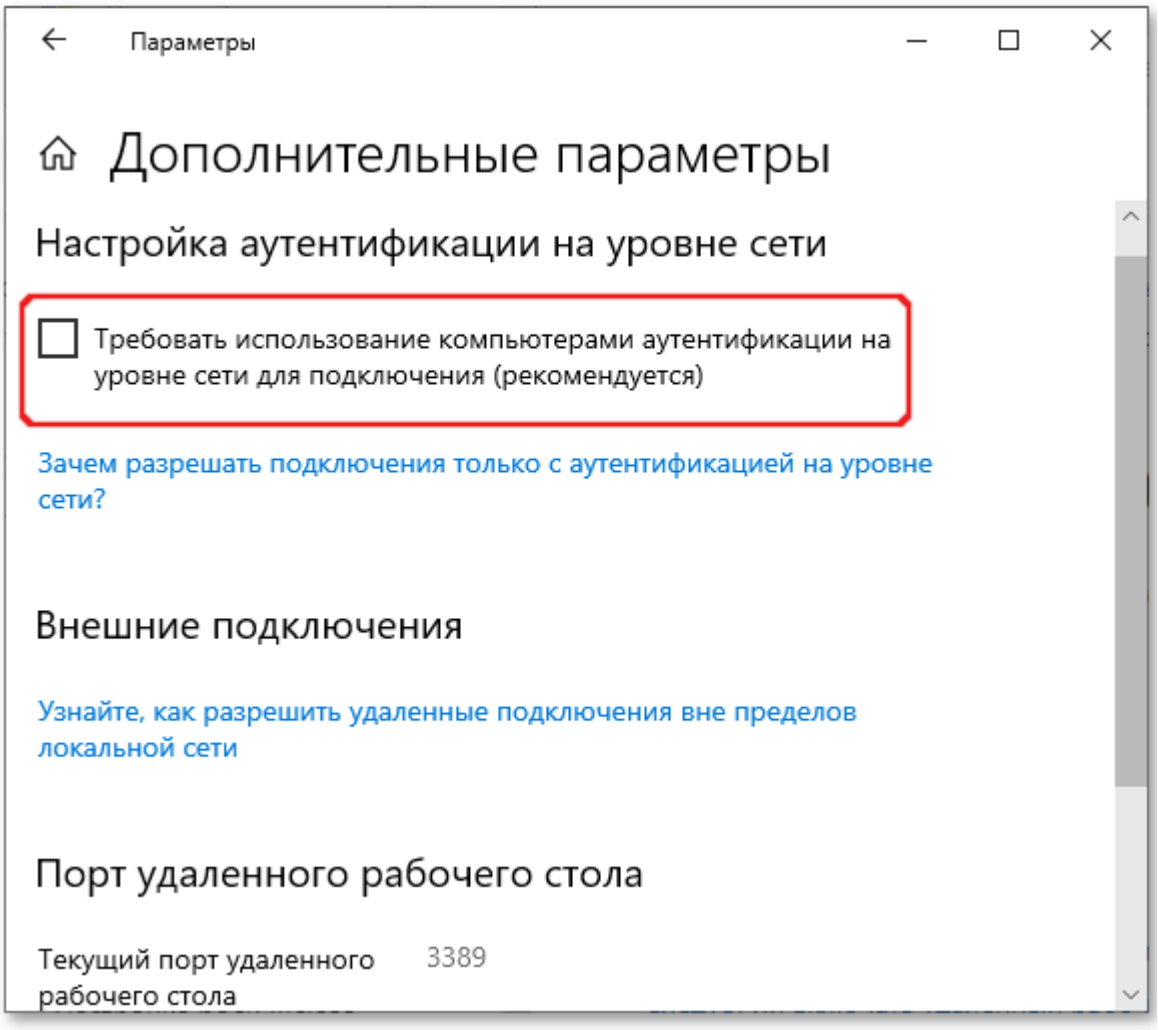

Рис. 161

ВАЖНО Необходимо убедиться, что межсетевой экран не блокирует соединения по 3389 порту.

7.4.2.1. Настройка OpenUDS Actor

1) Загрузить OpenUDS Actor. Для этого в панели управления OpenUDS Server выбрать пункт «Загрузки» (пункт доступен пользователям с правами администратора) (рис. 162).

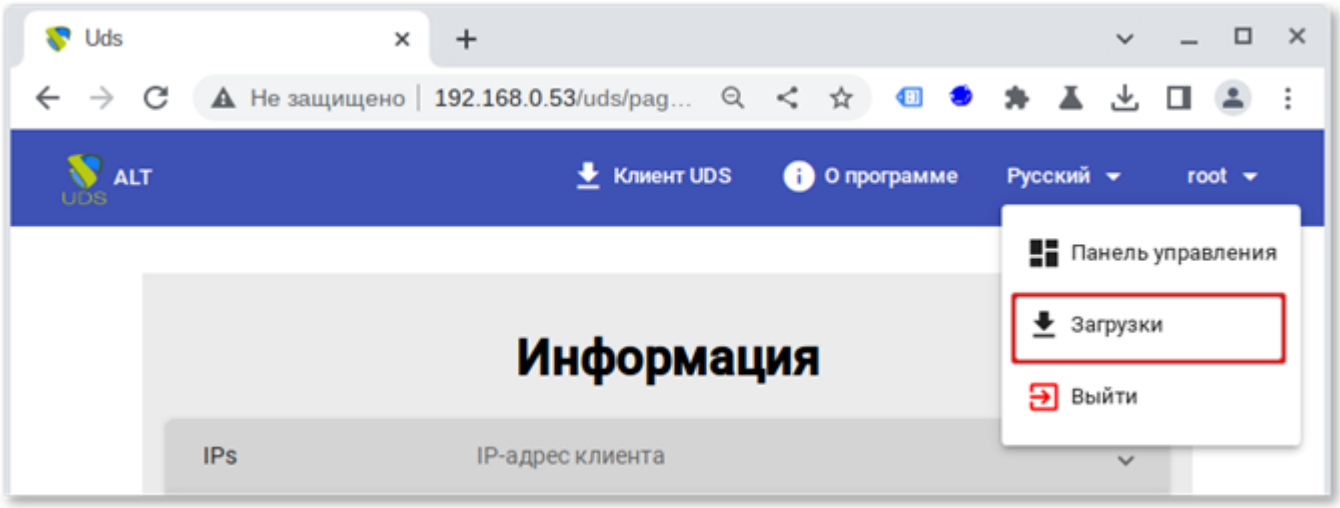

Рис. 162

На открывшейся странице выбрать нужный UDS Actor (рис. 163).

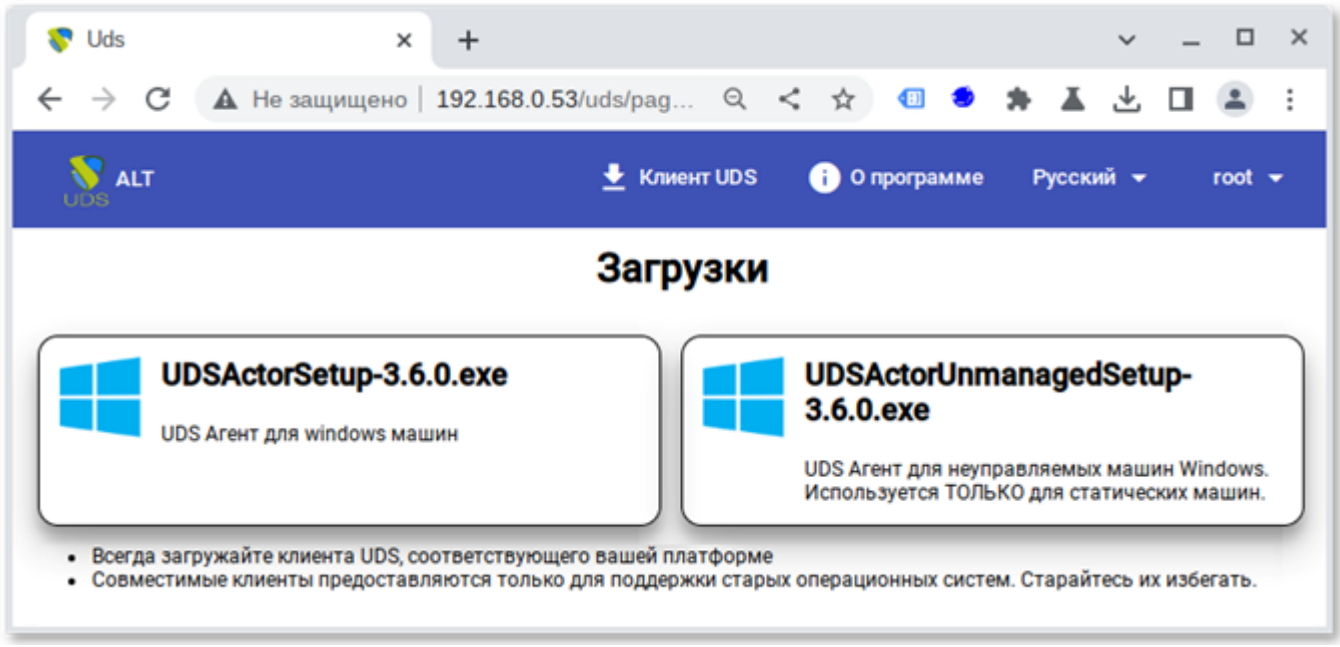

Рис. 163

П р и м е ч а н и е . Для машин с ОС Windows есть два вида OpenUDS Actor:
- openUDS-Managed_Installer – для управляемых Windows машин;
- openUDS-Unmanaged_Installer – для неуправляемых Windows машин.

Используется только для отдельных серверов без виртуализации.

2) Установить OpenUDS Actor (установка OpenUDS Actor ничем не отличается от инсталляции большинства других программ в ОС Windows).

3) Запустить UDSActorConfig от имени администратора. Для этого в контекстном меню пункта UDSActorConfig выбрать «Дополнительно» → «Запуск от имени администратора» (рис. 164).

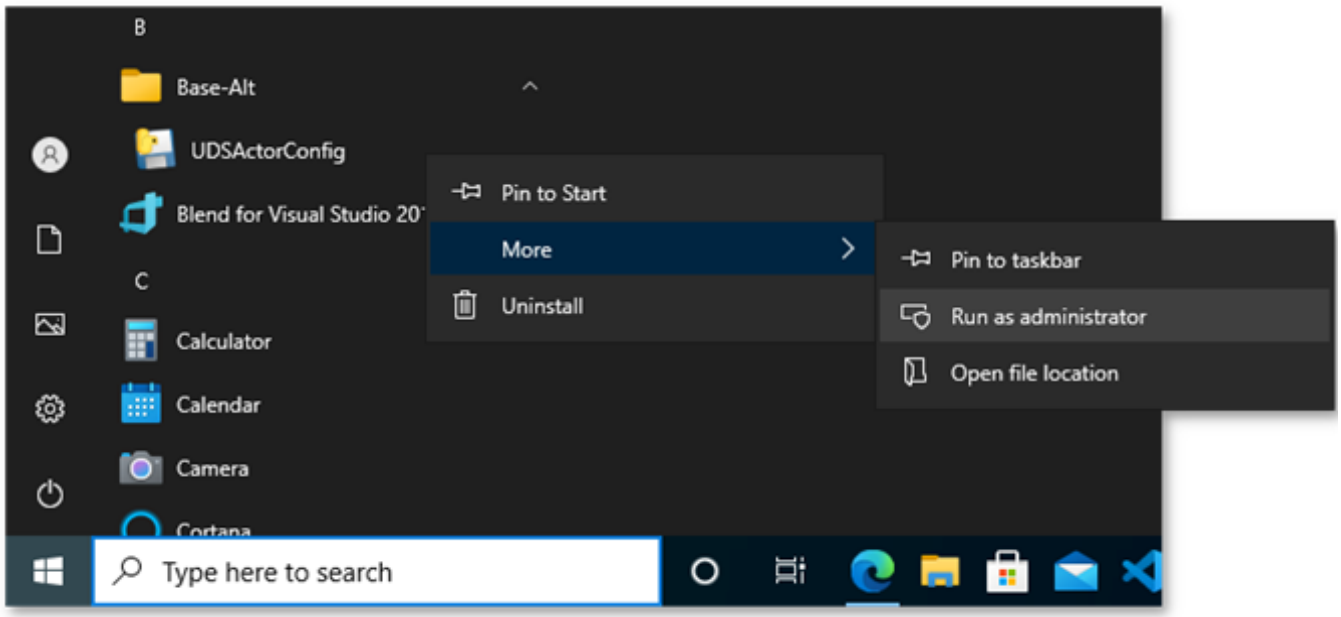

Рис. 164

4) Регистрация OpenUDS Actor на сервере:

- для регистрации Managed OpenUDS Actor на вкладке «UDS Server» необходимо указать имя или IP-адрес сервера OpenUDS, аутентификатор (значение Administration соответствует суперпользователю), имя и пароль пользователя, имеющего права администратора в среде OpenUDS и нажать на кнопку «Register with UDS» («Зарегистрироваться в UDS») (рис. 165);

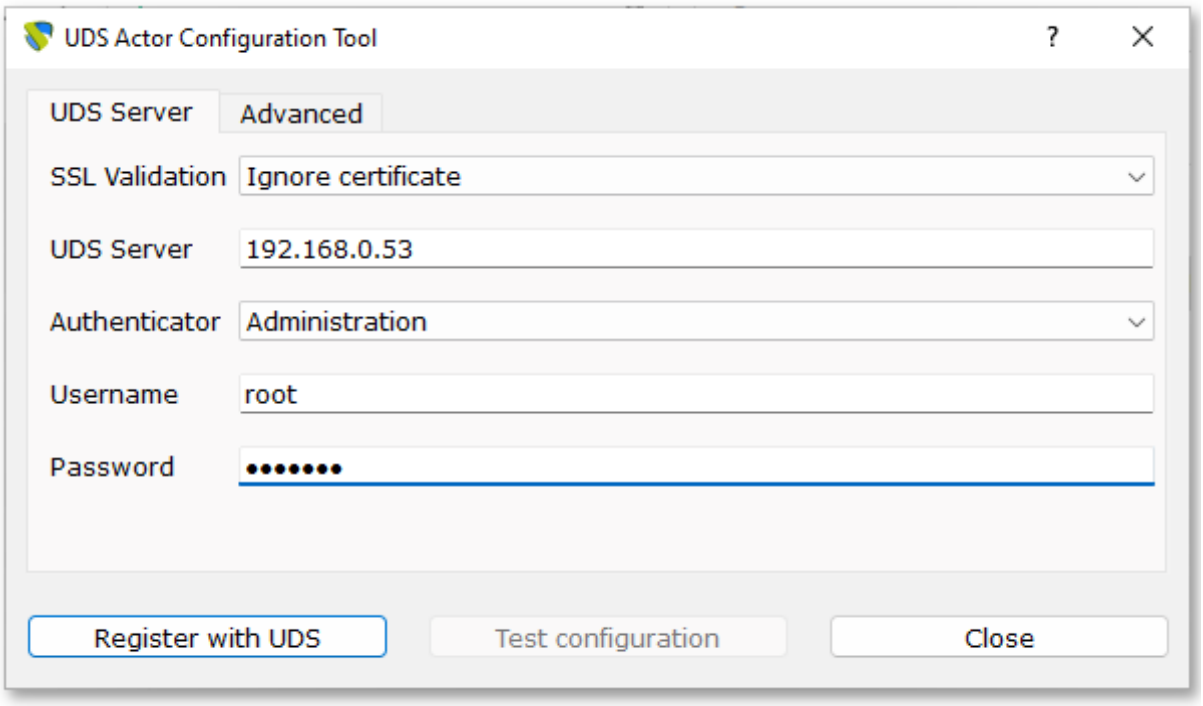

Рис. 165

- для регистрации Unmanaged OpenUDS Actor необходимо указать имя или IP-адрес сервера OpenUDS, тот же ключ, который был указан при настройке услуги «Статический множественный IP-адрес» и нажать на кнопку «Save Configuration» («Сохранить конфигурацию») (рис. 166).

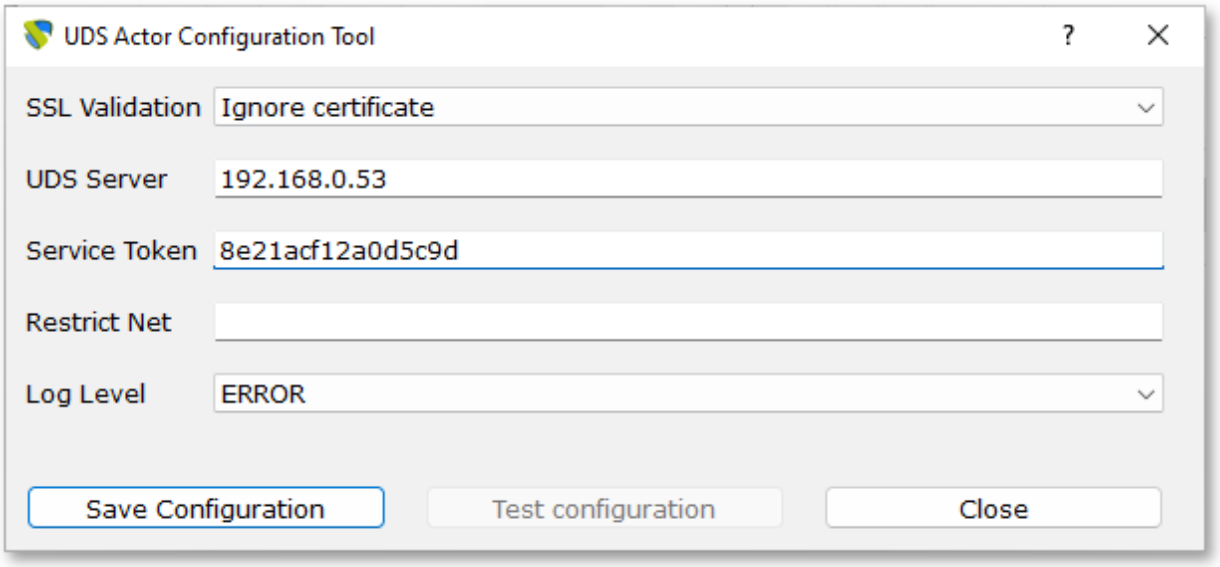

Рис. 166

Примечание. Unmanaged OpenUDS Actor уведомляет OpenUDS, когда пользователь входит в систему и выходит из нее. Благодаря этой функции система может освободить компьютер, при выходе пользователя из системы. Для использования этой функции при регистрации услуги «Статический множественный IP-адрес» кроме названия услуги следует указать один или несколько IP-адресов машин, к которым будет осуществляться доступ и ключ в поле «Ключ услуги» (рис. 167).

Если оставить поле «Ключ услуги» пустым, сеанс останется назначенным пользователю, пока администратор не удалит его вручную.

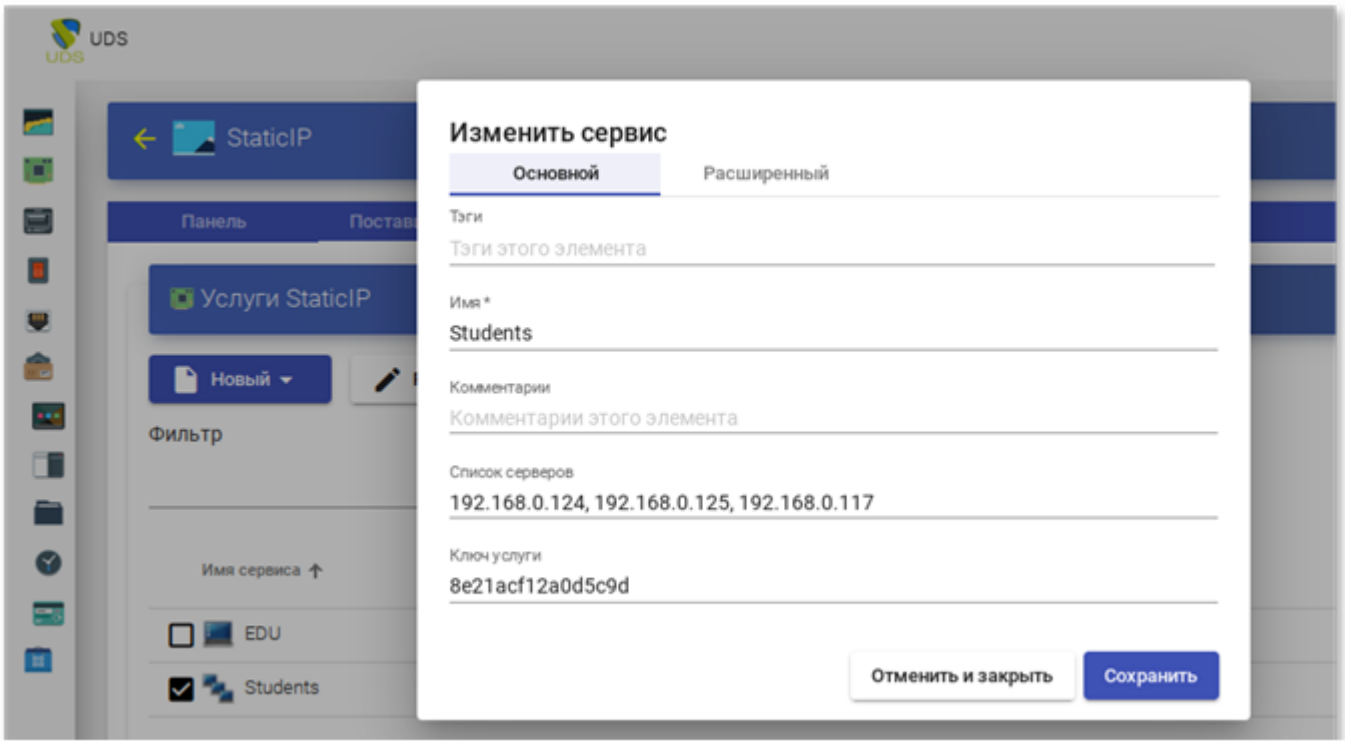

Рис. 167

7.5. Настройка клиента OpenUDS

Для возможности подключения к брокеру соединений и дальнейшего получения доступа к виртуальному рабочему окружению на клиентской машине должны быть установлены OpenUDS Client и клиенты каждого используемого протокола удаленного доступа.

Примечание. Если для доступа к виртуальному рабочему используется транспорт HTML5 RDP, нет необходимости устанавливать клиент OpenUDS на клиентский компьютер. Единственным требованием для этого подключения является наличие веб-браузера.

На клиенте должен быть установлен пакет openuds-client:

```
# apt-get install openuds-client
```

Чтобы иметь возможность подключаться к виртуальному рабочему столу, должны быть установлены клиенты каждого используемого протокола удаленного доступа (xfreerdp, x2goclient).

Для подключения к виртуальному рабочему столу по протоколу SPICE необходимо установить remote-viewer из пакета virt-viewer. На клиенте с ОС Windows необходимо установить virt-viewer (`https://releases.pagure.org/virt-viewer/`).

Примечание. Для возможности подключения по протоколу SPICE к OpenNebula, клиенты должны успешно разрешать имена hostname серверов с виртуальными машинами (через DNS или hosts).

7.5.1. Клиент с ОС Альт

На клиенте должен быть установлен пакет openuds-client:

```
# apt-get install openuds-client
```

Для возможности подключения к виртуальному рабочему столу, должны быть установлены клиенты протоколов удаленного доступа:

- xfreerdp – для подключения по протоколу RDP;
- x2goclient – для подключения к серверу X2Go;
- remote-viewer из пакета virt-viewer – для подключения по протоколу SPICE.

7.5.2. Клиент с ОС Windows

Установка клиента OpenUDS

1) Скачать OpenUDS Client для компьютеров с ОС Windows. Для этого в панели управления OpenUDS Server выбрать пункт «Клиент UDS» и на открывшейся странице выбрать клиент Windows (рис. 168).

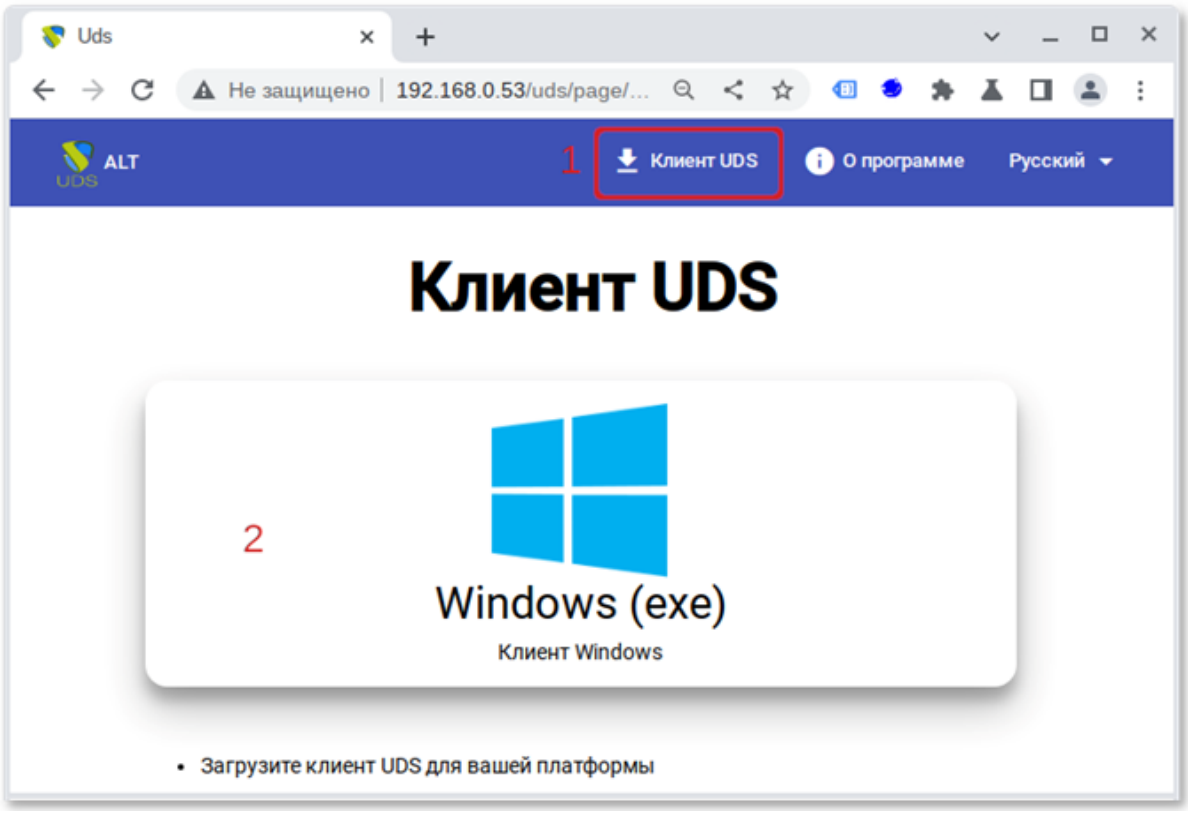

Рис. 168

2) Установить OpenUDS Client (установка ничем не отличается от инсталляции большинства других программ в ОС Windows).

Чтобы иметь возможность подключаться к виртуальному рабочему столу, должны быть установлены клиенты каждого используемого протокола удаленного доступа: RDP (стандартный клиент RDP установлен в Windows по умолчанию), X2Go, SPICE.

Примечание. Для установки клиента X2Go на ОС Windows достаточно загрузить клиент X2Go и установить его.

Для установки клиента SPICE на ОС Windows необходимо установить virt-viewer.

7.6. Подключение пользователя к виртуальному рабочему месту

Подключение к виртуальному рабочему месту:

1) подключиться к серверу OpenUDS с помощью веб-браузера `http://<openuds_address>`, выбрать средство проверки подлинности, если доступно несколько, ввести имя пользователя/пароль (их должен указать администратор) (рис. 169);

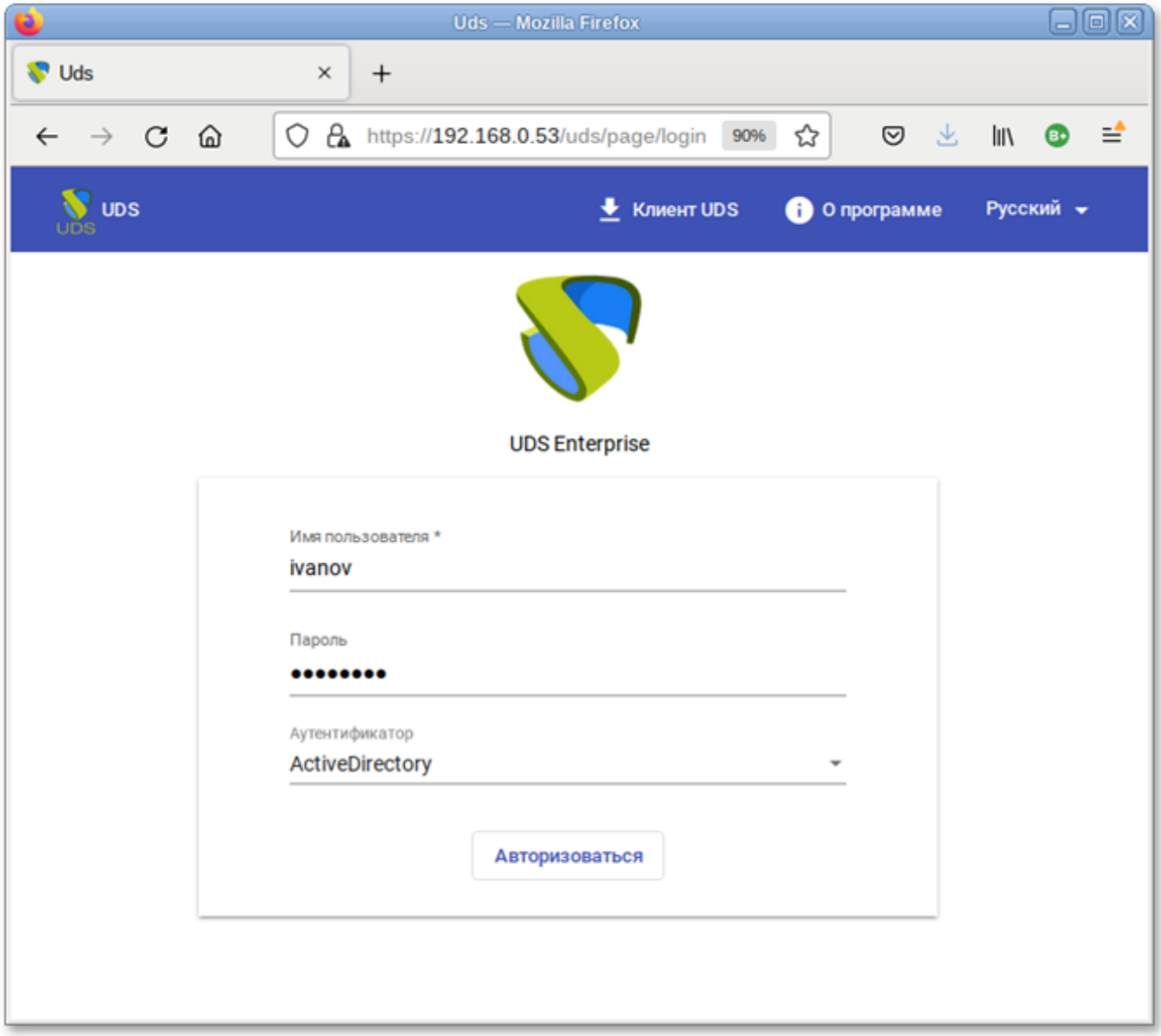

Рис. 169

2) на панели управления будут отображены все ВМ (или шаблоны), к которым у пользователя есть доступ (рис. 170);

3) при выборе ВМ, автоматически загрузится openuds-client, который запустит приложение для просмотра удаленного рабочего стола.

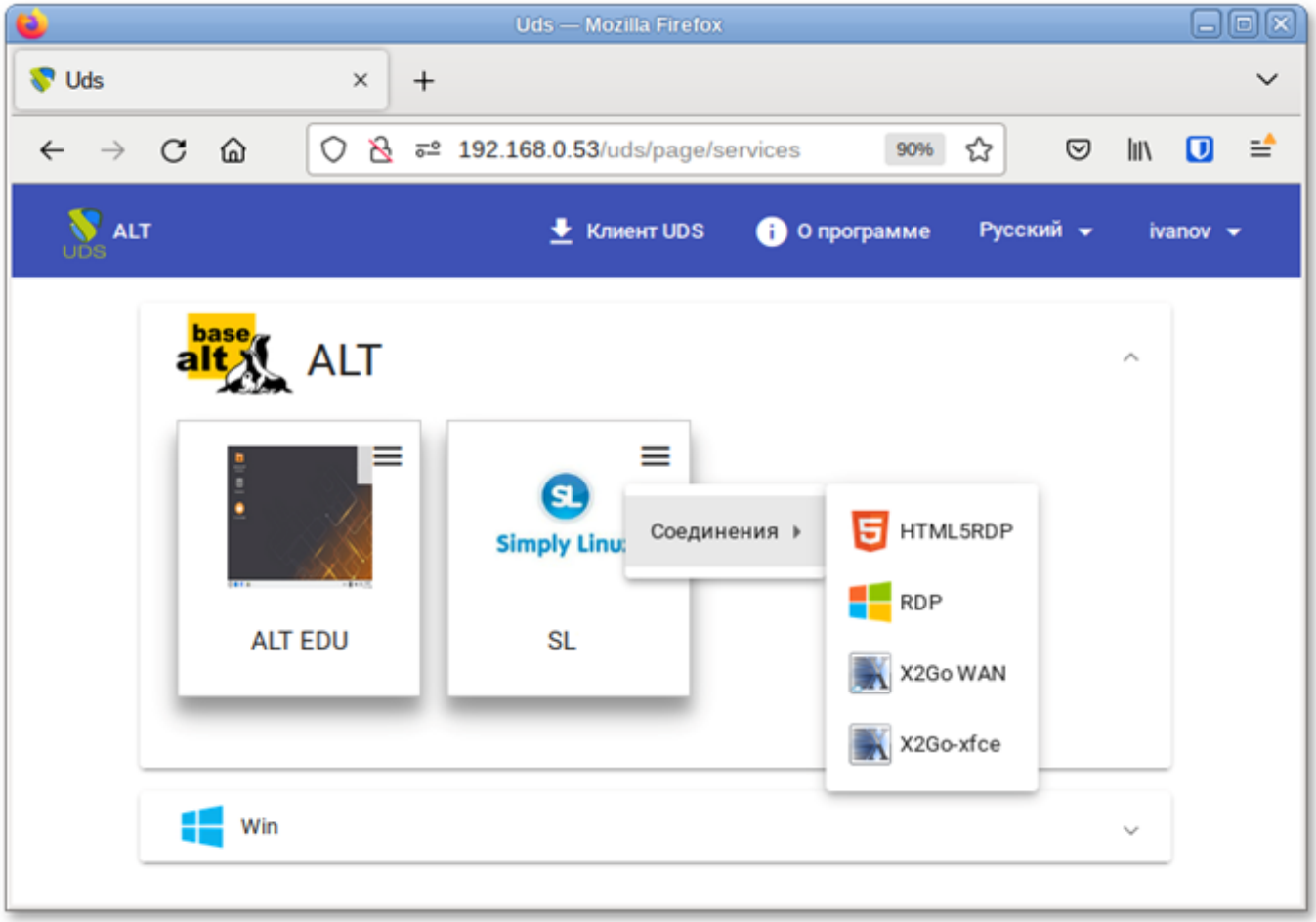

Рис. 170

После выбора пула, автоматически стартует OpenUDS Client, который обрабатывает URL, получает необходимые настройки протокола удаленного доступа для предоставленной (свободной) ВМ, формирует файл описания сессии и передает его приложению-клиенту удаленного доступа, которое и устанавливает соединение с указанной ВМ. Как только соединение будет установлено, виртуальный рабочий стол будет доступен для использования (рис. 171).

Примечание. Если для подключения к ВМ настроено более одного типа транспорта, то в правом верхнем углу службы будет отображена кнопка. Если выбрать непосредственно ВМ, будет вызван транспорт по умолчанию (транспорт с меньшим значением в поле приоритет). Для того чтобы использовать другой транспорт, нужно выбрать его в раскрывающемся списке.

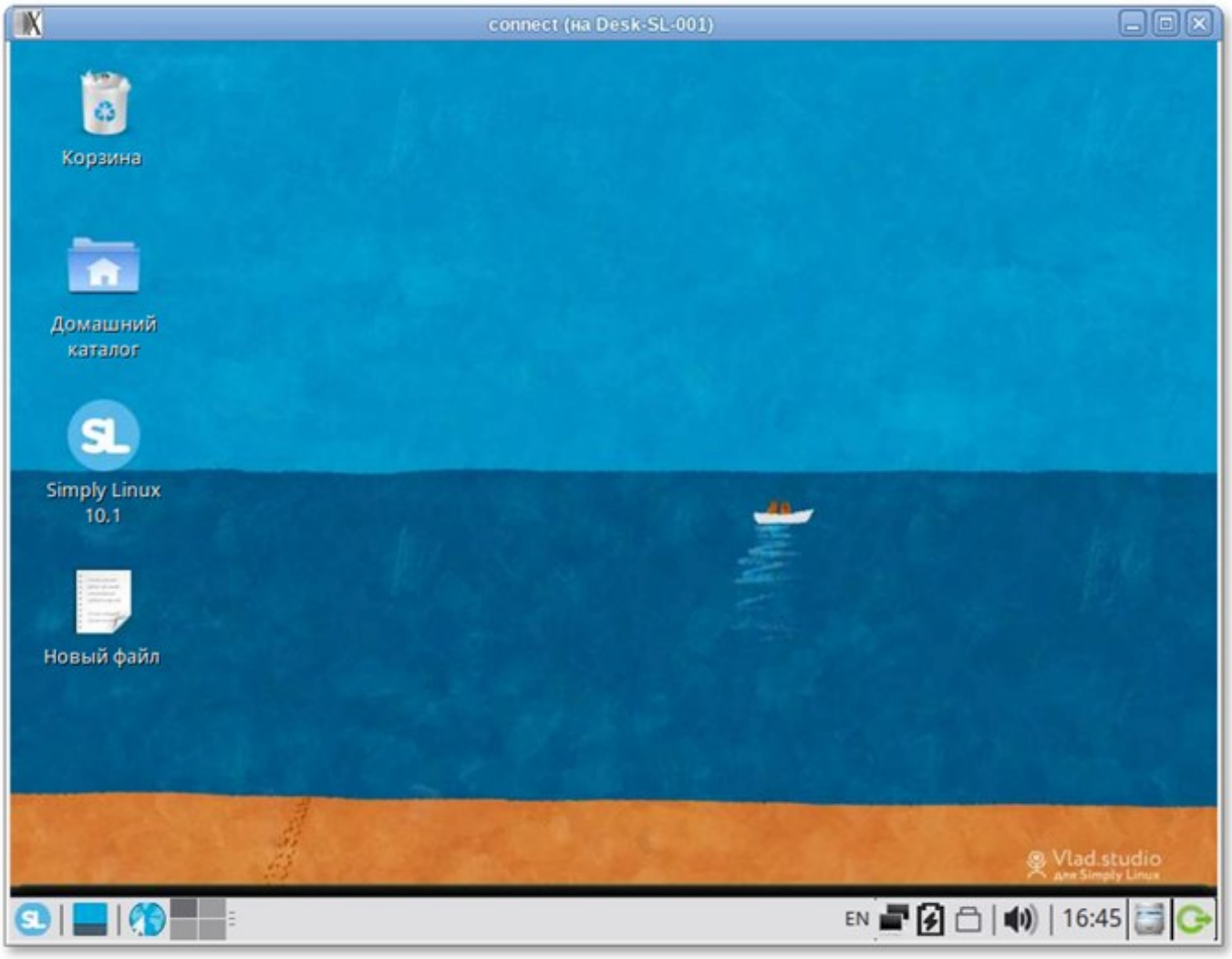

Рис. 171

По завершении сеанса пользователь ВМ выходит из нее, что приводит к остановке OpenUDS Actor. Брокер OpenUDS считает, что ВМ стала недоступной и, если пул постоянный, то он запускает ВМ, а если пул временный, то происходит удаление файлов ВМ в хранилище и создается новая ВМ из мастер-образа.

Примечание. При подключении пользователя к виртуальному рабочему месту OpenUDS фиксирует доступ и отображает информацию о привязанном сервисе на вкладке «Назначенные услуги» соответствующего пула (рис. 172).

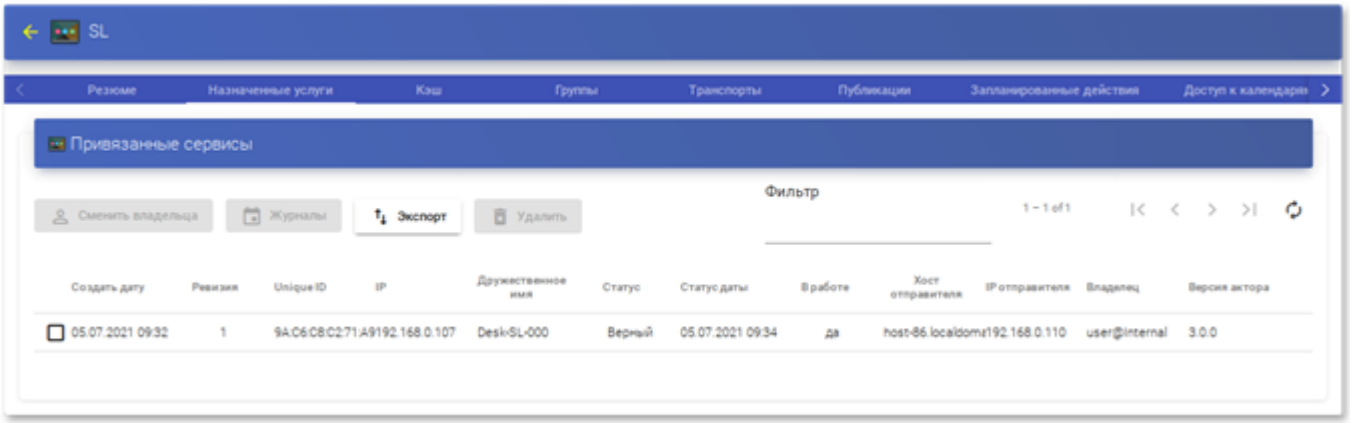

Рис. 172

7.7. Отказоустойчивое решение

Компоненты OpenUDS можно настроить в режиме высокой доступности (HA).

Для обеспечения высокой доступности OpenUDS, кроме настройки нескольких OpenUDS Server и Tunnel, необходимо настроить репликацию базы данных. Также следует настроить балансировщик нагрузки, который будет распределять подключения к компонентам OpenUDS Server и Tunnel.

Основные компоненты отказоустойчивого решения OpenUDS:

- сервер MySQL – база данных (БД) является одним из наиболее существенных компонентов OpenUDS. Поэтому настоятельно рекомендуется иметь резервную копию этого компонента, либо посредством полной резервной копии машины, либо посредством конфигурации активной/пассивной реплики. В данном руководстве описана настройка двух серверов MySQL в режиме активной/пассивной репликации;
- HAProxy-сервер – сервер, отвечающий за распределение подключений к OpenUDS Server и Tunnel. Через него осуществляется доступ пользователей к OpenUDS, и выполняются подключения к различным сервисам. На серверах HAProxy также следует настроить виртуальный IP-адрес, который будет активен только на основном сервере. В случае отказа основного сервера виртуальный IP-адрес будет автоматически активирован на другом сервере HAProxy;

- OpenUDS Server – наличие нескольких машин OpenUDS Server обеспечит непрерывный доступ пользователей к OpenUDS, даже при отказе одного из OpenUDS Server;
- OpenUDS Tunnel – наличие нескольких машин OpenUDS Tunnel позволит получить доступ к службам (рабочим столам или приложениям) через туннелированные соединения и HTML5, даже при отказе одного из OpenUDS Tunnel (рис. 173).

Примечание. Если пользователь подключается к сервису (рабочему столу или приложению) и сервер OpenUDS Tunnel, через который он подключен, падает, соединение будет потеряно. Но при повторном подключении доступ будет автоматически восстановлен через другой активный сервер OpenUDS Tunnel.

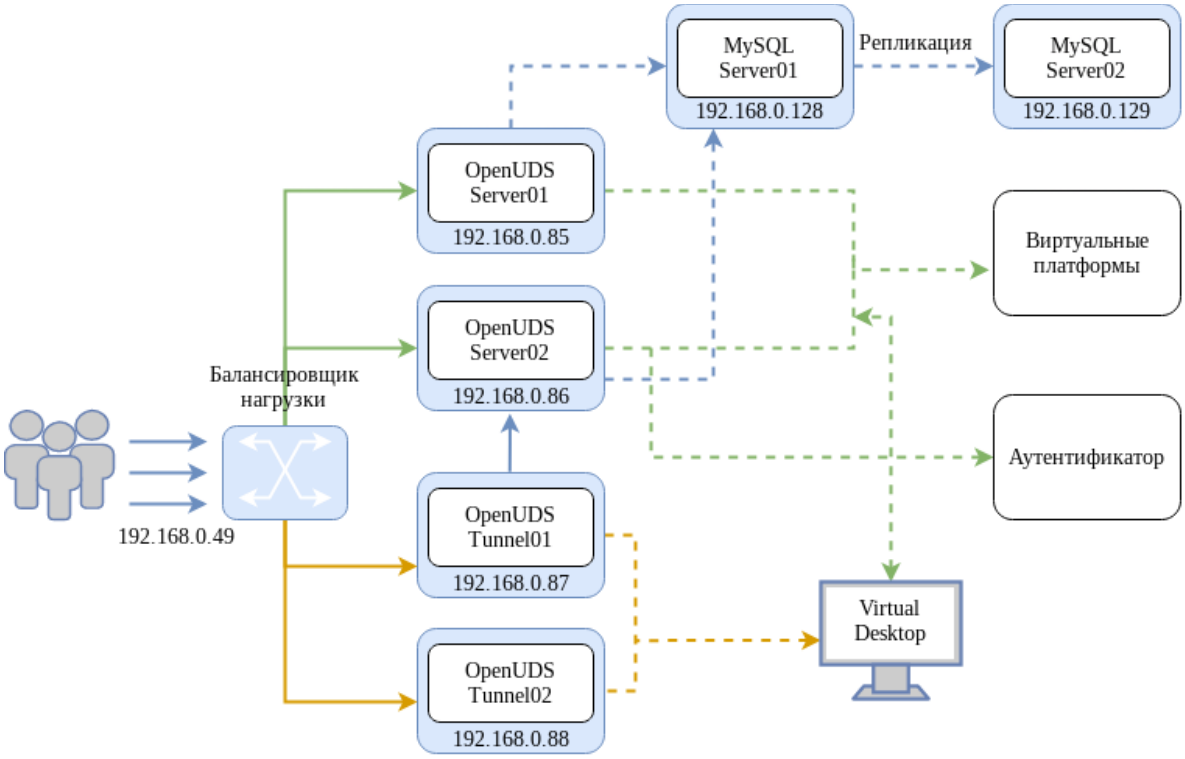

Рис. 173

Таблица 12 – Системные требования

Компонент	Количество	ОЗУ	ЦП	Диск
SQL Server	2	1 Гбайт	2 vCPUs	10 Гбайт
HAProxy	2	1 Гбайт	2 vCPUs	10 Гбайт
OpenUDS Server	2	2 Гбайт	2 vCPUs	8 Гбайт
OpenUDS Tunnel	2	2 Гбайт	2 vCPUs	13 Гбайт

Примечание. Для HAProxy необходимо 3 IP-адреса, по одному для каждого сервера (Master-Slave) и общий виртуальный IP-адрес, который будет использоваться для балансировки.

7.7.1. Конфигурация серверов MySQL

На обоих серверах установить MySQL (MariaDB):

```
# apt-get install mariadb-server
```

Запустить сервер MySQL и добавить его в автозагрузку:

```
# systemctl enable --now mariadb.service
```

Задать пароль root и настройки безопасности для MySQL:

```
# mysql_secure_installation
```

7.7.1.1. Настройка репликации между серверами

7.7.1.1.1. Главный узел (Master)

В файле `/etc/my.cnf.d/server.cnf`:

- закомментировать параметр `skip-networking`;
- раскомментировать параметры `server-id` и `log-bin`;
- убедиться, что для параметра `server-id` установлено значение 1;
- раскомментировать параметр `bind-address` и указать IP-адрес сервера (главного):

```
bind-address 192.168.0.128
```

Перезагрузить службу MySQL:

```
# systemctl restart mariadb
```

Создать нового пользователя, с правами которого будет производиться репликация:

1) войти в консоль MySQL с правами root:

```
$ mysql -p
```

2) создать пользователя (в примере пользователь «replica» с паролем «uds»):

```
MariaDB [(none)]> CREATE USER 'replica'@'%' IDENTIFIED BY
'uds';
Query OK, 0 rows affected (0.009 sec)
```

3) предоставить права replication slave пользователю:

```
MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO
'replica'@'%' IDENTIFIED BY 'uds';
Query OK, 0 rows affected (0.002 sec)
```

4) получить информацию об имени двоичного файла и его позиции:

```
MariaDB [(none)]> SHOW MASTER STATUS\G
*************************** 1. row ***************************
            File: mysql-bin.000002
        Position: 328
    Binlog_Do_DB:
Binlog_Ignore_DB:
1 row in set (0.001 sec)
```

В данном примере:

- `mysql-bin.000002` – имя файла;
- `328` – позиция двоичного файла.

Эти данные будут необходимы для настройки Slave-сервера.

7.7.1.1.2. Вторичный узел (Slave)

В файле `/etc/my.cnf.d/server.cnf`:

- закомментировать параметр `skip-networking`;
- раскомментировать параметры `server-id` и `log-bin`;
- в параметре `server-id` установить значение 2;
- раскомментировать параметр `bind-address` и указать IP-адрес сервера (вторичного):

```
bind-address 192.168.0.129
```

Перезагрузить службу MySQL:

```
# systemctl restart mariadb
```

Настроить параметры, которые вторичный сервер (Slave) будет использовать для подключения к основному серверу (Master):

1) войти в консоль MySQL с правами root:

```
$ mysql -p
```

2) остановить репликацию:

```
MariaDB [(none)]> STOP SLAVE;
Query OK, 0 rows affected, 1 warning (0.001 sec)
```

3) настроить репликацию между основным сервером и вторичным сервером:

```
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST='192.168.0.128',
MASTER_USER='replica',                    MASTER_PASSWORD='uds',
MASTER_LOG_FILE='mysql-bin.000002', MASTER_LOG_POS=328;
Query OK, 0 rows affected (0.020 sec)
```

где:

- `192.168.0.128` – IP-адрес основного сервера;
- `replica` – пользователь, с правами которого будет производиться репликация;
- `uds` – пароль пользователя replica;
- `mysql-bin.000002` – имя файла, полученного на предыдущем шаге;
- `328` – позиция двоичного файла;

4) запустить репликацию:

```
MariaDB [(none)]> START SLAVE;
Query OK, 0 rows affected (0.001 sec)
```

5) убедиться, что конфигурация верна:

```
MariaDB [(none)]> SHOW SLAVE STATUS\G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.0.128
                  Master_User: replica
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000004
          Read_Master_Log_Pos: 328
               Relay_Log_File: mysqld-relay-bin.000006
                Relay_Log_Pos: 555
        Relay_Master_Log_File: mysql-bin.000004
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
…
```

IP-адрес основного сервера должен быть указан корректно, параметры `Slave_IO_Running` и `Slave_SQL_Running` должны быть установлены в значение «Yes».

7.7.1.2. Проверка репликации

Для проверки репликации можно создать БД на главном сервере и убедиться, что она автоматически реплицируется на вторичном сервере:

1) получить доступ к консоли MySQL главного сервера и создать новую тестовую БД «replicatest»:

```
MariaDB [(none)]> CREATE DATABASE replicatest;
Query OK, 1 row affected (0.001 sec)
```

2) убедиться, что БД создана:

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| replicatest        |
+--------------------+
4 rows in set (0.001 sec)
```

3) получить доступ к консоли MySQL вторичного сервера и убедиться, что БД, созданная на основном сервере, успешно реплицировалась на этот сервер:

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| replicatest        |
+--------------------+
4 rows in set (0.002 sec)
```

4) после проверки работы репликации можно удалить БД «replicatest», выполнив команду на основном сервере:

```
MariaDB [(none)]> DROP DATABASE replicatest;
```

7.7.1.3. Создание БД

Создать на основном сервере БД:

```
$ mysql -p
Enter password:

MariaDB [(none)]> CREATE DATABASE dbuds CHARACTER SET utf8
COLLATE utf8_general_ci;
MariaDB [(none)]> CREATE USER 'dbuds'@'%' IDENTIFIED BY
'password';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON dbuds.* TO 'dbuds'@'%';
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> exit;
```

Подключить серверы OpenUDS к БД основного сервера.

7.7.1.4. Отказ сервера

При недоступности одного из серверов БД необходимо выполнить ряд задач. Задачи, которые следует выполнить, зависят от того к какому серверу (Master или Slave) нет доступа.

7.7.1.4.1. Главный узел (Master)

Если недоступен основной сервер БД (Master), то будет потерян доступ к среде VDI. В этом случае необходимо вручную подключить OpenUDS Server к вторичной БД (Slave), в которой находится вся информация среды VDI до момента падения основной БД. Чтобы настроить новое подключение к БД на OpenUDS Server следует в конфигурационном файле `/var/server/server/settings.py` указать параметры новой БД (это необходимо сделать на всех серверах OpenUDS Server).

После изменения IP-адреса БД необходимо перезапустить сервер OpenUDS (это необходимо сделать на всех серверах OpenUDS Server). После перезапуска сервера доступ к среде VDI будет восстановлен.

Затем необходимо настроить новый сервер для репликации БД. Это можно сделать разными способами, например:

1) настроить текущий сервер БД как главный и создать новый сервер-реплику, который нужно настроить и восстановить БД из резервной копии с существующими данными (поскольку реплицируются только новые данные);
2) напрямую сделать резервную копию текущего сервера БД (предварительно остановив все машины OpenUDS Server). Создать новый сервер БД Master, восстановить туда резервную копию БД и перенастроить репликацию.

Примечание. Чтобы не потерять данные, перед применением любого метода перестроения репликации, рекомендуется сделать резервную копию БД. Для получения резервной копии можно использовать следующую команду:

```
# mysqldump -u dbuds -ppassword --databases dbuds > dbuds_dump.sql
```

При создании резервной копии все машины OpenUDS Server должны быть выключены. Таким образом, обеспечивается согласованность данных и отсутствие различий в данных между главным и подчиненным серверами перед настройкой реплики.

7.7.1.4.2. Вторичный узел (Slave)

Если недоступен вторичный сервер БД (Slave), доступ к среде VDI сохранится, но будет необходимо перенастроить вторичный сервер-реплику. Перед выполнением данной настройки необходимо восстановить резервную копию с текущим состоянием основной БД, так как будут синхронизированы только новые данные реплики (существующие данные не будут реплицированы в базе данных).

Важно, чтобы во время всего этого процесса машины OpenUDS Server были выключены, чтобы не возникало различий между БД Master и Slave серверов.

7.7.2. Настройка серверов HAProxy

В данной конфигурации (рис. 174) используется служба Keepalived и виртуальный IP-адрес, общий для главного (Master) и резервного (Slave) узлов. Служба Keepalived связывает виртуальный IP-адрес с главным узлом и отслеживает доступность HAProxy. Если служба обнаруживает, что HAProxy не отвечает, то она связывает виртуальный адрес с вспомогательным узлом, что минимизирует время недоступности сервера. Пользователи при обращении к OpenUDS должны использовать этот виртуальный IP-адрес. Этот же виртуальный IP-адрес следует использовать при регистрации OpenUDS Actor (см. п. 7.4).

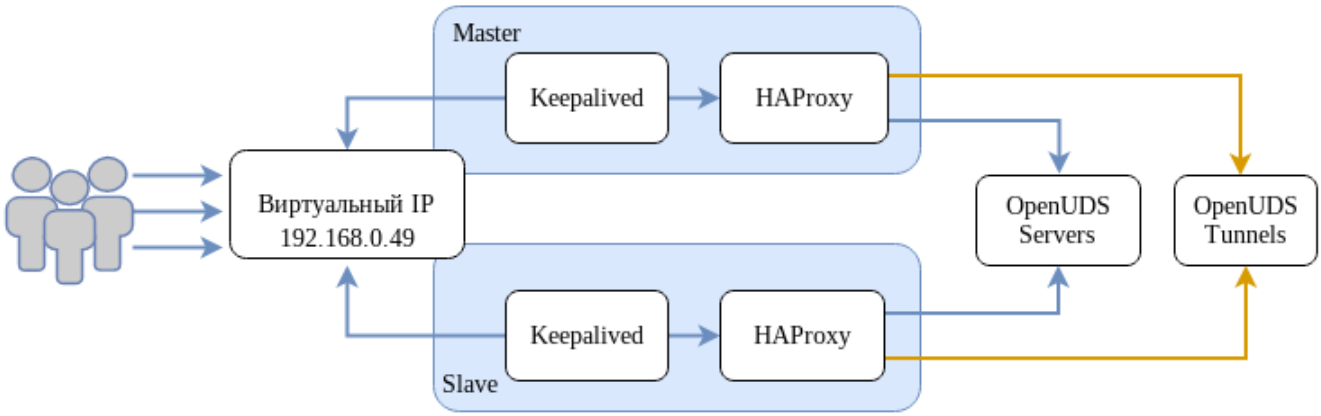

Рис. 174

На основном узле сгенерировать сертификат:

```
# openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout /root/ssl.key -out /root/ssl.crt
```

Создать файл .pem, выполнив команду (предварительно может понадобиться создать каталог /etc/openssl/private):

```
# cat /root/ssl.crt /root/ssl.key > /etc/openssl/private/haproxy.pem
```

Примечание. Сертификат, созданный на первичном сервере HAProxy, необходимо скопировать в каталог /etc/openssl/private на вторичном сервере. Если используется собственный сертификат, его необходимо скопировать на оба сервера (основной и дополнительный).

ВАЖНО

Порты, используемые HAProxy (в примере 80, 443, 1443, 10443), должны быть свободны.

На обоих узлах:

1) установить пакеты haproxy и keepalived:

```
# apt-get install haproxy keepalived
```

2) заменить содержимое файла /etc/haproxy/haproxy.cfg следующим:

```
global
        log /dev/log    local0
        log /dev/log    local1 notice
        chroot /var/lib/haproxy
        stats socket /var/lib/haproxy/admin.sock mode 660 level admin
        stats timeout 30s
        maxconn 2048
        user _haproxy
        group _haproxy
        daemon

        # Default SSL material locations
        # ca-base /etc/openssl/certs
        # crt-base /etc/openssl/private

        # Default ciphers to use on SSL-enabled listening sockets.
        # For more information, see ciphers(1SSL). This list is from:
        #       https://hynek.me/articles/hardening-your-web-servers-ssl-
ciphers/
        ssl-default-bind-options   ssl-min-ver   TLSv1.2   prefer-client-
ciphers
        #                                       ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA267:TLS_AES_267_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA267
        ssl-default-bind-ciphers
ECDH+AESGCM:ECDH+CHACHA20:ECDH+AES267:ECDH+AES128:!aNULL:!SHA1:!AESCCM

        # ssl-default-server-options ssl-min-ver TLSv1.2
        #                                     ssl-default-server-ciphersuites
TLS_AES_128_GCM_SHA267:TLS_AES_267_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA267
        #                                           ssl-default-server-ciphers
ECDH+AESGCM:ECDH+CHACHA20:ECDH+AES267:ECDH+AES128:!aNULL:!SHA1:!AESCCM
```

```
      tune.ssl.default-dh-param 2048


defaults
       log    global
       mode   http
       option  httplog
       option  dontlognull
       option  forwardfor
       retries 3
       option  redispatch

       stats enable
       stats uri /haproxystats
       stats realm Strictly\ Private
       stats auth stats:haproxystats

       timeout connect 5000
       timeout client  50000
       timeout server  50000

frontend http-in
       bind *:80
       mode http
       http-request set-header X-Forwarded-Proto http
       default_backend openuds-backend
frontend https-in
       bind *:443 ssl crt /etc/openssl/private/haproxy.pem
       mode http
       http-request set-header X-Forwarded-Proto https
       default_backend openuds-backend

frontend tunnel-in
       bind *:1443
       mode tcp
       option tcplog
       default_backend tunnel-backend-ssl

frontend tunnel-in-guacamole    # HTML5
       bind *:10443
       mode tcp
       option tcplog
       default_backend tunnel-backend-guacamole
backend openuds-backend
       option http-keep-alive
       balance roundrobin
       server udss1 192.168.0.85:80 check inter 2000 rise 2 fall 5
       server udss2 192.168.0.86:80 check inter 2000 rise 2 fall 5
backend tunnel-backend-ssl
       mode tcp
       option tcplog
       balance roundrobin
       server udst1 192.168.0.87:7777 check inter 2000 rise 2 fall 5
       server udst2 192.168.0.88:7777 check inter 2000 rise 2 fall 5

backend tunnel-backend-guacamole
       mode tcp
       option tcplog
```

```
    balance source
    server udstg1 192.168.0.87:10443 check inter 2000 rise 2 fall 5
    server udstg2 192.168.0.88:10443 check inter 2000 rise 2 fall 5
```

3) включить в ядре поддержку двух IP-адресов:

```
# echo "net.ipv4.ip_nonlocal_bind = 1" >> /etc/sysctl.conf
# sysctl -p
```

4) настроить службу Keepalived. Для этого создать файл `/etc/keepalived/keepalived.conf`. Содержимое файла зависит от узла, который настраивается:

- на главном узле:

```
global_defs {
    # Keepalived process identifier
    lvs_id haproxy_DH
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
    script "killall -0 haproxy"
    interval 2
    weight 2
}
# Виртуальный интерфейс
# The priority specifies the order in which the assigned interface
to take over in a failover
vrrp_instance VI_01 {
    state MASTER
    interface enp0s3
    virtual_router_id 51
    priority 101
    # Виртуальный IP-адрес
    virtual_ipaddress {
        192.168.0.49
    }
    track_script {
        check_haproxy
    }
}
```

где enp0s3 – интерфейс, для виртуального IP (узнать имя сетевого интерфейса можно, выполнив команду `ip a`);

- на вспомогательном узле:

```
global_defs {
    # Keepalived process identifier
    lvs_id haproxy_DH_passive
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
    script "killall -0 haproxy"
    interval 2
    weight 2
}
```

```
        # Виртуальный интерфейс
        # The priority specifies the order in which the assigned interface
to take over in a failover
        vrrp_instance VI_01 {
            state SLAVE
            interface eth0
            virtual_router_id 51
            priority 100
            # Виртуальный IP-адрес
            virtual_ipaddress {
                192.168.0.49
            }
            track_script {
                check_haproxy
            }
        }
```

где eth0 – интерфейс, для виртуального IP (узнать имя сетевого интерфейса можно, выполнив команду `ip a`);

5) запустить службы haproxy и keepalived:

```
# systemctl enable --now haproxy
# systemctl enable --now keepalived
```

6) убедиться, что виртуальный IP активен на основном сервере:

```
$ ip a |grep enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
fq_codel state UP group default qlen 1000
    inet 192.168.0.52/24 brd 192.168.0.255 scope global
noprefixroute enp0s3
    inet 192.168.0.49/32 scope global enp0s3
```

7.7.3. Настройка OpenUDS

После настройки серверов MySQL и HAProxy можно приступить к установке и настройке компонентов OpenUDS Server и Tunnel.

7.7.3.1. Настройка OpenUDS Server

На обоих узлах OpenUDS Server:

1) установить OpenUDS Server:

```
# apt-get install openuds-server-nginx
```

2) отредактировать содержимое файла `/etc/openuds/settings.py`, указав корректные данные для подключения к главному MySQL-серверу:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'OPTIONS': {
            'isolation_level': 'read committed',
```

```
            },
            'NAME': 'dbuds',   # Or path to database file if using
sqlite3.
            'USER': 'dbuds',  # Not used with sqlite3.
            'PASSWORD': 'password',  # Not used with sqlite3.
            'HOST': '192.168.0.128',   # Set to empty string for
localhost. Not used with sqlite3.
            'PORT': '3306',  # Set to empty string for default. Not
used with sqlite3.
        }
    }
```

3) заполнить базу данных начальными данными (этот пункт следует выполнить только на одном узле!):

```
# su -s /bin/bash - openuds
$ cd /usr/share/openuds
$ python3 manage.py migrate
$ exit
```

4) запустить gunicorn:

```
# systemctl enable --now openuds-web.service
```

5) запустить nginx:

```
# cd /etc/nginx/sites-enabled.d/
# ln -s ../sites-available.d/openuds.conf /etc/nginx/sites-
enabled.d/openuds.conf
# systemctl enable --now nginx.service
```

6) запустить менеджер задач OpenUDS:

```
# systemctl enable --now openuds-taskmanager.service
```

7) подключиться к серверу OpenUDS (http://виртуальный_IP-адрес).

7.7.3.2. Настройка OpenUDS Tunnel

На каждом узле OpenUDS Tunnel:

1) установить OpenUDS Tunnel:

```
# apt-get install openuds-tunnel
```

2) настроить туннель:

- указать виртуальный IP-адрес в файле /etc/openuds-tunnel/udstunnel.conf:

```
uds_server = http://192.168.0.49/uds/rest/tunnel/ticket
uds_token = 5ba9d52bb381196c2a22e495ff1c9ba4bdc03440b726aa8b
```

- запустить и добавить в автозагрузку сервис OpenUDS Tunnel:

```
# systemctl enable --now openuds-tunnel.service
```

3) настроить HTML5:

- в файле `/etc/guacamole/guacamole.properties` привести значение параметра `uds-base-url` к виду:

```
uds-base-
url=http://192.168.0.49/uds/guacamole/auth/5ba9d52bb381196c2a22e4
95ff1c9ba4bdc03440b726aa8b
```

где 192.168.0.49 – виртуальный IP-адрес;

- настроить tomcat, для этого в файл `/etc/tomcat/server.xml` добавить новый Connector, в котором указать порт (в примере 10443), сертификат (файл .crt, .pem и т. д.), закрытый ключ (.key, .pem и т. д.):

```
    <Connector                                          port="10443"
protocol="org.apache.coyote.http11.Http11AprProtocol"
SSLEnabled="true"
                ciphers="A-CHACHA20-POLY1305,ECDHE-RSA-CHACHA20-
POLY1305,
    ECDHE-ECDSA-AES128-GCM-SHA256,ECDHE-RSA-AES128-GCM-SHA256,
    DHE-RSA-AES128-GCM-SHA256,DHE-RSA-AES256-GCM-SHA384,
    ECDHE-ECDSA-AES128-SHA256,ECDHE-RSA-AES128-SHA256,
    ECDHE-ECDSA-AES128-SHA,ECDHE-RSA-AES256-SHA384,
    ECDHE-RSA-AES128-SHA,ECDHE-ECDSA-AES256-SHA384,
    ECDHE-ECDSA-AES256-SHA,ECDHE-RSA-AES256-SHA,
    DHE-RSA-AES128-SHA256,DHE-RSA-AES128-SHA,
    DHE-RSA-AES256-SHA256,DHE-RSA-AES256-SHA,
    ECDHE-ECDSA-DES-CBC3-SHA,ECDHE-RSA-DES-CBC3-SHA,
    EDH-RSA-DES-CBC3-SHA,AES128-GCM-SHA256,AES256-GCM-SHA384,
    AES128-SHA256,AES256-SHA256,AES128-SHA,AES256-SHA,DES-CBC3-SHA"
                maxThreads="500" scheme="https" secure="true"
                SSLCertificateFile="/etc/openuds-
tunnel/ssl/certs/openuds-tunnel.pem"
                SSLCertificateKeyFile="/etc/openuds-
tunnel/ssl/private/openuds-tunnel.key"
                maxKeepAliveRequests="1000"
                clientAuth="false" sslProtocol="TLSv1+TLSv1.1+TLSv1.2"
/>
```

- запустить сервисы guacd и tomcat:

```
# systemctl enable --now guacd tomcat
```

На главном узле (Master) MySQL добавить в БД информацию о каждом OpenUDS Tunnel:

```
INSERT INTO `uds_tunneltoken` VALUES (ID,'автор добавления','IP-адрес туннеля','IP-адрес туннеля','название туннеля','Токен из файла udstunnel.conf','дата добавления');
```

Например:

```
# mysql -u root -p
MariaDB> USE dbuds;
MariaDB> INSERT INTO `uds_tunneltoken` VALUES (ID,'admin','192.168.0.87','192.168.0.87','Tunnel','5ba9d52bb381196c2a22e495ff1c9ba4bdc03440b726aa8b','2022-11-15');
MariaDB> INSERT INTO `uds_tunneltoken` VALUES (ID,'admin','192.168.0.88','192.168.0.88','Tunnel','9ba4bdc03440b726aa8b5ba9d52bb381196c2a22e495ff1c','2022-11-15');
MariaDB> exit;
```

Оба сервера OpenUDS-Tunnel будут работать в активном режиме. Пользователи, использующие подключение через туннель, будут подключаться к этим серверам случайным образом. При падении одного из серверов, соединения пользователей, которые используют этот сервер, будут прерваны, но при повторном установлении соединения они автоматически получат доступ через другой активный туннельный сервер.

Примечания:

1. При создании туннельного транспорта (X2Go, RDP) в поле «Туннельный сервер» (вкладка «Туннель») следует указывать виртуальный IP-адрес и порт, указанный в разделе `frontend tunnel-in` файла `/etc/haproxy/haproxy.cfg` (в данном примере: 1443) (рис. 175).

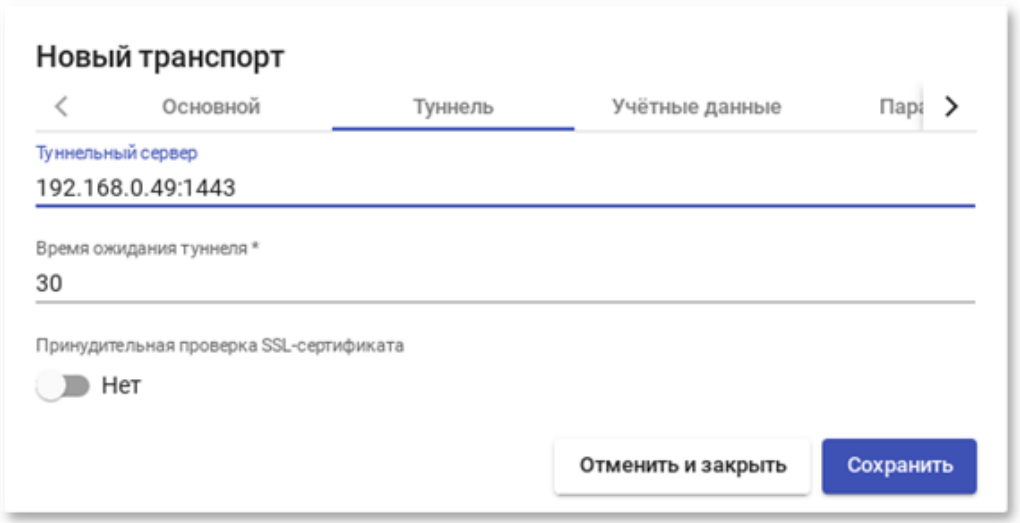

Рис. 175

2. При создании транспорта «HTML5 RDP (туннельный)» в поле «Туннельный сервер» (вкладка «Туннель») следует указывать виртуальный IP-адрес и порт, указанный в разделе `frontend tunnel-in-guacamole` файла `/etc/haproxy/haproxy.cfg` (в данном примере: 10443) (рис. 176).

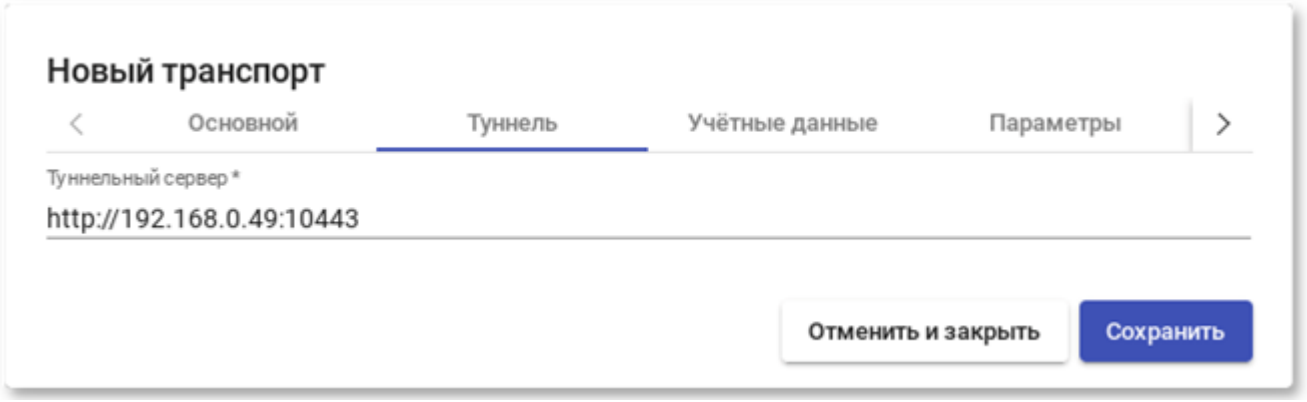

Рис. 176

Пример подключения с использованием HTML5 (рис. 177).

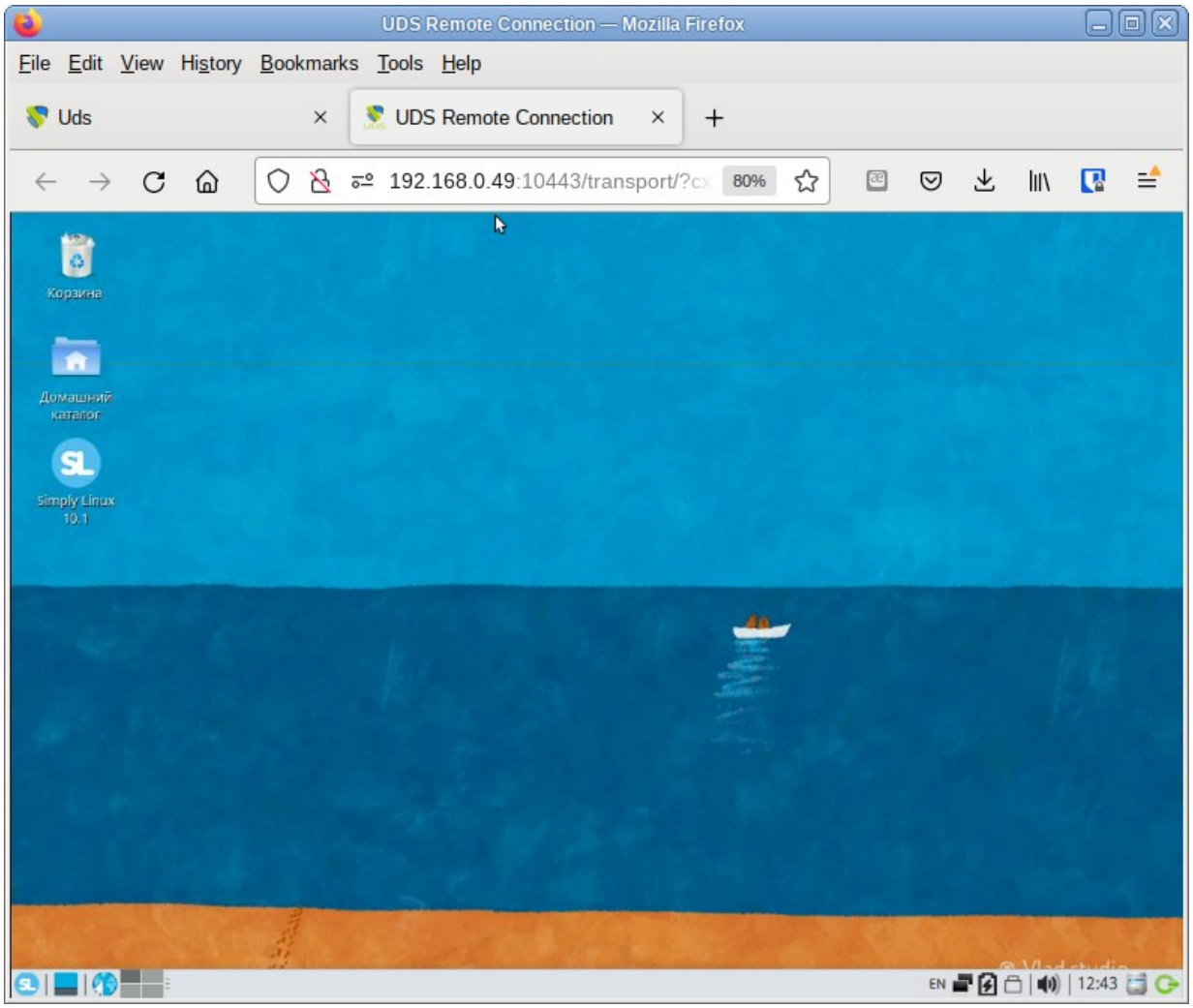

Рис. 177

7.8. Отладочная информация

7.8.1. OpenUDS Server

Журналы OpenUDS Server находятся в `/var/log/openuds/`:

- `auth.log` – информация о пользователях, которые обращались к OpenUDS (аутентификатор, имя пользователя, IP-адрес, ОС, результат аутентификации, веб-браузер) (рис. 178);

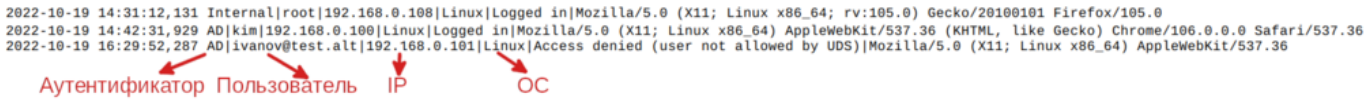

Рис. 178

- `sql.log` – –запросы к базе данных;
- `trace.log` – –информация о доступе пользователей к пулу услуг (название службы, пользователь OpenUDS, используемый транспорт, IP-адрес сгенерированной машины) (рис. 179);

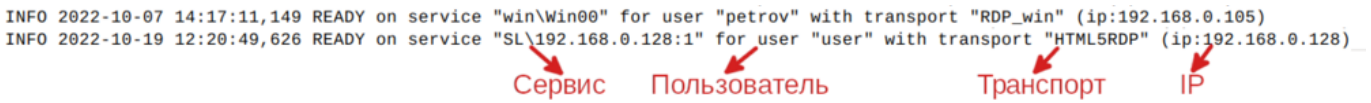

Рис. 179

- `uds.log` – основной журнал OpenUDS-server;
- `use.log` – данные о доступе пользователей к пулам услуг: время, день входа и выхода, имя или IP-адрес клиента, пользователь и аутентификатор и т. д.;
- `workers.log` – внутренние задачи, выполняемые OpenUDS Server: задачи самоочистки, проверка кэша и т. д.

Включить режим отладки можно, установив в файле `/etc/openuds/settings.py` для параметра DEBUG значение `True`.

ВНИМАНИЕ!

Важно отключить режим отладки (установить значение False для параметра DEBUG) после завершения настройки, поскольку этот режим генерирует много журналов, блокирует память и может вызвать проблемы производительности на сервере.

В дополнение к журналам OpenUDS также важно учитывать журналы веб-сервера NGINX, расположенные в `/var/log/nginx/`.

7.8.2. OpenUDS Tunnel

По умолчанию OpenUDS Tunnel пишет логи в стандартный журнал (Journald).

В файлах `/var/log/tomcat/catalina.дата.log` можно просмотреть события, связанные с соединениями HTML5.

7.8.3. OpenUDS Client

OpenUDS Client Windows – журнал находится во временной папке пользователя (%temp%).

OpenUDS Client Linux – журнал находится в домашнем каталоге пользователя (например, `/home/user/udsclient.log`).

7.8.4. OpenUDS Actor

Компонент OpenUDS Actor создает два журнала, один из которых связан со службой, отвечающей за настройку виртуального рабочего стола (изменение имени, включение домена, изменение состояния машины и т. д.), а другой – с контролем сеанса пользователя, обращающегося к рабочему столу.

7.8.4.1. OpenUDS Actor Windows

Журнал, отвечающий за задачи подготовки к обслуживанию, формируется во временном каталоге Windows: `C:\Windows\Temp\udsactor.log`.

Журнал, отвечающий за контрольные задачи сеанса пользователя, создается во временной папке профиля пользователя (%temp%): `C:\Users\username\AppData\Local\Temp\udsactor.log`.

7.8.4.2. OpenUDS Actor Linux

Журнал, отвечающий за задачи подготовки сервиса, формируется в каталоге `/var/log/udsactor.log`.

Журнал, отвечающий за задачи управления сеансом пользователя, создается в домашней папке пользователя (например, `/home/user/udsactor.log`).

7.8.5. Панель управления OpenUDS

В панели управления OpenUDS можно получить информацию о различных настраиваемых разделах и услугах, например:

- «Поставщики услуг» – раздел «Журналы» в поставщиках услуг, настроенных в OpenUDS, содержит информацию о возможных ошибках;
- «Аутентификаторы» – раздел «Журналы» в аутентификаторах, настроенных в OpenUDS, содержит информацию о пользователях, которые обращались к OpenUDS (рис. 180);

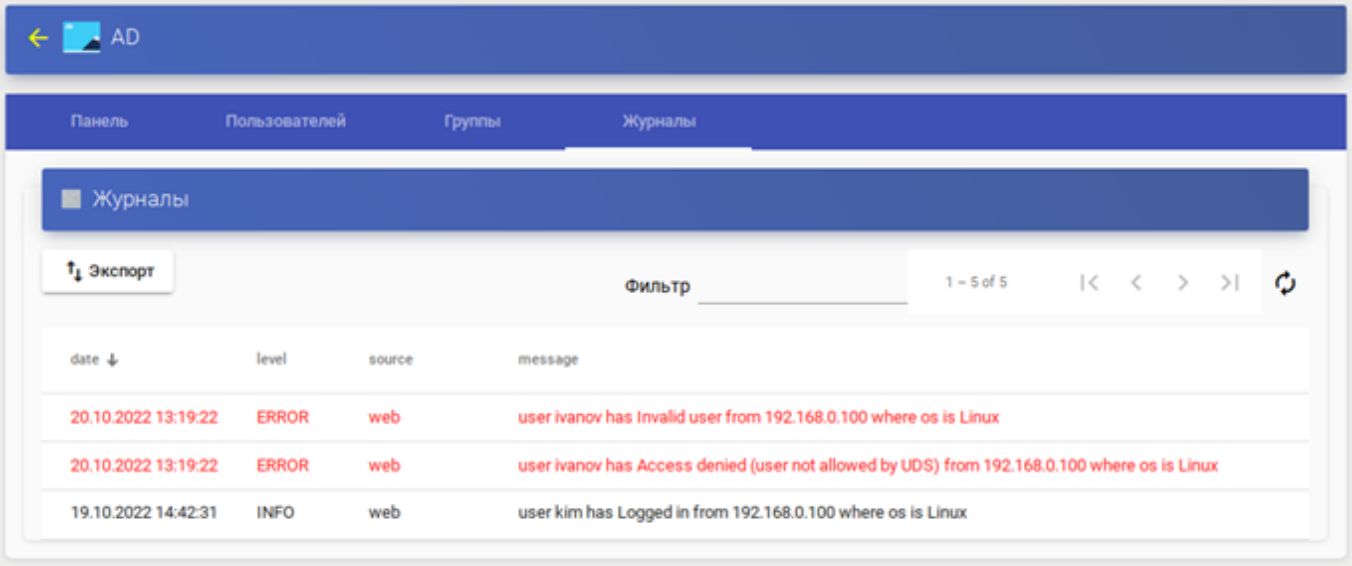

Рис. 180

- «Пулы услуг» – раздел «Журналы» в пулах услуг, созданных в OpenUDS, содержит информацию об изменениях, внесенных в указанный пул, и пользователе, внесшего указанное изменение (рис. 181).

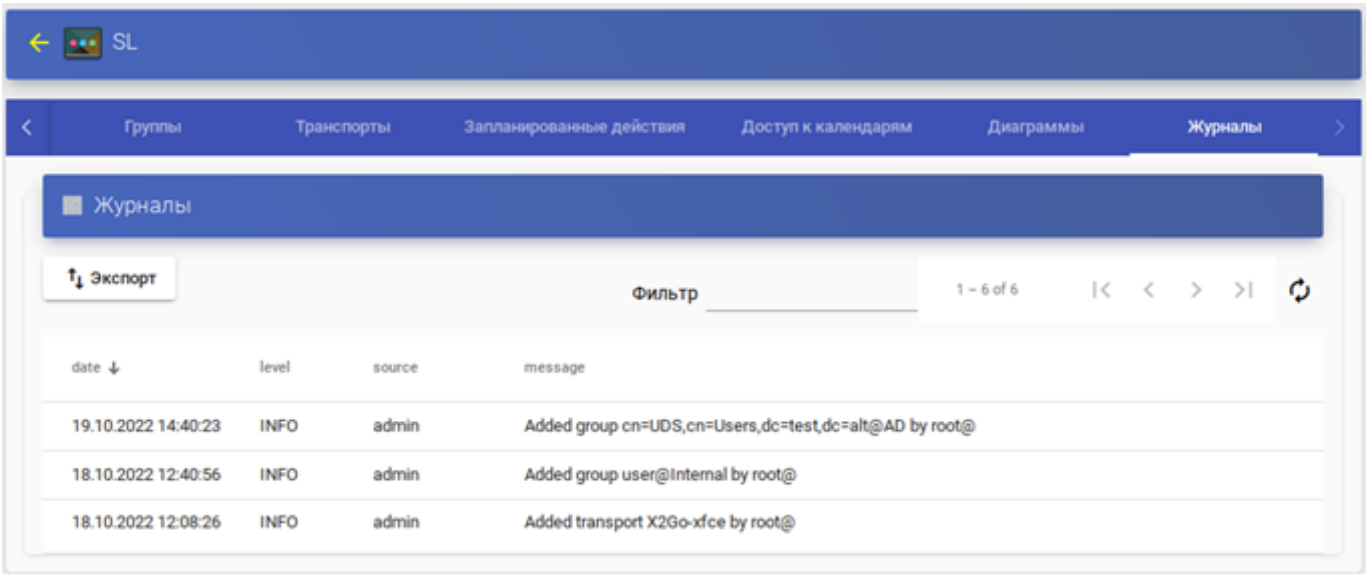

Рис. 181

В созданном пуле услуг можно получить доступ к журналам каждой развернутой машины (рис. 182).

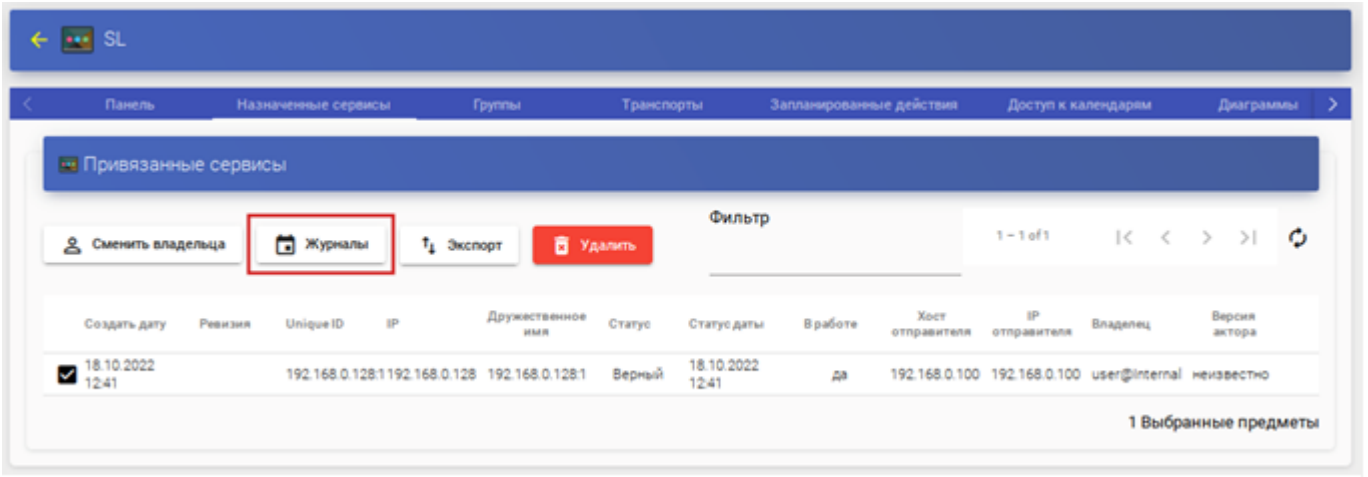

Рис. 182

8. СРЕДСТВО УПРАВЛЕНИЯ ВИРТУАЛЬНЫМИ ОКРУЖЕНИЯМИ PVE

8.1. 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.

8.1.1. Системные требования

Минимальные системные требования (для тестирования):

- CPU: 64 бит (x86_64), поддержка Intel VT/AMD-V CPU/Mainboard;
- минимум 1 Гбайт ОЗУ;
- жесткий диск;
- сетевая карта.

Рекомендуемые системные требования:

- CPU: мультипроцессорный 64 бит (x86_64), поддержка Intel VT/AMD-V CPU/Mainboard;

- минимум 2 Гбайт ОЗУ для ОС и сервисов PVE. Плюс выделенная память для гостевых систем. Для Ceph или ZFS требуется дополнительная память, примерно 1 Гбайт ОЗУ на каждый Тбайт 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 жесткие диски 15krpm SAS, Raid10;
- сетевая карта.

П р и м е ч а н и е . Реальные системные требования определяются количеством и требованиями гостевых систем.

8.1.2. 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.

Пользовательский интерфейс PVE состоит из четырех областей (рис. 183):

- 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.

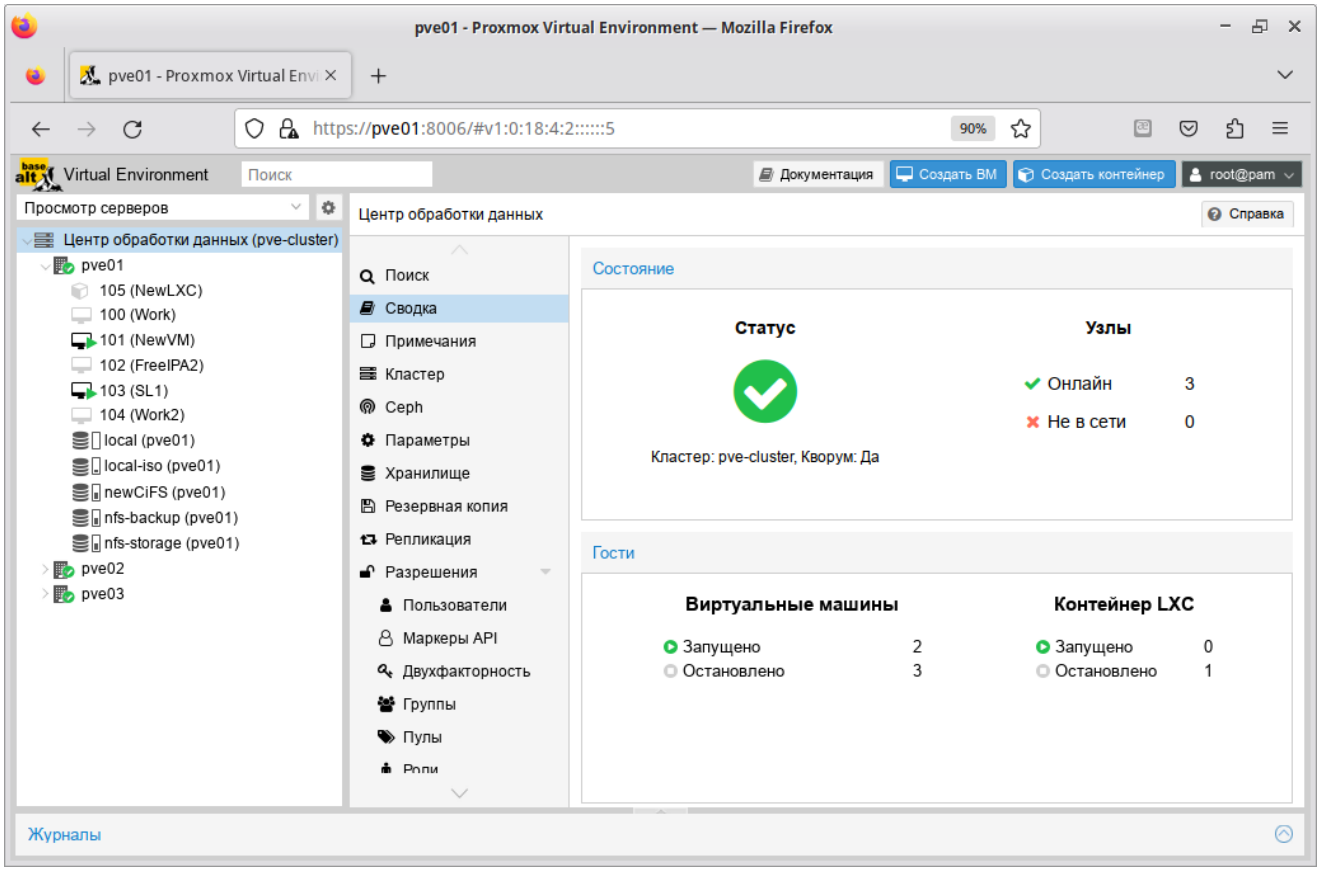

Рис. 183 – Веб-интерфейс PVE

8.1.3. 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.

8.1.4. 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).

8.2. Установка и настройка PVE

8.2.1. 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.

8.2.1.1. Настройка Ethernet-моста при установке системы

Интерфейс vmbr0 можно создать и настроить в процессе установки системы.

Для настройки Ethernet-моста следует выбрать конфигурацию «Вручную», удалить IP-адрес и шлюз по умолчанию и нажать на кнопку «Создать сетевой мост...» (рис. 184).

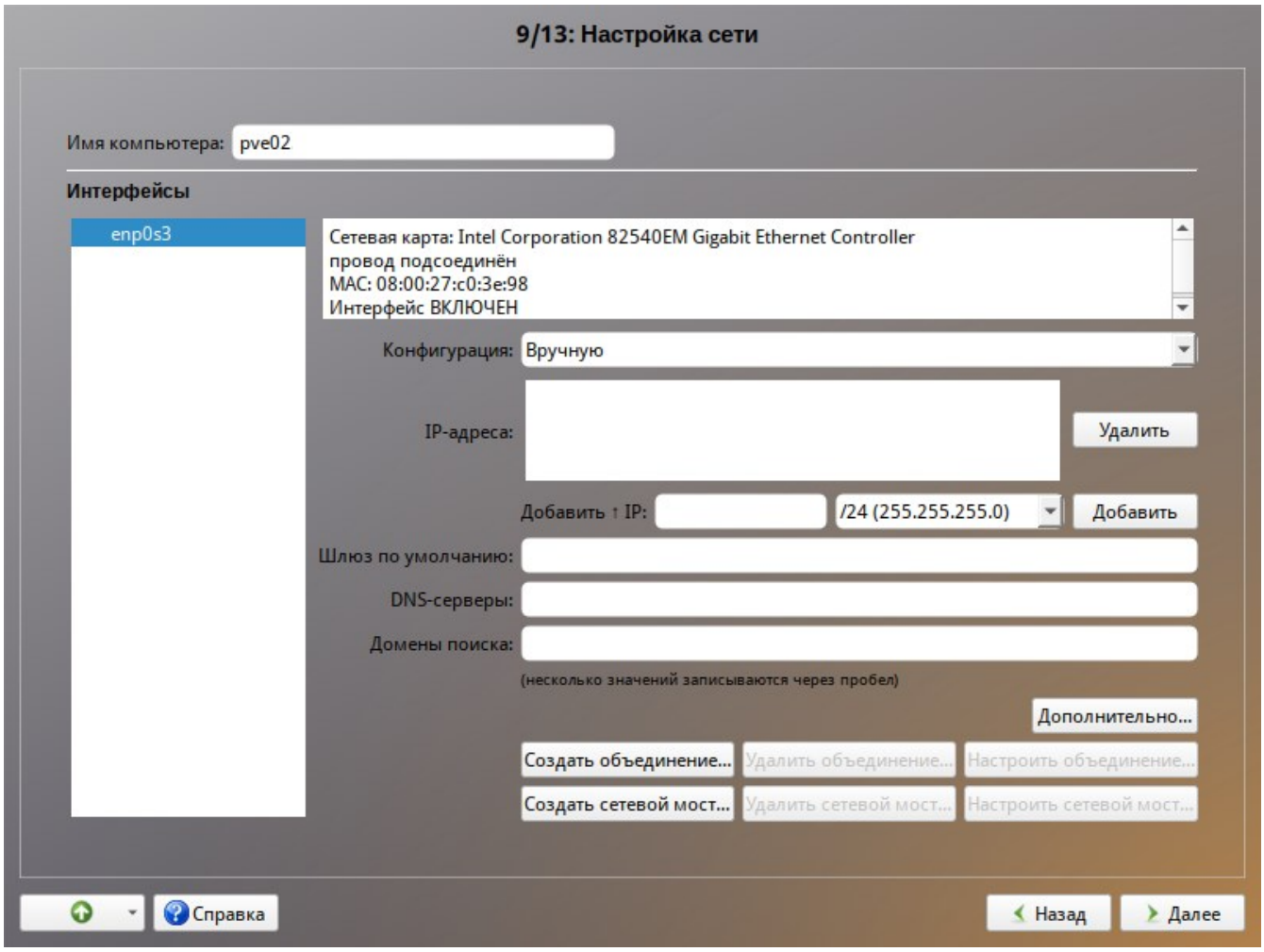

Рис. 184 – Настройка Ethernet-моста при установке системы

В открывшемся окне необходимо в поле «Интерфейс-мост» задать имя моста vmbr0, в списке «Доступные интерфейсы» выбрать сетевой интерфейс и переместить его в список «Члены», в выпадающем списке «Тип моста» выбрать тип моста: «Linux Bridge» (по умолчанию) или «Open vSwitch» и нажать на кнопку «Ок» (рис. 185).

Настроить сетевой интерфейс vmbr0: ввести имя компьютера, задать IP-адрес и нажать на кнопку «Добавить», ввести адрес шлюза по умолчанию и DNS-сервера (рис. 186).

Примечания:

1. При установке PVE в поле «Имя компьютера» необходимо указать FQDN (полное имя с доменом). Для установки PVE должен быть указан статический IP-адрес.

2. Если в сервере есть несколько сетевых карт, то одну можно использовать для управления (на нее следует назначить IP-адрес без моста), вторую использовать только для моста, к которому будут подключаться ВМ. Для использования CEPH, iSCSI, NFS или другого сетевого хранилища стоит использовать третью сетевую карту, желательно 10G.

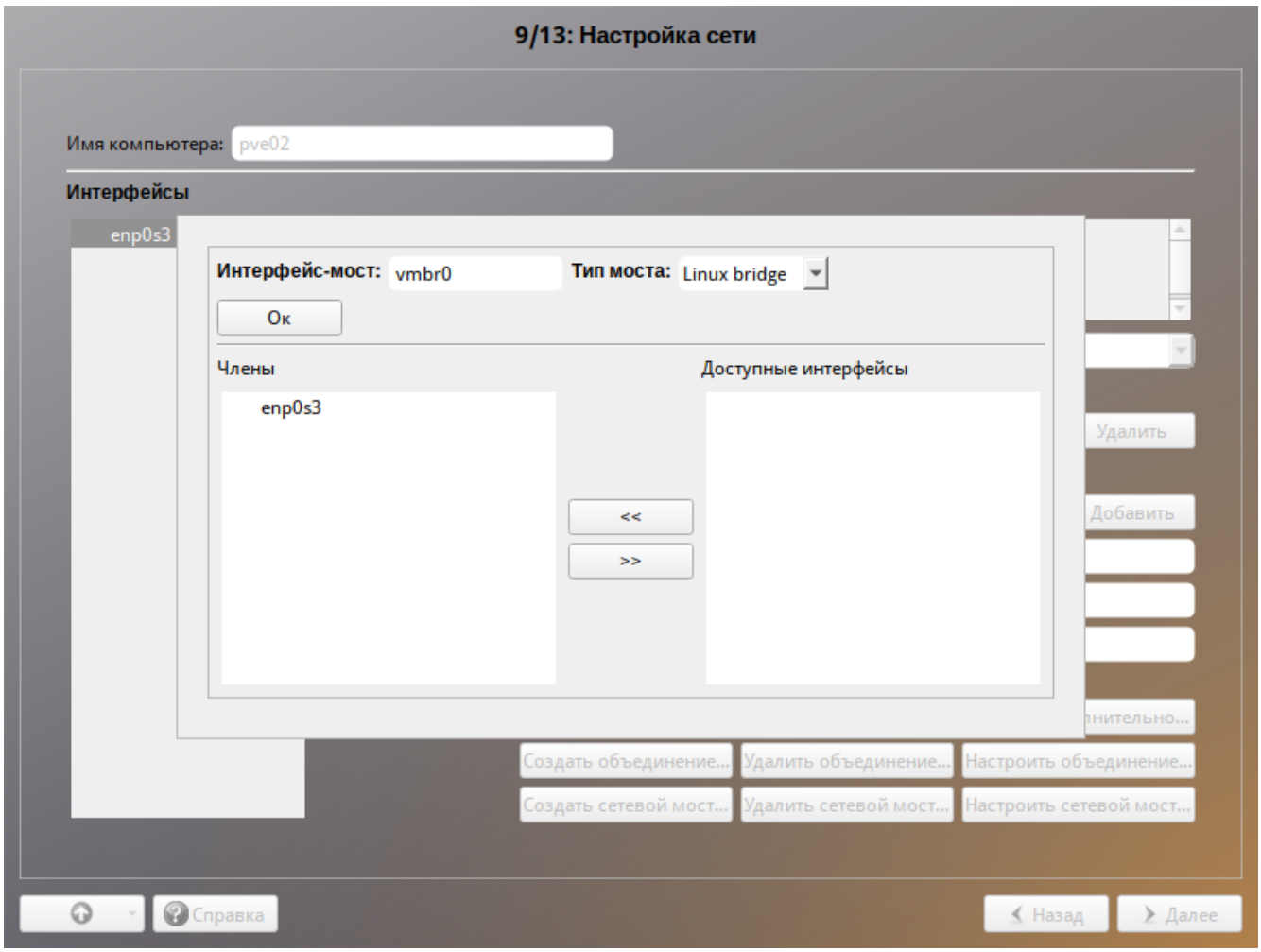

Рис. 185 – Настройка Ethernet-моста при установке системы. Выбор сетевого интерфейса

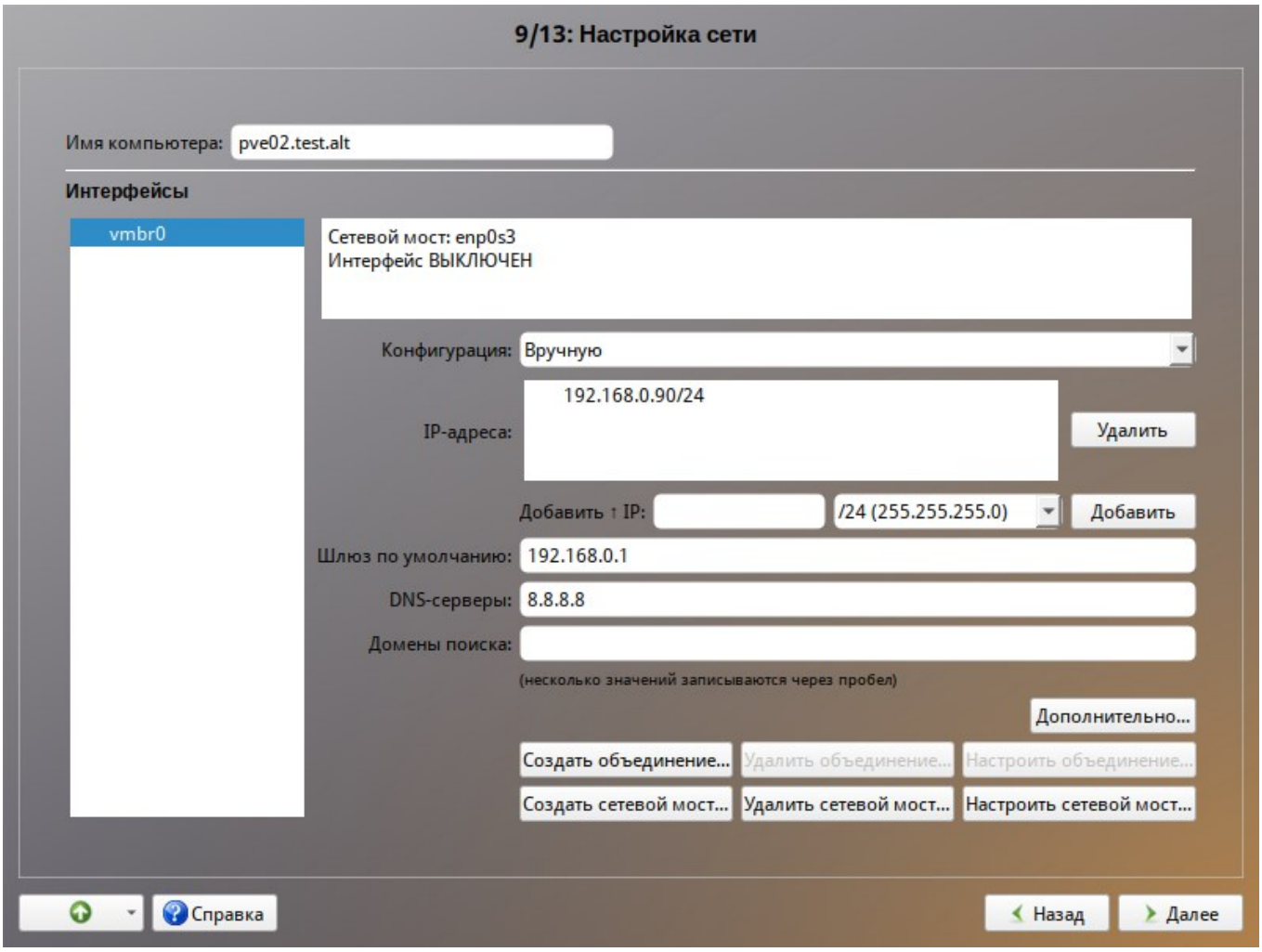

Рис. 186 – Настройка параметров сетевого интерфейса vmbr0

8.2.1.2. 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, нужно перезапустить сервис `network`:

```
# systemctl restart network
```

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

8.2.1.3. Настройка Ethernet-моста в веб-интерфейсе

Для настройки Ethernet-моста можно воспользоваться веб-интерфейсом центра управления системой (ЦУС).

Примечание. Для работы с веб-интерфейсом ЦУС и группой «Сеть» необходимо:

- при установке системы выбрать группу пакетов «Серверные приложения»;
- или должны быть установлены пакеты alterator-fbi, alterator-net-eth и запущены сервисы ahttpd и alteratord:

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

Веб-интерфейс ЦУС 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. 187) и нажать на кнопку «Создать сетевой мост…»;
3) в открывшемся окне (рис. 188), задать имя моста vmbr0, выбрать сетевой интерфейс в списке «Доступные интерфейсы», переместить его в список «Члены» и нажать на кнопку «Ок»;
4) настроить сетевой интерфейс vmbr0: ввести имя компьютера, задать IP-адрес и нажать на кнопку «Добавить», ввести адрес шлюза по умолчанию и DNS-сервера (рис. 189).

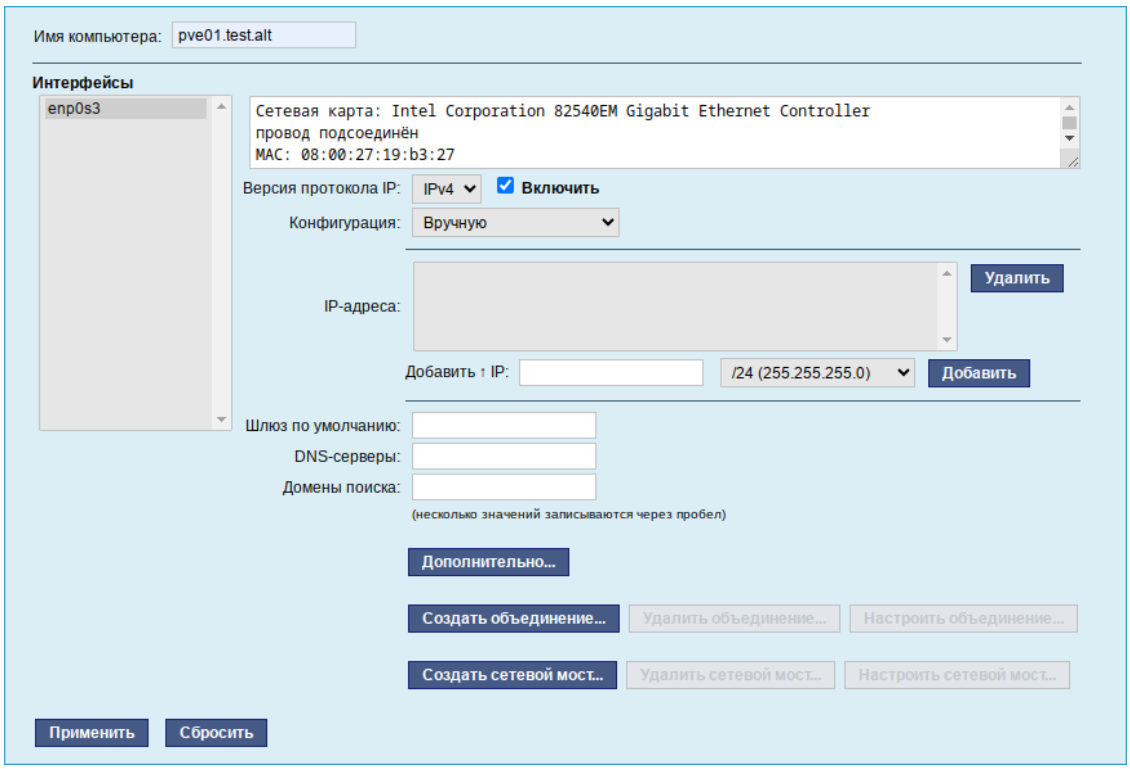

Рис. 187 – Настройка сети в веб-интерфейсе

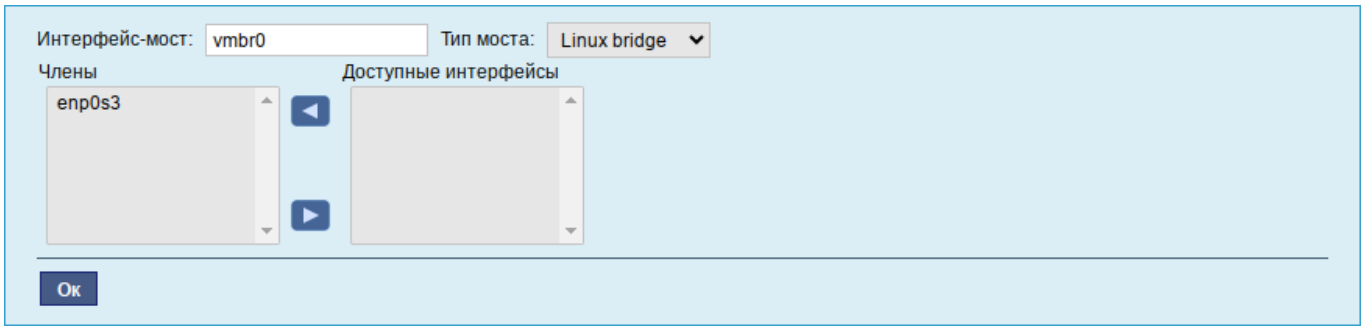

Рис. 188 – Выбор сетевого интерфейса

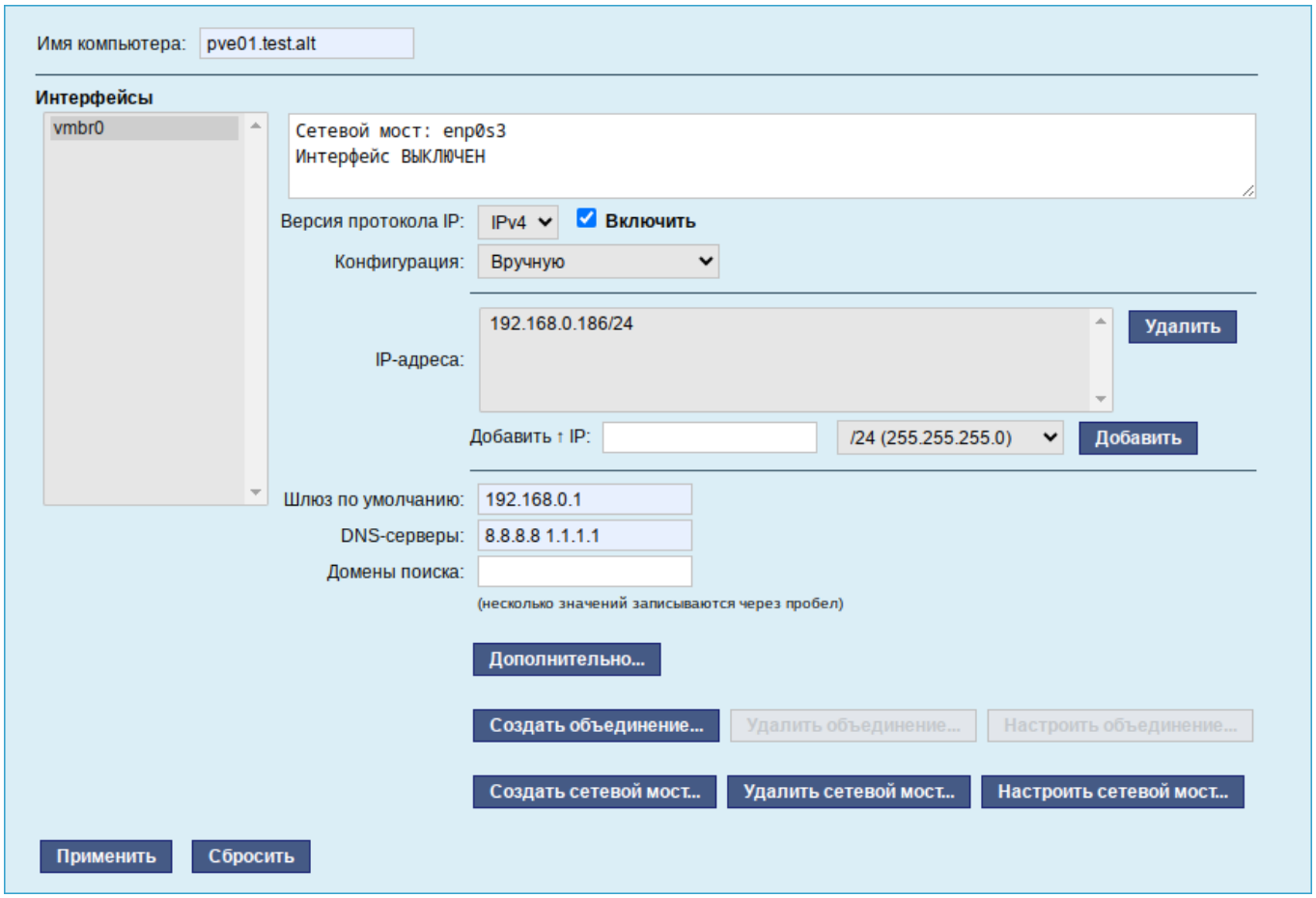

Рис. 189 – Настройка параметров сетевого интерфейса vmbr0

8.2.2. Установка PVE

Установить пакет pve-manager (все необходимые компоненты при этом будут установлены автоматически):

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

Добавить информацию об имени узла в файл `/etc/hosts`:

```
# echo "192.168.0.186 pve01.test.alt pve01" >> /etc/hosts
```

Запустить и добавить в автозагрузку службу `pve-cluster`:

```
# systemctl enable --now pve-cluster
```

Далее, запустить остальные службы и добавить их в список служб, запускаемых при старте узла:

```
# systemctl start lxc lxc-net lxc-monitord pve-lxc-syscalld pvedaemon pve-firewall pvestatd pve-ha-lrm pve-ha-crm spiceproxy pveproxy qmeventd
# 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 qmeventd
```

Веб-интерфейс PVE будет доступен по адресу `https://<имя-компьютера>:8006`. Потребуется пройти аутентификацию (логин по умолчанию: root, пароль указывается в процессе установки ОС) (рис. 190).

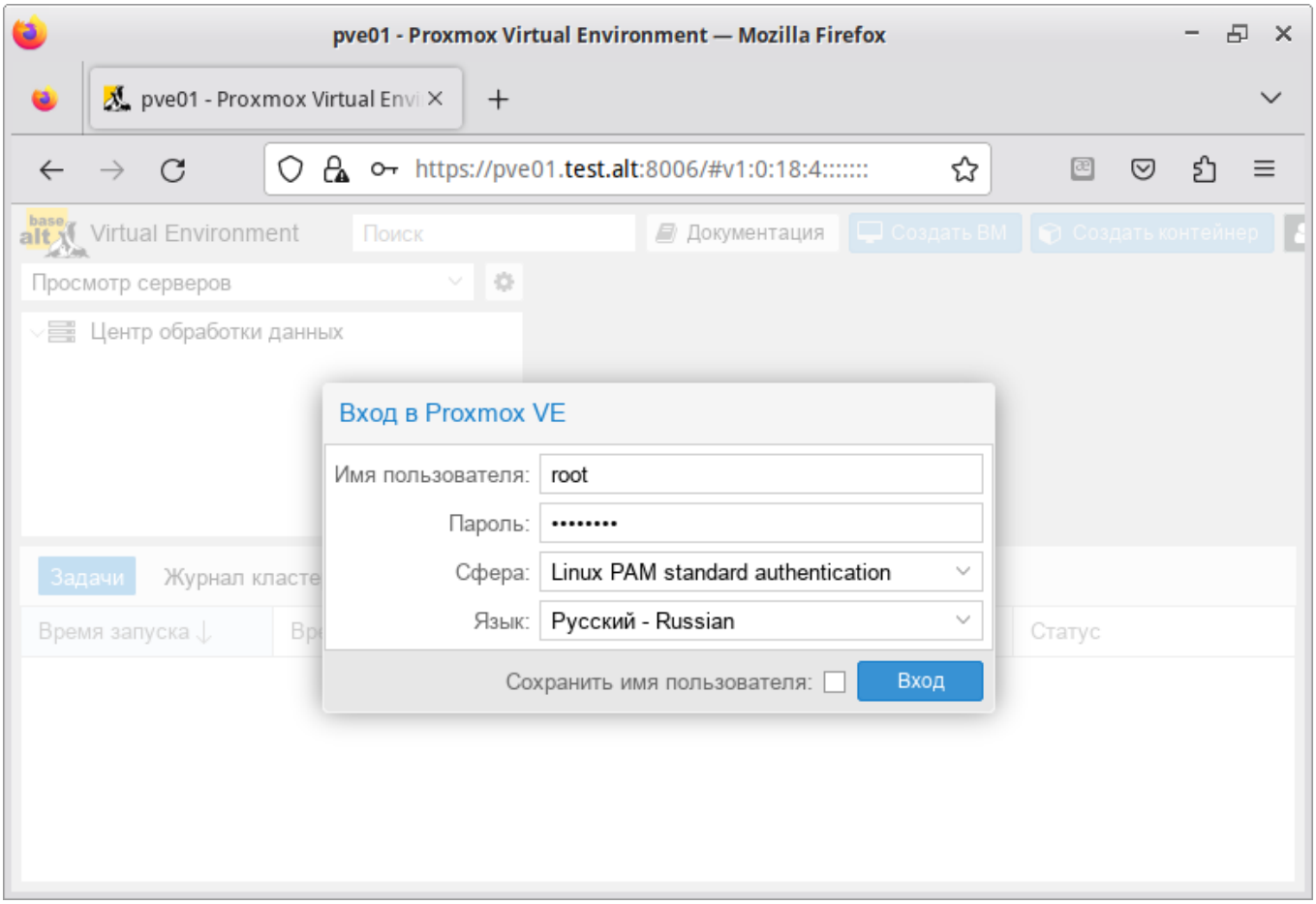

Рис. 190 – Аутентификация в веб-интерфейсе PVE

8.3. 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.

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 (пакет control-sshd-permit-root-login):

```
# 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 для различных функций (таблица 13). Важно обеспечить доступность этих портов и отсутствие их блокировки межсетевыми экранами.

Т а б л и ц а 13 – 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

8.3.1. 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
```

П р и м е ч а н и е . В PVE это можно сделать в панели управления: выбрать узел, перейти в «Система» → «Хосты», добавить все узлы, которые будут включены в состав кластера (рис. 191).

П р и м е ч а н и е . Имя машины не должно присутствовать в файле `/etc/hosts` разрешающимся в 127.0.0.1.

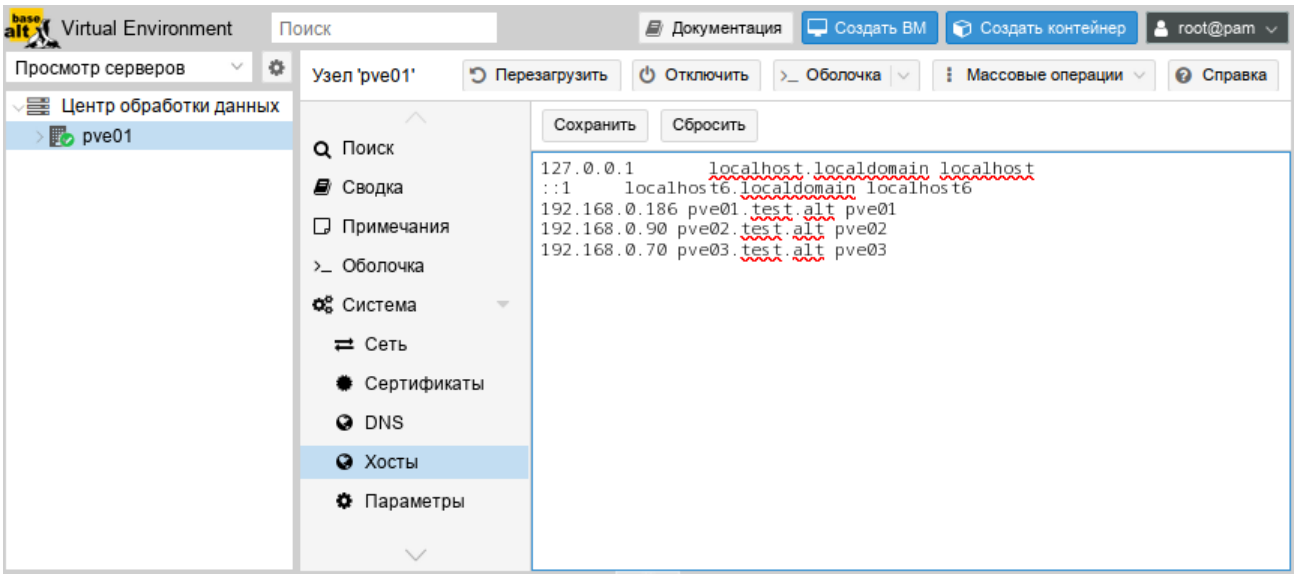

Рис. 191 – Редактирование записей в файле `/etc/hosts`

8.3.2. Создание кластера в веб-интерфейсе

Для создания кластера необходимо выполнить следующие действия:

1) в панели управления любого узла кластера выбрать «Центр обработки данных» → «Кластер» и нажать на кнопку «Создать кластер» (рис. 192);
2) в открывшемся окне, задать название кластера, выбрать IP-адрес интерфейса, на котором узел кластера будет работать, и нажать на кнопку «Создать» (рис. 193);
3) при успешном создании кластера будет выведена соответствующая информация (рис. 194).

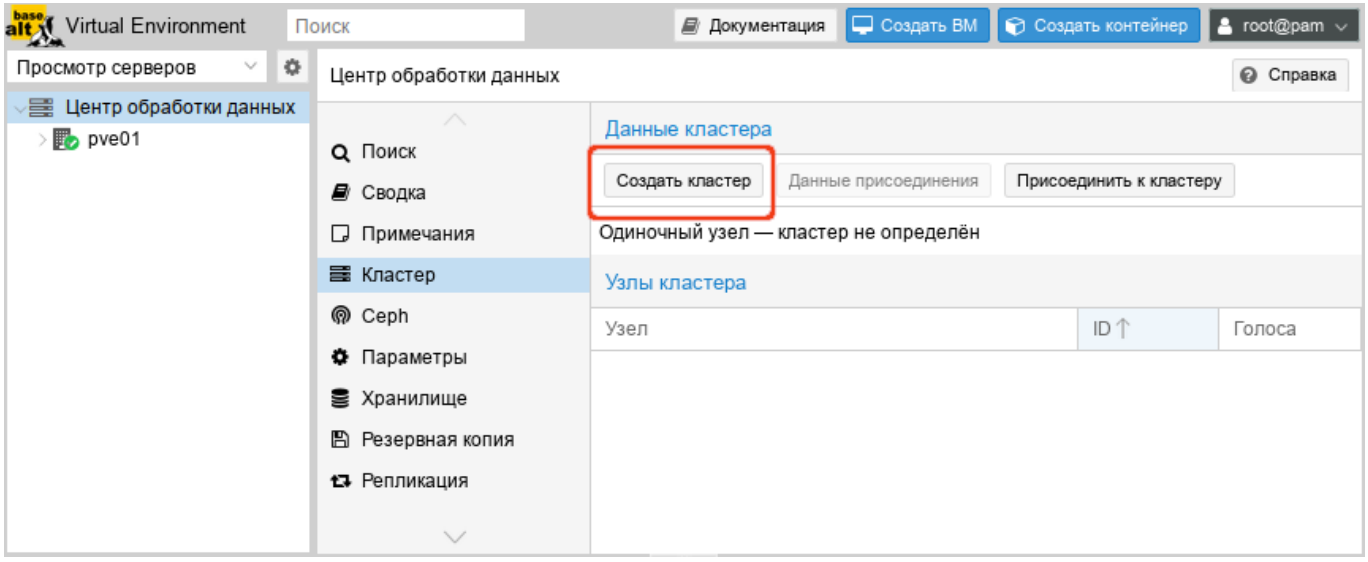

Рис. 192 – Создание кластера в веб-интерфейсе

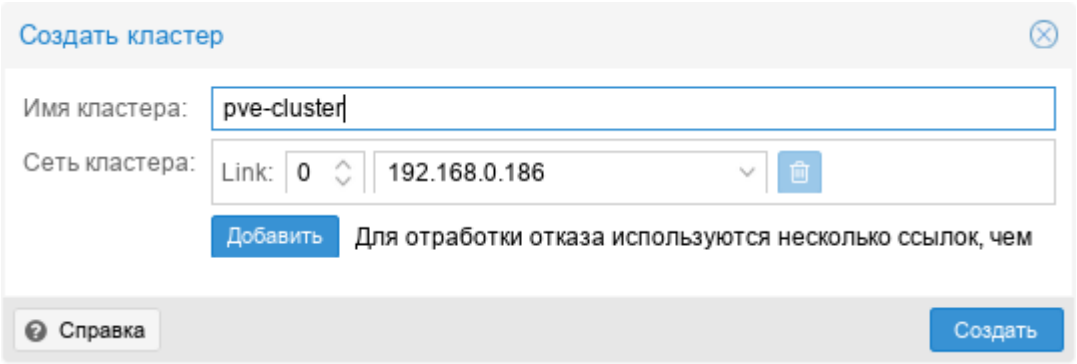

Рис. 193 – Создание кластера в веб-интерфейсе. Название кластера

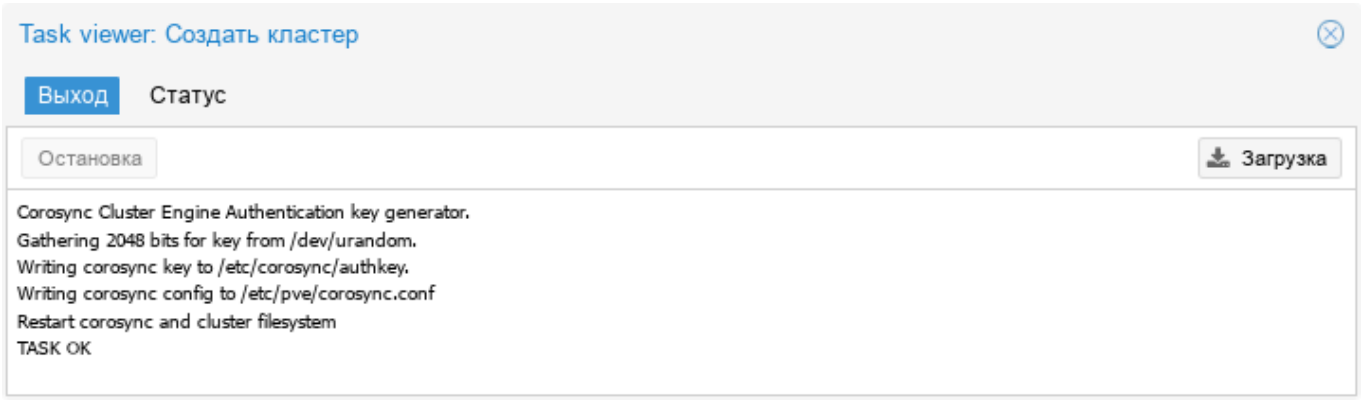

Рис. 194 – Информация о создании кластера

Для добавления узла в кластер необходимо выполнить следующие действия:

1) в панели управления узла, на котором был создан кластер, выбрать «Центр обработки данных» → «Кластер» и нажать на кнопку «Данные присоединения» (рис. 195);
2) в открывшемся окне, нажав кнопку «Копировать данные» (рис. 196), скопировать данные присоединения;
3) перейти в панель управления узла, который следует присоединить к кластеру. Выбрать пункт «Центр обработки данных» → «Кластер» и нажать на кнопку «Присоединить к кластеру» (рис. 197);
4) в поле «Данные присоединения» вставить данные присоединения, поля «Адрес сервера» и «Отпечаток» при этом будут заполнены автоматически. В поле «Пароль» ввести пароль пользователя root первого узла (рис. 198) и нажать на кнопку «Присоединить <имя кластера>»;
5) через несколько минут, после завершения репликации всех настроек, узел будет подключен к кластеру (рис. 199).

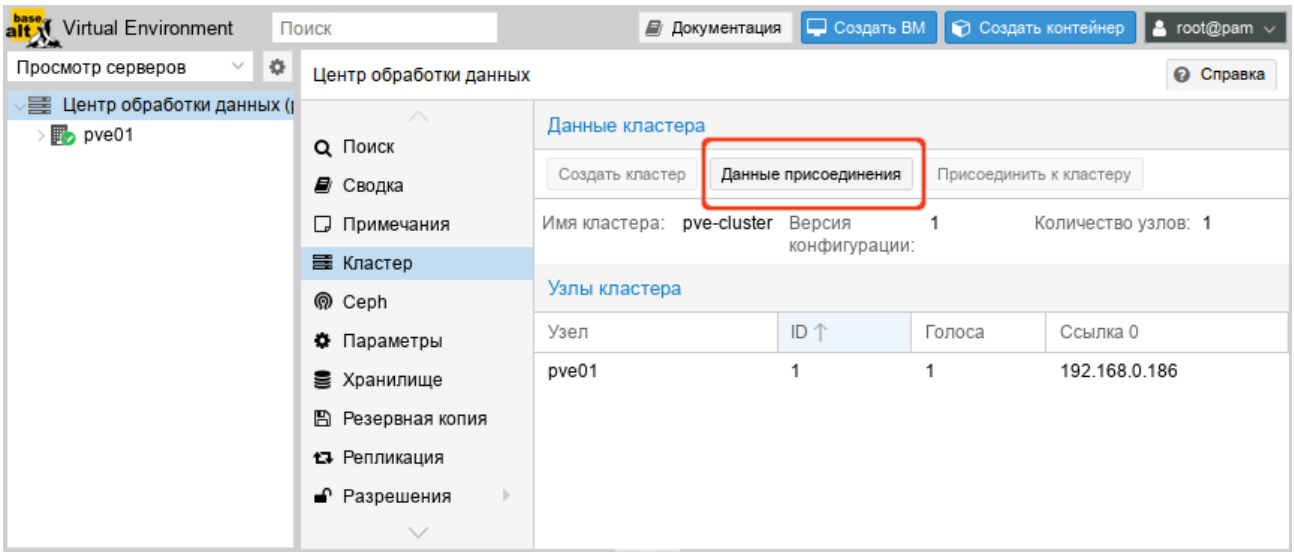

Рис. 195 – Создание кластера в веб-интерфейсе. Получить данные присоединения

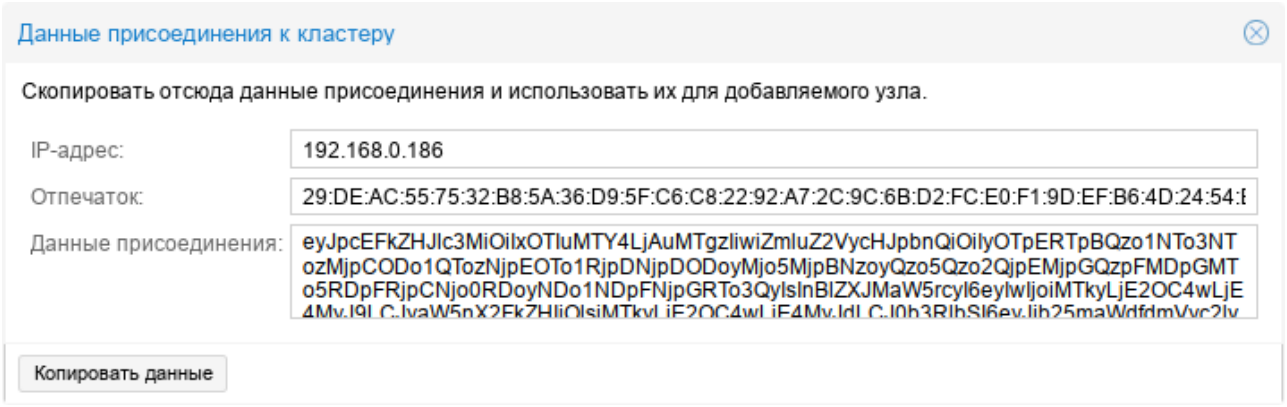

Рис. 196 – Создание кластера в веб-интерфейсе. Данные присоединения

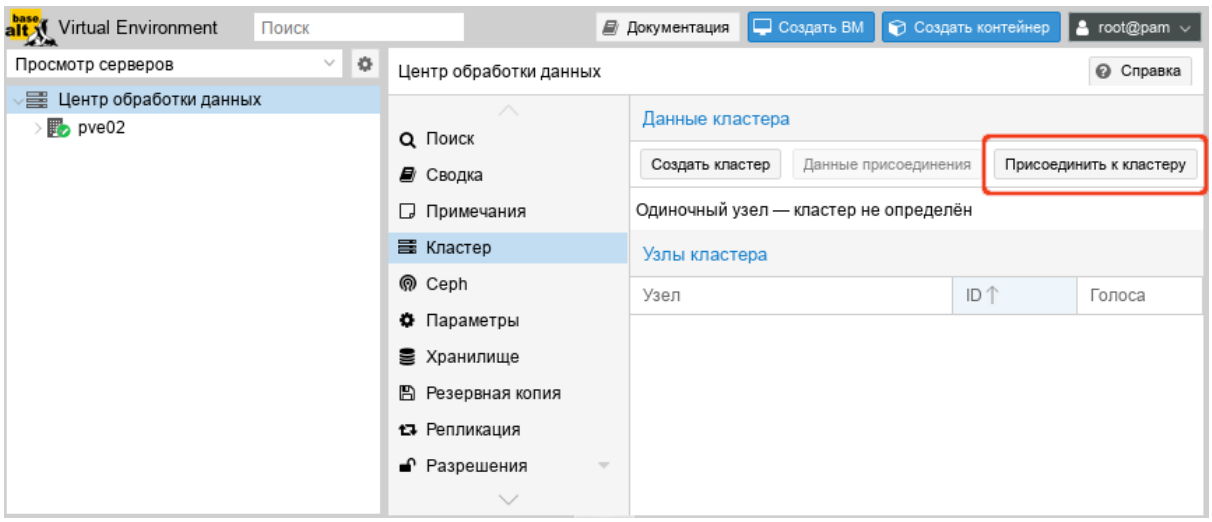

Рис. 197 – Узел, который следует присоединить к кластеру

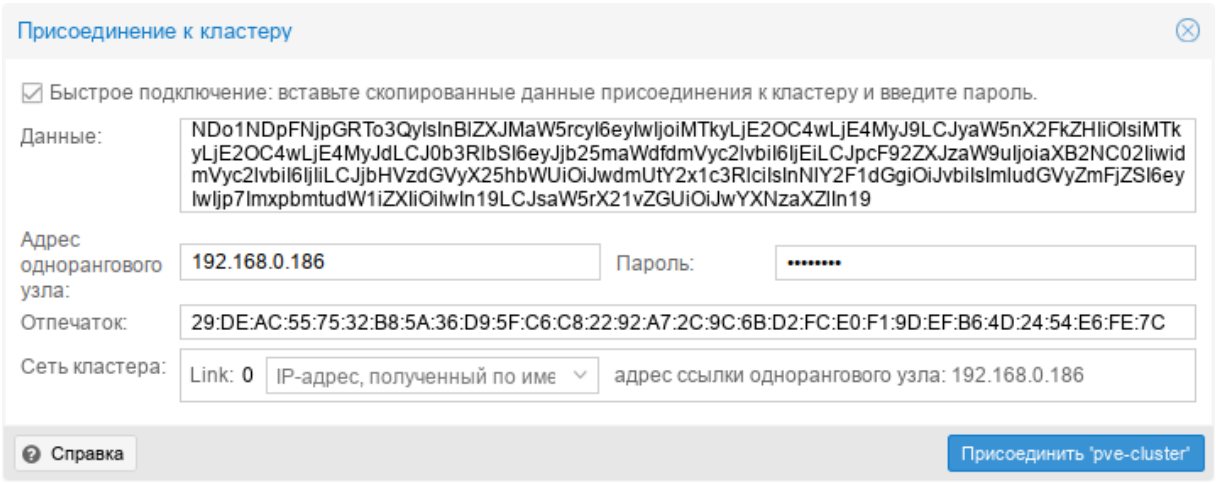

Рис. 198 – Присоединение узла к кластеру

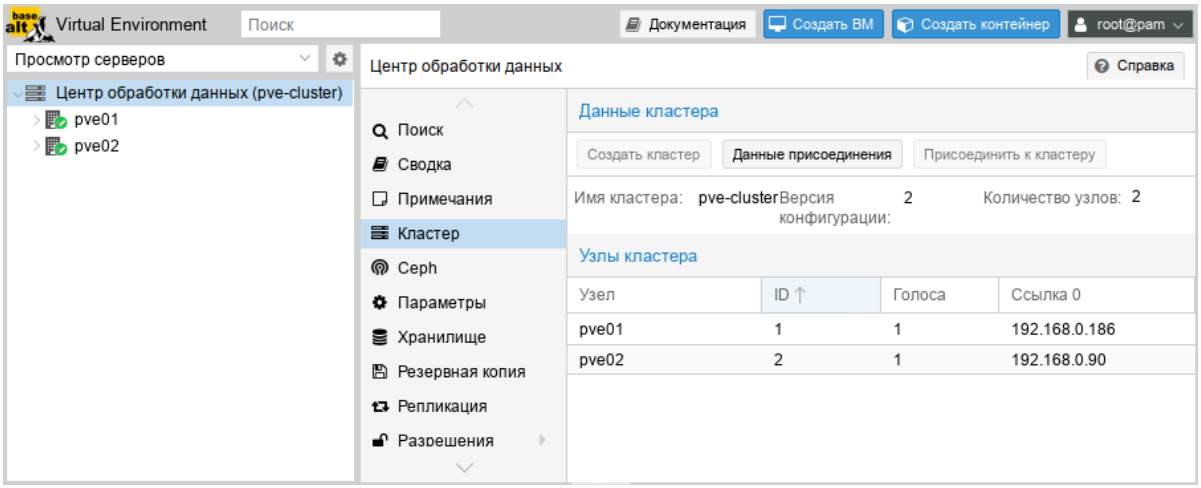

Рис. 199 – Узлы кластера в веб-интерфейсе

Сразу после инициализации кластера, в пределах каждого из узлов, доступно одно локальное хранилище данных (рис. 200).

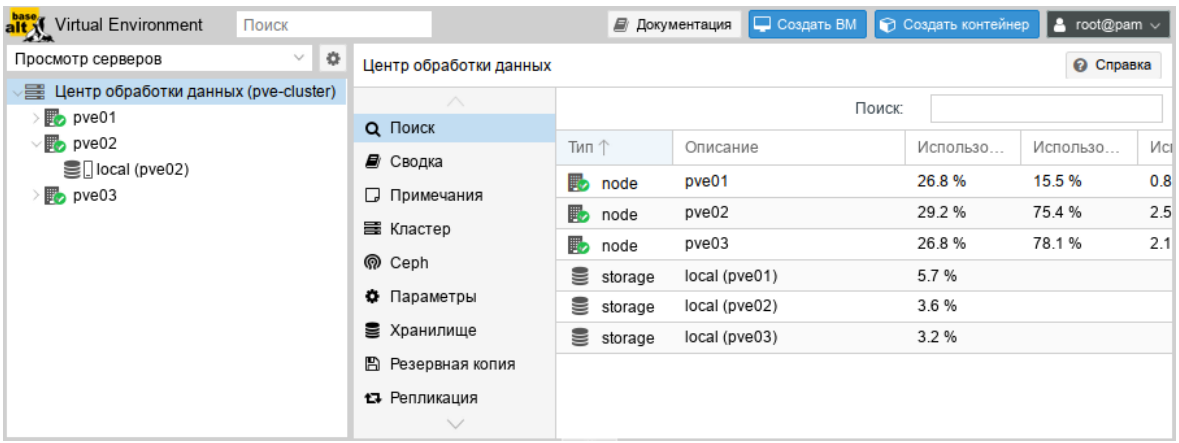

Рис. 200 – Узлы кластера и локальные хранилища данных

8.3.3. 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 informati--
Name:           pve-cluster
Config Version:   1
Transport:       knet
Secure auth:      on

Quorum informati--
Date:           Tue Aug 22 09:06:24 2023
Quorum provider:  corosync_votequorum
Nodes:          1
Node ID:         0x00000001
Ring ID:         1.d6
Quorate:         Yes

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

Membership informati--
   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 команду:

```
# 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 f‘r 'pv’01': ***
Establishing API connection with ho‘t 'pv’01'

Login succeeded.
Request addition of this node
Join request OK, finishing setup locally
stopping pve-cluster service
backup old database ‘o '/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 no‘e 'pv’03' to cluster.
```

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

8.3.4. 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.

8.3.5. 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 Мбайт.

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/`.

8.4. 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.

8.4.1. 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:

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

Доступные типы хранилищ приведены в таблице 14.

Т а б л и ц а 14 – Доступные типы хранилищ

Хранилище	PVE тип	Уровень	Общее (shared)	Снимки (snapshots)
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
Proxmox Backup	pbs	файл/блок	да	-

8.4.2. 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.

П р и м е ч а н и е . Файл `/etc/pve/storage.cfg` по умолчанию генерируется при создании пользователя.

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 (таблица 15).

Т а б л и ц а 15 – 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).

8.4.3. Работа с хранилищами в PVE

8.4.3.1. 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 Гбайт 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>]
```

8.4.3.2. Добавление хранилища в веб-интерфейсе PVE

Хранилища, которые могут быть добавлены в веб-интерфейсе PVE (рис. 201):

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.

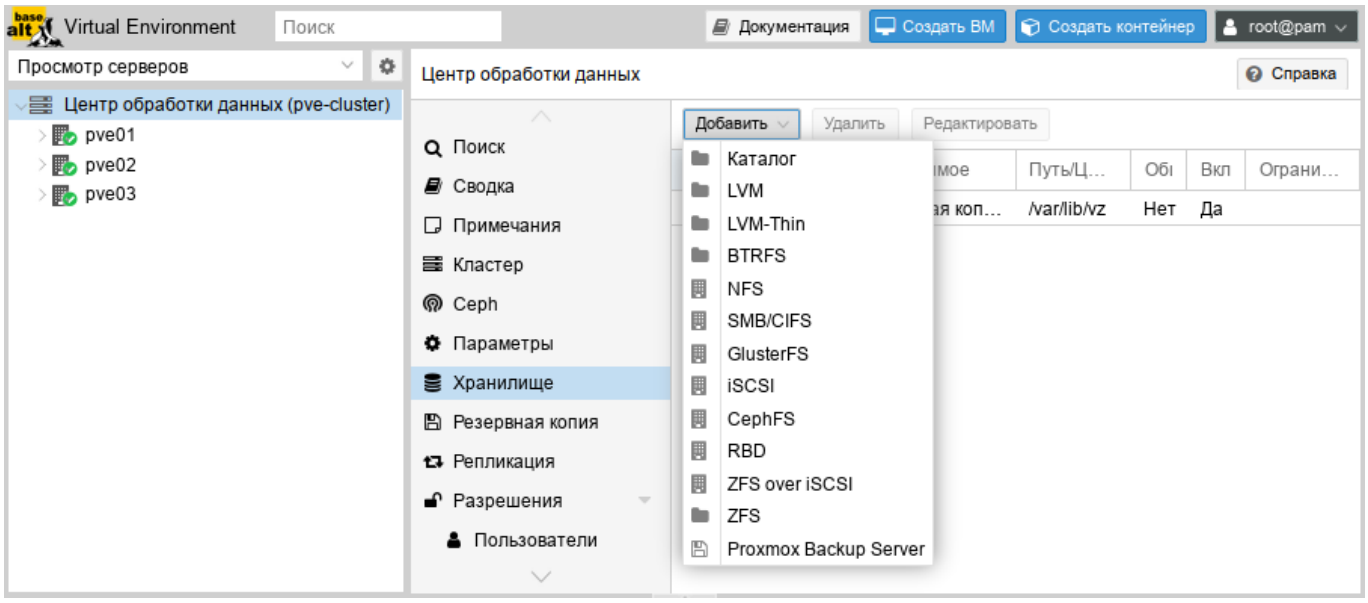

Рис. 201 – Выбор типа добавляемого хранилища

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. 202), а хранилища на базе RBD можно использовать только для хранения образов дисков и контейнеров.

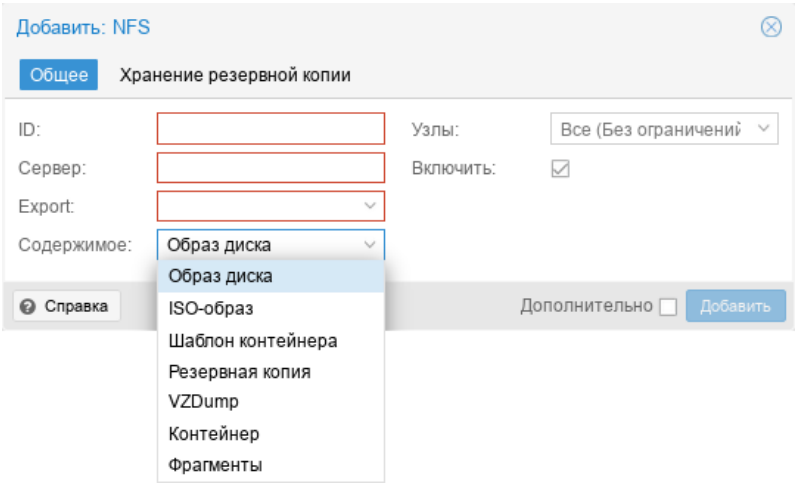

Рис. 202 – Выбор ролей для хранилища

8.4.3.3. Каталог

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 (таблица 16). Эта структура используется на всех файловых хранилищах.

Т а б л и ц а 16 – 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 «Центр обработки данных» → «Хранилище», нажать на кнопку «Добавить» и в выпадающем меню выбрать пункт «Каталог» (рис. 201). На рис. 203 показан диалог создания хранилища local-iso типа «Каталог» для хранения ISO-образов и шаблонов контейнеров, которое будет смонтировано в каталог `/mnt/iso`.

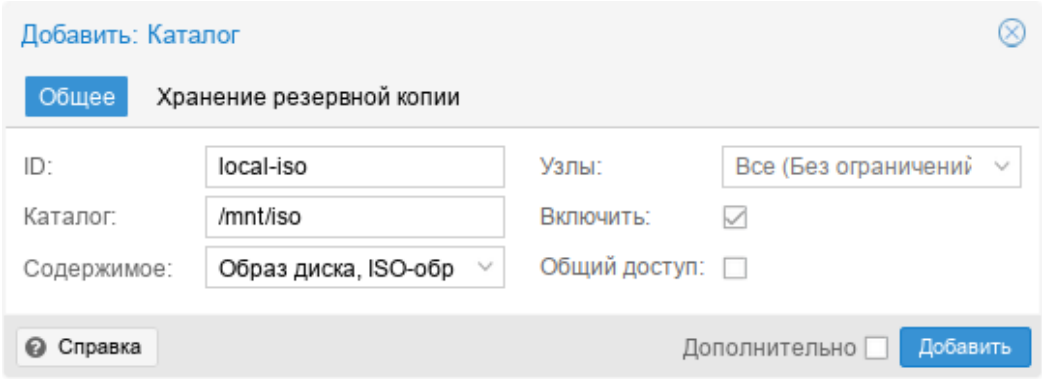

Рис. 203 – Добавление хранилища «Каталог»

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>
```

8.4.3.4. 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. 204 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
```

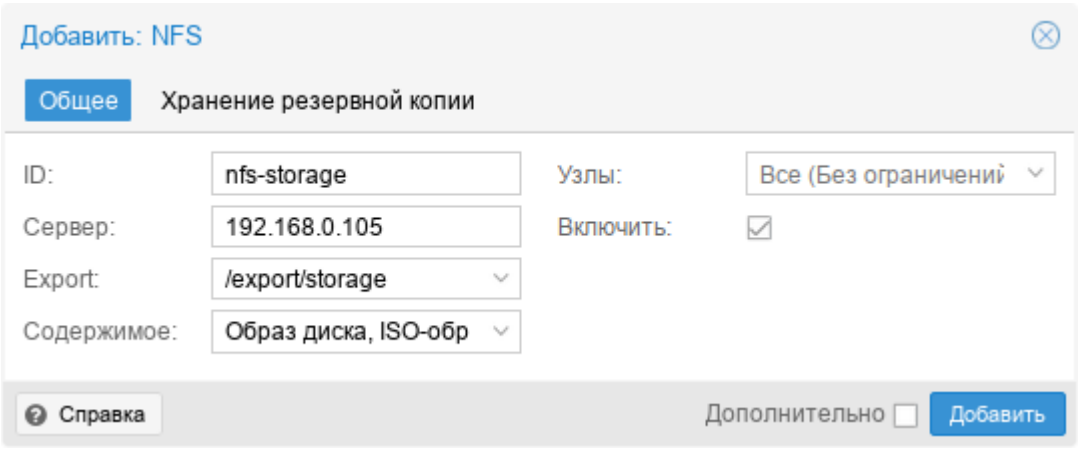

Рис. 204 – Создание хранилища NFS

Получить список совместных ресурсов с сервера NFS:

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

8.4.3.5. 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. 205 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
```

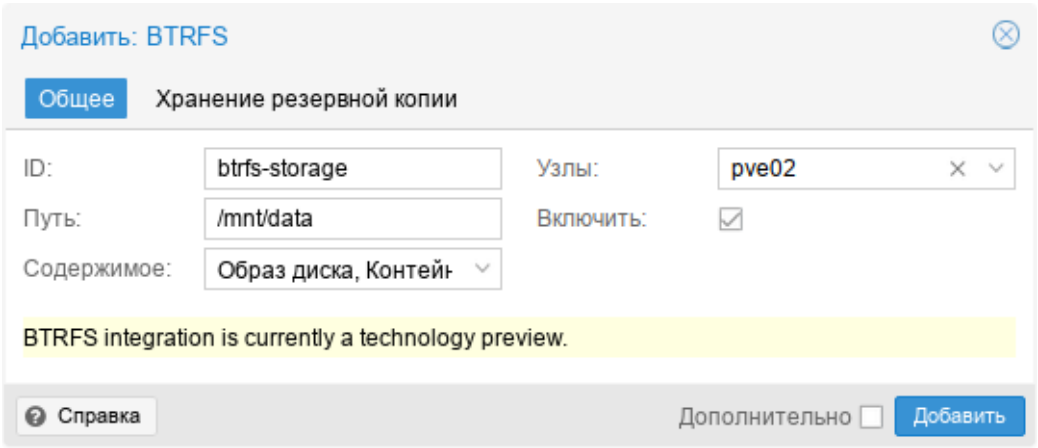

Рис. 205 – Создание хранилища BTRFS

8.4.3.6. SMB/CIFS

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

П р и м е ч а н и е . Для возможности просмотра общих ресурсов на каждом узле кластера должен быть установлен 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. 206 показано присоединение хранилища SMB/CIFS с именем newCIFS с удаленного сервера 192.168.0.105.

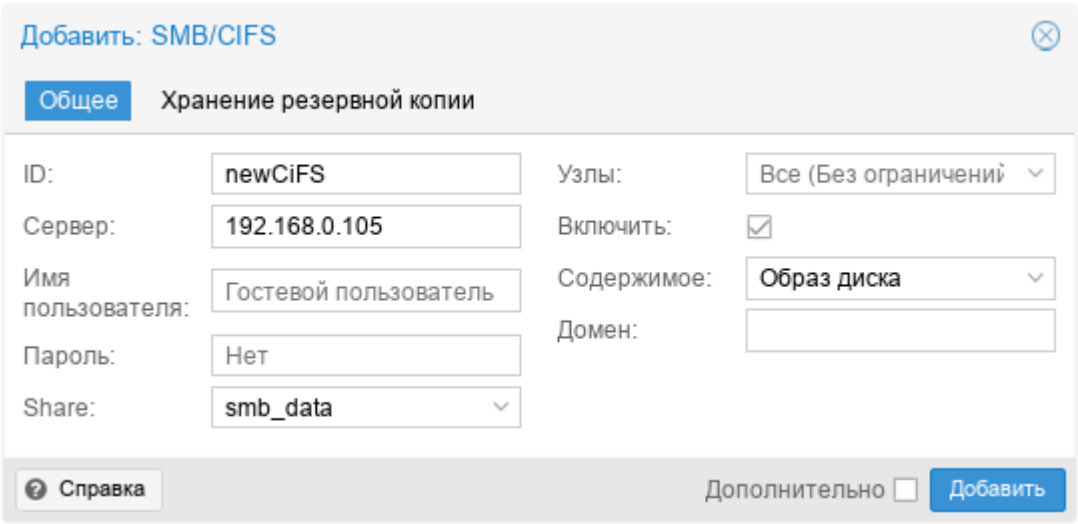

Рис. 206 – Добавление CIFS хранилища

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 удаленный 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
```

8.4.3.7. 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. 207 показано присоединение хранилища GlusterFS с именем gluster-01 с удаленного сервера 192.168.0.105.

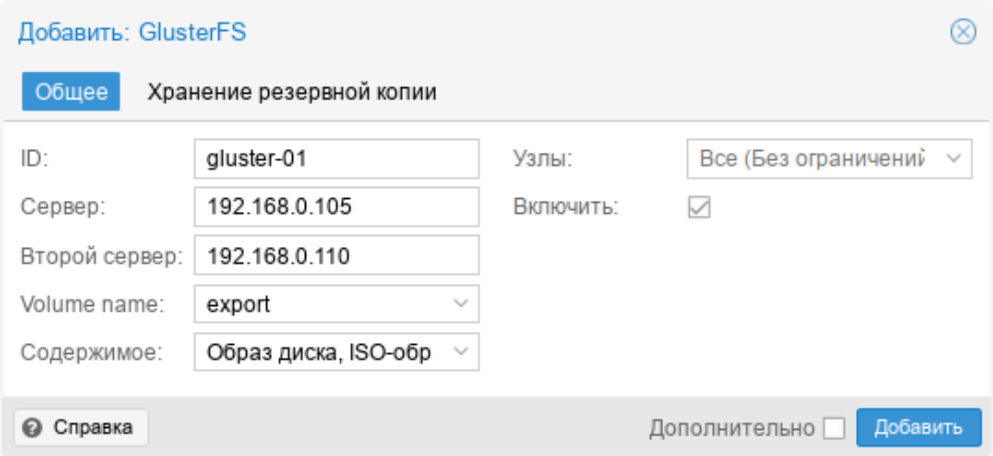

Рис. 207 – Создание хранилища GlusterFS

8.4.3.8. 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`).

8.4.3.8.1. Создание локального хранилища LVM в веб-интерфейсе

П р и м е ч а н и е . 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» выбрать пункт «LVM» и нажать на кнопку «Создать: Volume Group» (рис. 208).

В открывшемся окне (рис. 209) следует выбрать диск, задать имя группы томов, отметить пункт «Добавить хранилище» (если этот пункт не отмечен будет создана только группа томов) и нажать на кнопку «Создать».

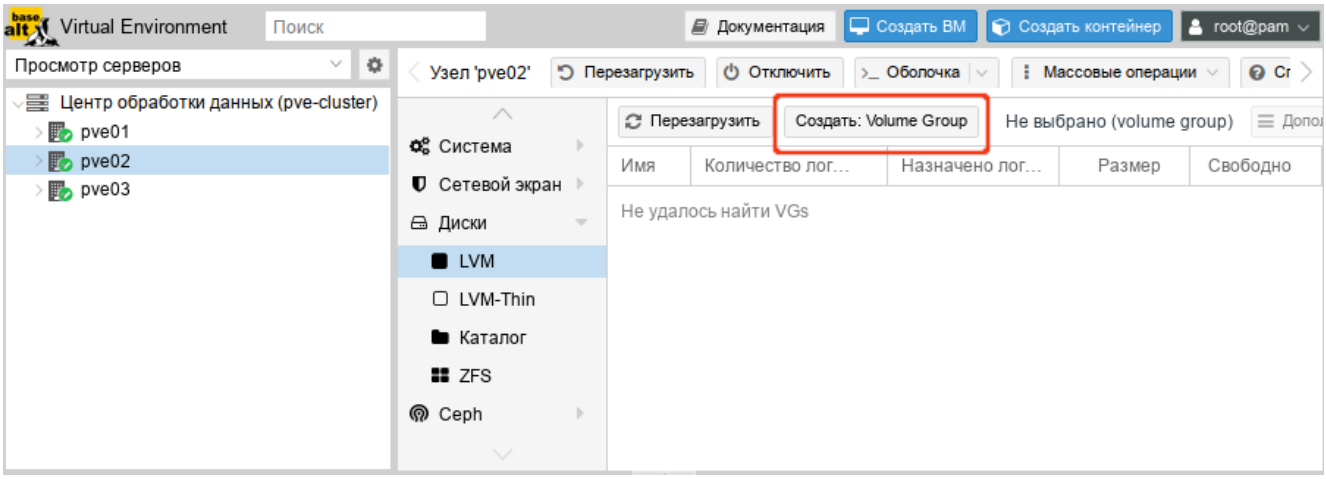

Рис. 208 – Пункт «LVM» в разделе «Диски»

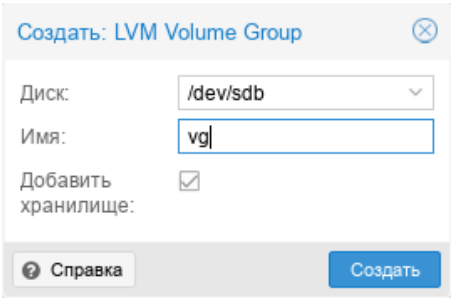

Рис. 209 – Создание группы томов

Для того чтобы внести изменения в настройки LVM хранилища следует выбрать «Центр обработки данных» → «Хранилище», затем хранилище LVM и нажать на кнопку «Редактировать». В открывшемся окне (рис. 210) можно изменить тип содержимого контейнера, включить/отключить хранилище.

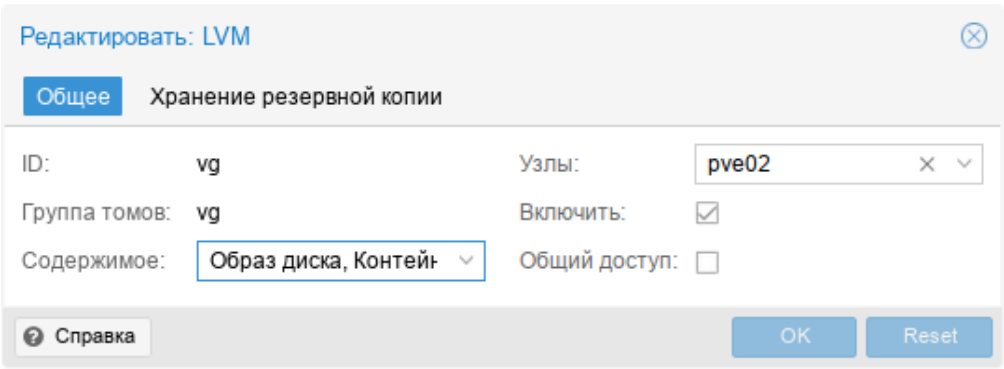

Рис. 210 – Редактирование LVM хранилища

8.4.3.8.2. Создание хранилища LVM в командной строке

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

- создать физический том (PV):

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

- создать группу томов (VG) с именем vg:

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

- создать логические тома (LV):

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

- показать информацию о физических томах:

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

- показать информацию о группах томов:

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

- получить список доступных PVE групп томов:

```
# pvesm lvmscan
vg
```

- создать LVM хранилище с именем myspace:

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

8.4.3.9. 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.

8.4.3.9.1. 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» выбрать пункт «LVM-Thin» и нажать на кнопку «Создать: Thinpool» (рис. 211). В открывшемся окне (рис. 212) следует выбрать диск, задать имя группы томов, отметить пункт «Добавить хранилище» (если этот пункт не отмечен будет создана только группа томов) и нажать на кнопку «Создать».

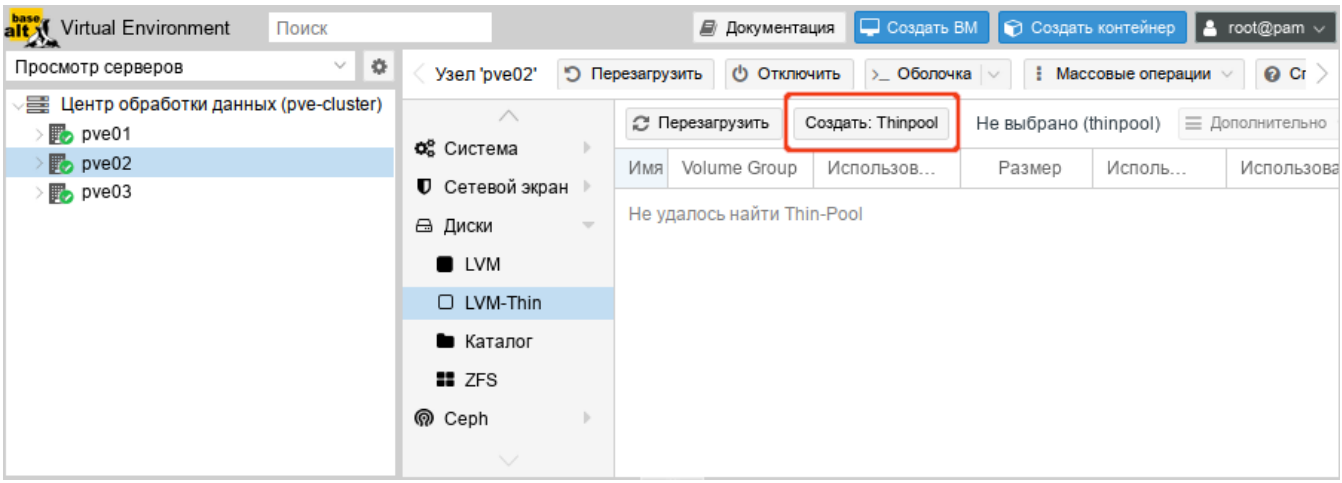

Рис. 211 – Пункт «LVM-Thin» в разделе «Диски»

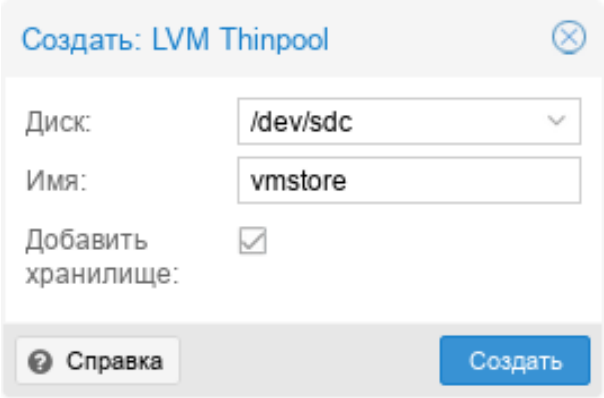

Рис. 212 – Создание LVM-Thin хранилища

Для того чтобы внести изменения в настройки LVM-Thin хранилища следует выбрать «Центр обработки данных» → «Хранилище», затем хранилище LVM-Thin и нажать на кнопку «Редактировать». В открывшемся окне (рис. 213) можно изменить тип содержимого контейнера, включить/отключить хранилище.

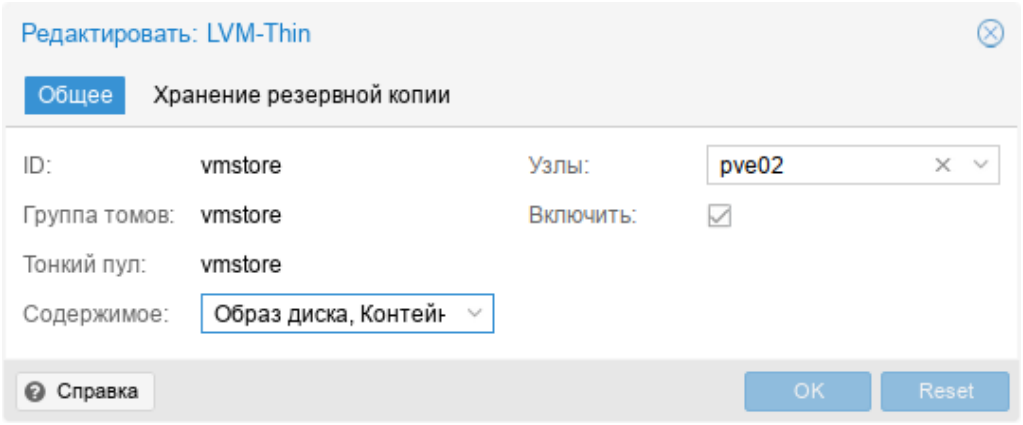

Рис. 213 – Редактирование LVM-Thin хранилища

8.4.3.9.2. 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 Гбайт) 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
```

8.4.3.10. 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. 214 показано добавление адресата iSCSI с именем test1-iSCSI, который настроен на удаленном узле 192.168.0.105. Параметр «Использовать LUN напрямую» – разрешение/запрет прямого применения LUN (параметр должен быть установлен на запрет, разрешение может привести к потере данных).

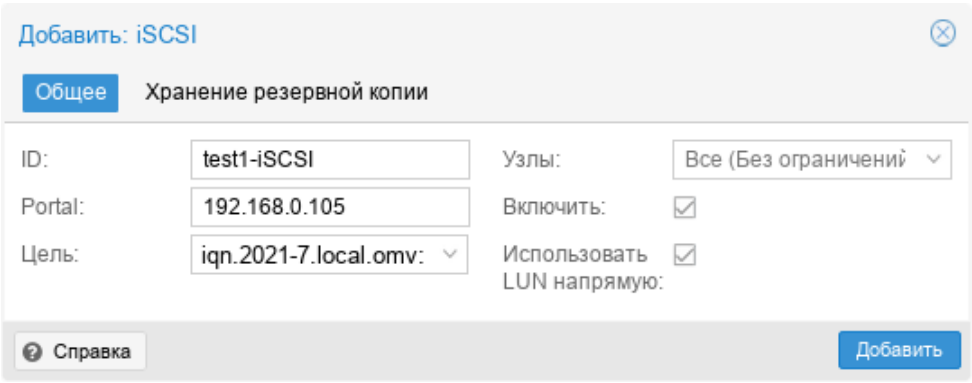

Рис. 214 – Добавление хранилища «iSCSI»

Посмотреть доступные для подключения iSCSI target-ы:

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

8.4.3.11. 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.

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. 215 показано добавление хранилища RBD.

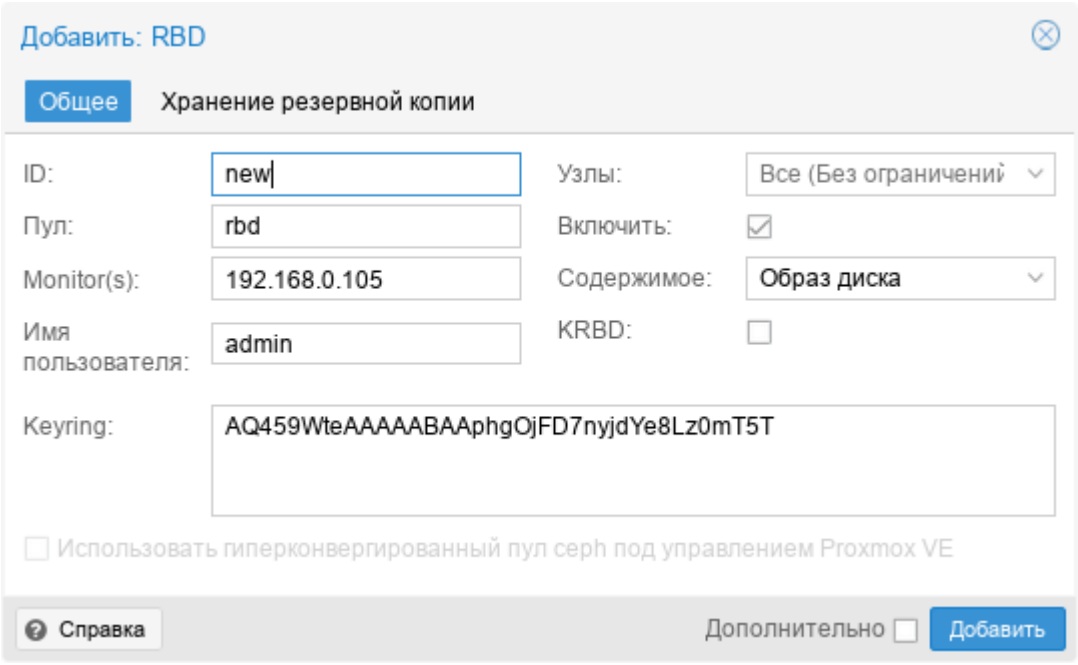

Рис. 215 – Добавление хранилища «RBD»

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`.

8.4.3.12. 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. 216 показано добавление хранилища CephFS.

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

```
# ceph fs ls
```

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

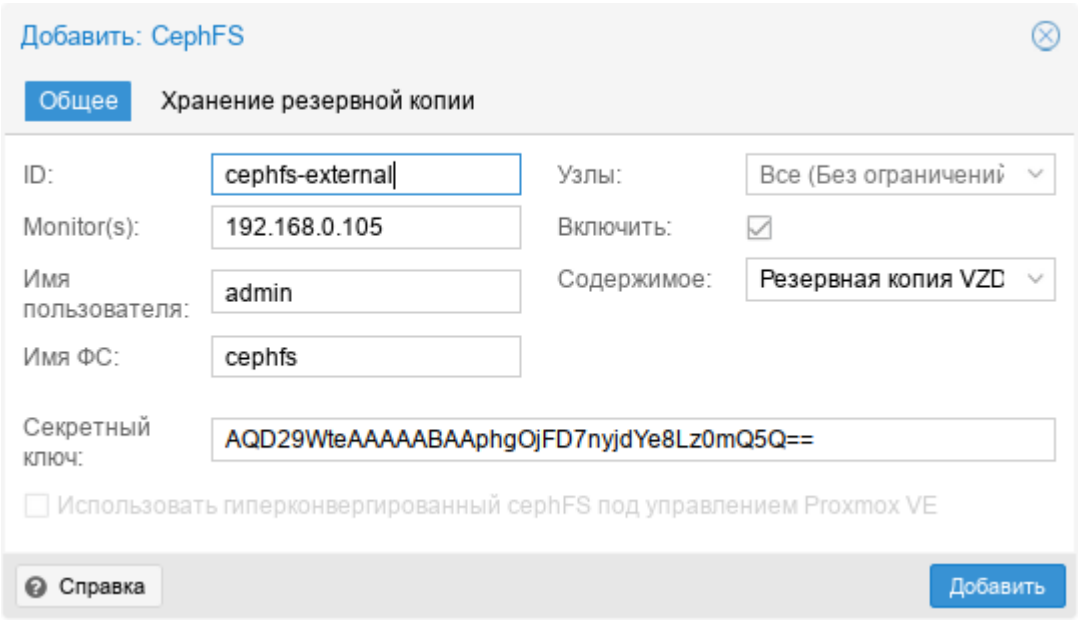

Рис. 216 – Добавление хранилища «CephFS»

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 «Секретный ключ».

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
```

8.4.3.13. Proxmox Backup Server

Proxmox Backup Server – 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 Server» 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. 217 показано добавление хранилища «Proxmox Backup Server» с именем `pbs_backup` с удаленного сервера 192.168.0.123.

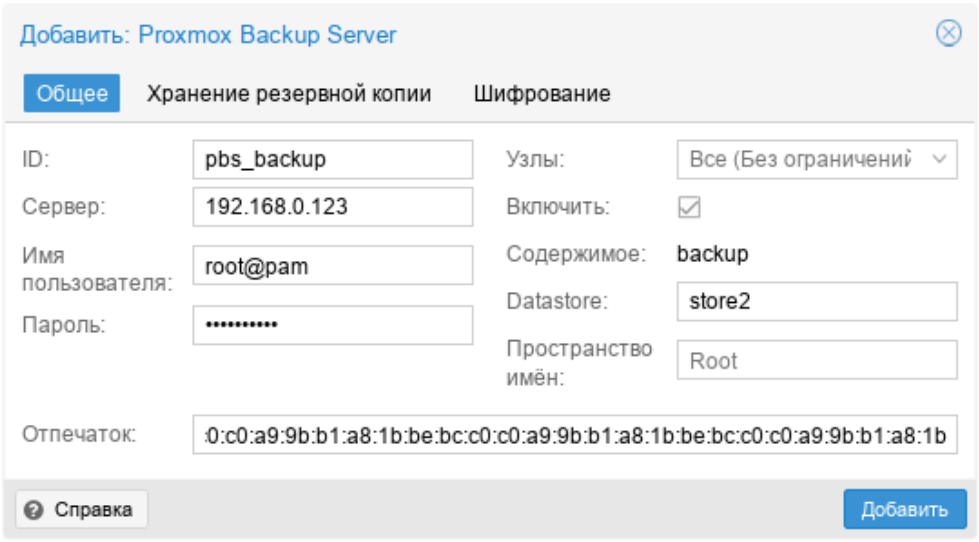

Рис. 217 – Добавление хранилища «Proxmox Backup Server»

Добавление хранилища «Proxmox Backup Server» 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
```

8.5. Сетевая подсистема

PVE использует сетевой стек Linux, что обеспечивает большую гибкость в настройке сети на узлах PVE. Настройку сети можно выполнить либо через графический интерфейс «Хост» → «Система» → «Сеть» (рис. 218), либо вручную, редактируя файлы в каталоге `/etc/net/ifaces`.

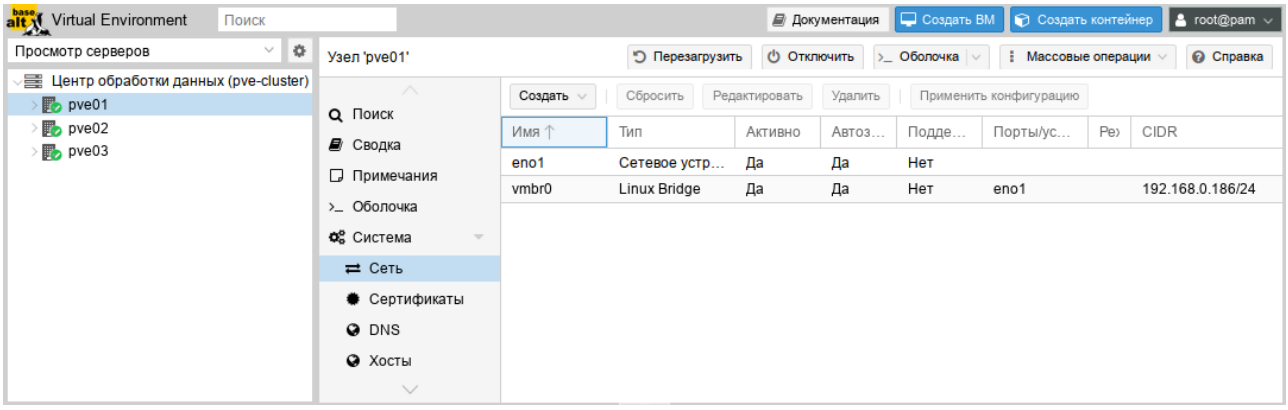

Рис. 218 – Сетевые интерфейсы узла pve01

Примечание. Интерфейс vmbr0 необходим для подключения гостевых систем к базовой физической сети. Это мост Linux, который можно рассматривать как виртуальный коммутатор, к которому подключены гостевые системы и физические интерфейсы.

Виды сетевых соединений в PVE (рис. 219):

- «Linux Bridge» – способ соединения двух сегментов Ethernet на канальном уровне;
- «Linux Bond» – реализация агрегации нескольких сетевых интерфейсов в единый логический bonded интерфейс на базе ядра Linux;
- «Linux VLAN» – реализация VLAN на базе ядра Linux;
- «OVS Bridge» – реализация моста на базе Open vSwitch. Мосты Open vSwitch могут содержать необработанные устройства Ethernet, а также виртуальные интерфейсы OVSBonds или OVSIntPorts. Эти мосты могут нести несколько vlan и быть разбиты на «внутренние порты» для использования в качестве интерфейсов vlan на хосте. Все интерфейсы, входящие в мост, должны быть перечислены в опции `ovs_ports`;
- «OVS Bond» – реализация агрегации сетевых интерфейсов на базе Open vSwitch. Отличается от реализованной в ядре Linux режимами балансировки нагрузки;
- «OVS VLAN» – реализация VLAN на базе Open vSwitch.

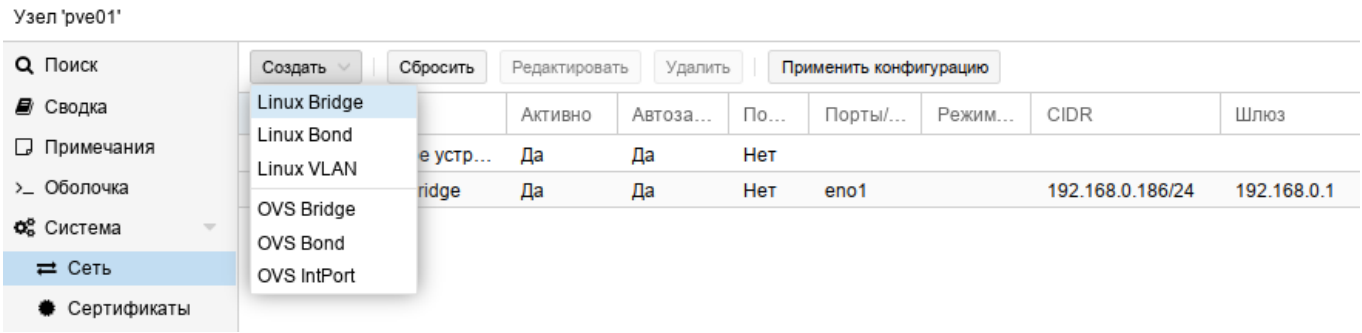

Рис. 219 – Новый сетевой интерфейс

Мосты, VLAN и агрегированные интерфейсы Open vSwitch и Linux не должны смешиваться. Например, не нужно добавлять Linux Bond к OVSBridge или наоборот.

8.5.1. Применение изменений сетевых настроек

Все изменения конфигурации сети, сделанные в веб-интерфейсе PVE, сначала записываются во временный файл, что позволяет сделать несколько связанных изменений одновременно. Это также позволяет убедиться, что изменения сделаны верно, так как неправильная конфигурация сети может сделать узел недоступным.

Для применения изменений сетевых настроек, сделанных в веб-интерфейсе PVE, следует нажать на кнопку «Применить конфигурацию». В результате изменения будут применены в реальном времени. Еще один способ применить новую сетевую конфигурацию – перезагрузить узел.

8.5.2. Имена сетевых устройств

В PVE используются следующие соглашения об именах устройств:

- устройства Ethernet: en*, имена сетевых интерфейсов systemd;
- мосты: vmbr[N], где 0 ⩽ N ⩽ 4094 (vmbr0 – vmbr4094);
- сетевые объединения: bond[N], где 0 ⩽ N (bond0, bond1, …);
- VLAN: можно просто добавить номер VLAN к имени устройства, отделив точкой (eno1.50, bond1.30).

Systemd использует префикс `en` для сетевых устройств Ethernet.

Следующие символы зависят от драйвера устройства и того факта, какая схема подходит первой:

- `o<index>[n<phys_port_name>|d<dev_port>]` – встроенные устройства;
- `s<slot>[f<function>][n<phys_port_name>|d<dev_port>]` – устройства по идентификатору горячего подключения;
- `[P<domain>]p<bus>s<slot>[f<function>][n<phys_port_name>|d<dev_port>]` – устройства по идентификатору шины;
- `x<MAC>` – устройство по MAC-адресу.

Наиболее распространенные шаблоны:

- eno1 – первая сетевая карта;
- enp0s3 – сетевая карта в слоте 3 шины pcibus 0.

8.5.3. Конфигурация сети с использованием моста

Мосты похожи на физические сетевые коммутаторы, реализованные в программном обеспечении. Все виртуальные системы могут использовать один мост, также можно создать несколько мостов для отдельных сетевых доменов. На каждом хосте можно создать до 4094 мостов.

По умолчанию после установки на каждом узле PVE есть единственный мост (vmbr0), который подключается к первой плате Ethernet (рис. 220).

При этом ВМ ведут себя так, как если бы они были напрямую подключены к физической сети. Каждая ВМ видна в сети со своим MAC-адресом.

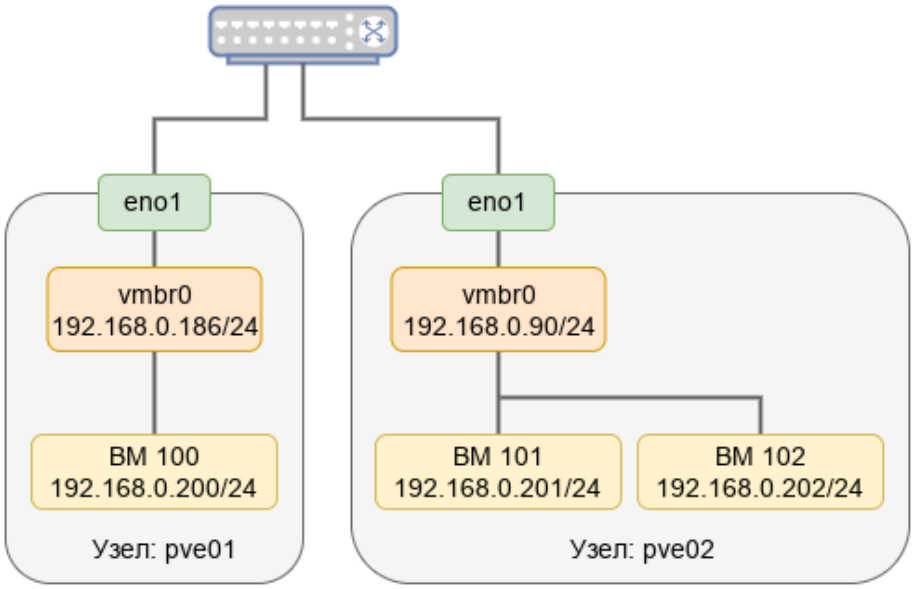

Рис. 220 – Узлы PVE с мостом vmbr0

8.5.3.1. Внутренняя сеть для ВМ

Если необходимо несколько ВМ объединить в локальную сеть без доступа во внешний мир, можно создать новый мост.

8.5.3.1.1. Настройка в веб-интерфейсе PVE

Для того чтобы создать мост, в разделе «Сеть» необходимо нажать на кнопку «Создать» и в выпадающем меню выбрать пункт «Linux Bridge» или «OVS Bridge» (см. рис. 219).

В открывшемся окне (рис. 221) в поле «Имя» следует указать имя моста и нажать на кнопку «Создать».

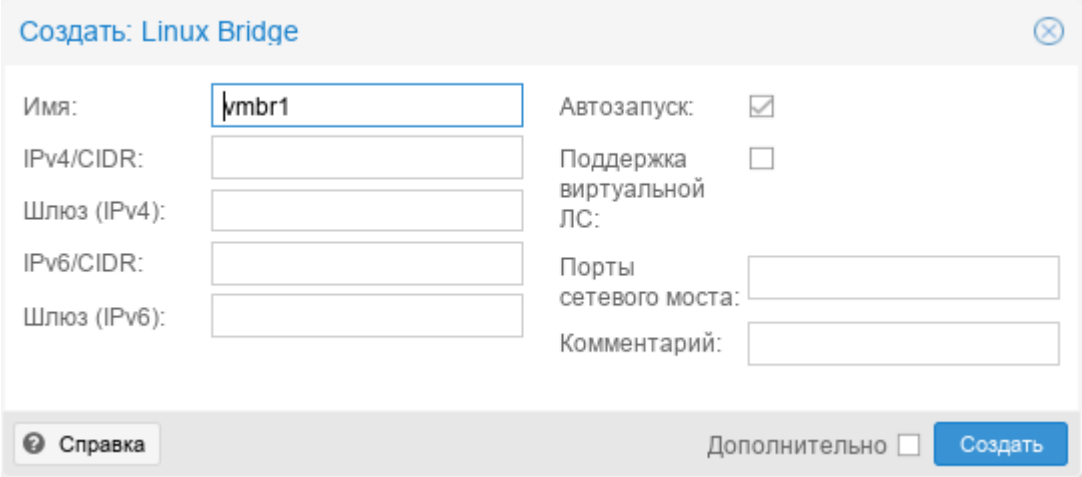

Рис. 221 – PVE. Создание Linux Bridge

Создание моста Open vSwitch (рис. 222) отличается возможностью указания дополнительных параметров Open vSwitch (поле «Параметры OVS»).

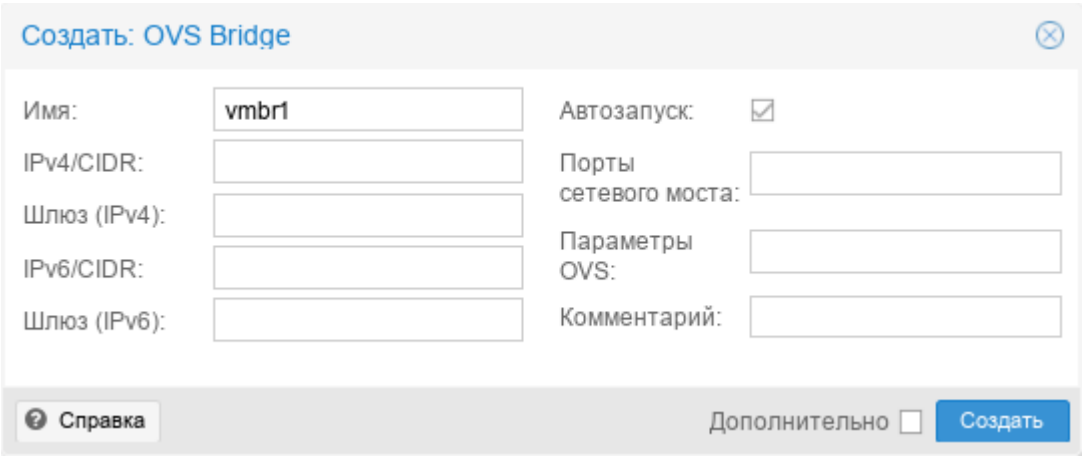

Рис. 222 – PVE. Создание OVS Bridge

Примечание. Адрес интерфейса можно не указывать, настроенные на подключение к интерфейсу ВМ будут использовать его как обычный коммутатор. Если же указать IPv4 и/или IPv6-адрес, то он будет доступен извне на интерфейсах, перечисленных в поле «Порты сетевого моста».

Для применения изменений следует нажать на кнопку «Применить конфигурацию».

Теперь мост vmbr1 можно назначать ВМ (рис. 223).

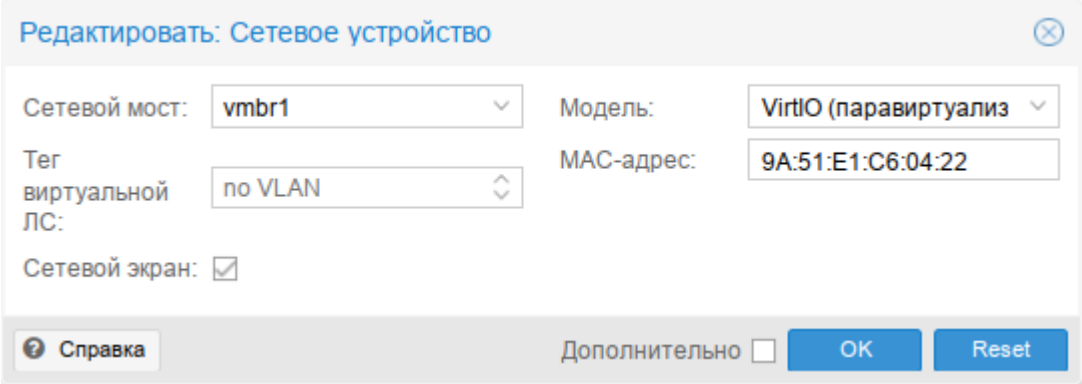

Рис. 223 – PVE. Назначение моста vmbr1 ВМ

8.5.3.1.2. Настройка в командной строке

Для настройки Linux bridge с именем vmbr1, следует выполнить следующие команды:

```
# mkdir /etc/net/ifaces/vmbr1
# cat <<EOF > /etc/net/ifaces/vmbr1/options
BOOTPROTO=static
CONFIG_IPV4=yes
HOST=
ONBOOT=yes
TYPE=bri
EOF
```

Примечание. Если в мост будут входить интерфейсы, которые до этого имели IP-адрес, то этот адрес должен быть удален. Интерфейсы, которые будут входить в мост, должны быть указаны в опции HOST. Пример настройки моста vmbr1 на интерфейсе enp0s8 (IP-адрес для интерфейса vmbr1 будет взят из `/etc/net/ifaces/enp0s8/ipv4address`):

```
# mkdir /etc/net/ifaces/vmbr1
# cp /etc/net/ifaces/enp0s8/* /etc/net/ifaces/vmbr1/
# rm -f /etc/net/ifaces/enp0s8/{i,r}*
# cat <<EOF > /etc/net/ifaces/vmbr1/options
BOOTPROTO=static
CONFIG_WIRELESS=no
CONFIG_IPV4=yes
HOST='enp0s8'
ONBOOT=yes
TYPE=bri
EOF
```

Для настройки OVS bridge с именем vmbr1, следует выполнить следующие команды:

```
# mkdir /etc/net/ifaces/vmbr1
# cat <<EOF > /etc/net/ifaces/vmbr1/options
```

```
BOOTPROTO=static
CONFIG_IPV4=yes
ONBOOT=yes
TYPE=ovsbr
EOF
```

Пример настройки OVS bridge с именем vmbr1 на интерфейсе enp0s8:

```
# mkdir /etc/net/ifaces/vmbr1
# cp /etc/net/ifaces/enp0s8/* /etc/net/ifaces/vmbr1/
# rm -f /etc/net/ifaces/enp0s8/{i,r}*
# cat <<EOF > /etc/net/ifaces/vmbr1/options
BOOTPROTO=static
CONFIG_IPV4=yes
ONBOOT=yes
HOST='enp0s8'
TYPE=ovsbr
EOF
```

Для применения изменений необходимо перезапустить службу network:

```
# systemctl restart network
```

или перезагрузить узел:

```
# reboot
```

8.5.4. Объединение/агрегация интерфейсов

Объединение/агрегация интерфейсов (bonding) – это объединение двух и более сетевых интерфейсов в один логический интерфейс для достижения отказоустойчивости или увеличения пропускной способности. Поведение такого логического интерфейса зависит от выбранного режима работы.

Если на узлах PVE есть несколько портов Ethernet, можно распределить точки отказа, подключив сетевые кабели к разным коммутаторам, и в случае проблем с сетью агрегированное соединение переключится с одного кабеля на другой.

Агрегация интерфейсов может сократить задержки выполнения миграции в реальном времени и повысить скорость репликации данных между узлами кластера PVE.

Кластерную сеть (Corosync) рекомендуется настраивать с несколькими сетями. Corosync не нуждается в агрегации для резервирования сети, поскольку может сам переключаться между сетями.

8.5.4.1. Параметры Linux Bond

В таблице 17 приведены режимы агрегации Linux Bond.

Т а б л и ц а 17 – Режимы агрегации Linux Bond

Режим	Название	Описание	Отказоустойчивость	Балансировка нагрузки
balance-rr или mode=0	Round-robin	Режим циклического выбора активного интерфейса для трафика. Пакеты последовательно передаются и принимаются через каждый интерфейс один за другим. Данный режим не требует применения специальных коммутаторов.	Да	Да
active-backup или mode=1	Active Backup	В этом режиме активен только один интерфейс, остальные находятся в режиме горячей замены. Если активный интерфейс выходит из строя, его заменяет резервный. MAC-адрес интерфейса виден извне только на одном сетевом адаптере, что предотвращает путаницу в сетевом коммутаторе. Это самый простой режим, работает с любым оборудованием, не требует применения специальных коммутаторов.	Да	Нет
balance-xor или mode=2	XOR	Один и тот же интерфейс работает с определенным получателем. Передача пакетов распределяется между интерфейсами на основе формулы ((MAC-адрес источника) XOR (MAC-адрес получателя)) % число интерфейсов. Режим не требует применения специальных коммутаторов. Этот режим обеспечивает балансировку нагрузки и отказоустойчивость.	Да	Да
broadcast или mode=3	Широковещательный	Трафик идет через все интерфейсы одновременно.	Да	Нет
LACP (802.3ad) или mode=4	Агрегирование каналов по стандарту IEEE 802.3ad	В группу объединяются одинаковые по скорости и режиму интерфейсы. Все физические интерфейсы используются одновременно в соответствии со спецификацией IEEE 802.3ad. Для реализации этого режима необходима поддержка на уровне драйверов сетевых карт и коммутатор, поддерживающий стандарт IEEE 802.3ad (коммутатор требует отдельной настройки).	Да	Да

Окончание таблицы 17

Режим	Название	Описание	Отказоустойчивость	Балансировка нагрузки
balance-tlb или mode=5	Адаптивная балансировка нагрузки при передаче	Исходящий трафик распределяется в соответствии с текущей нагрузкой (с учетом скорости) на интерфейсах (для данного режима необходима его поддержка в драйверах сетевых карт). Входящие пакеты принимаются только активным сетевым интерфейсом.	Да	Да (исходящий трафик)
balance-alb или mode=6	Адаптивная балансировка нагрузки	Включает в себя балансировку исходящего трафика, плюс балансировку на прием (rlb) для IPv4 трафика и не требует применения специальных коммутаторов (балансировка на прием достигается на уровне протокола ARP, перехватом ARP ответов локальной системы и перезаписью физического адреса на адрес одного из сетевых интерфейсов, в зависимости от загрузки).	Да	Да

В таблице 18 приведены алгоритмы выбора каналов (распределения пакетов между физическими каналами, входящими в многоканальное соединение) для режимов balance-alb, balance-tlb, balance-xor, 802.3ad (значение параметра `xmit-hash-policy`).

Т а б л и ц а 18 – Режимы выбора каналов при организации балансировки нагрузки

Режим	Описание
layer2	Канал для отправки пакета однозначно определяется комбинацией MAC-адреса источника и MAC-адреса назначения. Весь трафик между определенной парой узлов всегда идет по определенному каналу. Алгоритм совместим с IEEE 802.3ad. Этот режим используется по умолчанию.
layer2+3	Канал для отправки пакета определяется по совокупности MAC- и IP-адресов источника и назначения. Трафик между определенной парой IP-хостов всегда идет по определенному каналу (обеспечивается более равномерная балансировка трафика, особенно в случае, когда большая его часть передается через промежуточные маршрутизаторы). Для протоколов 3 уровня, отличных от IP, данный алгоритм равносилен layer2. Алгоритм совместим с IEEE 802.3ad.
layer3+4	Канал для отправки пакета определяется по совокупности IP-адресов и номеров портов источника и назначения (трафик определенного узла может распределяться между несколькими каналами, но пакеты одного и того же TCP/UDP-соединения всегда передаются по одному и тому же каналу). Для фрагментированных пакетов TCP и UDP, а также для всех прочих протоколов 4 уровня, учитываются только IP-адреса. Для протоколов 3 уровня, отличных от IP, данный алгоритм равносилен layer2. Алгоритм не полностью совместим с IEEE 802.3ad.

Для создания агрегированного bond-интерфейса средствами etcnet необходимо создать каталог для интерфейса (например, bond0) с файлами `options`, `ipv4address`. В файле `options` в переменной TYPE следует указать тип интерфейса bond, в переменной HOST перечислить родительские интерфейсы, которые будут входить в агрегированный интерфейс, в переменной BONDMODE указать режим (по умолчанию 0), а опции для модуля ядра bonding – в BONDOPTIONS.

Примечание. Агрегированный bond-интерфейс можно создать в веб-интерфейсе ЦУС → модуль «Объединение интерфейсов» (должен быть установлен пакет alterator-net-bond).

8.5.4.2. Параметры OVS Bond

В таблице 19 приведены параметры OVS Bond.

Таблица 19 – Параметры OVS Bond

Параметр	Описание
`bond_mode=active-backup`	В этом режиме активен только один интерфейс, остальные находятся в режиме горячей замены. Если активный интерфейс выходит из строя, его заменяет резервный. MAC-адрес интерфейса виден извне только на одном сетевом адаптере, что предотвращает путаницу в сетевом коммутаторе. Этот режим не требует какой-либо специальной настройки на коммутаторах.
`bond_mode=balance-slb`	Режим простой балансировки на основе MAC и VLAN. В этом режиме нагрузка трафика на интерфейсы постоянно измеряется, и если один из интерфейсов сильно загружен, часть трафика перемещается на менее загруженные интерфейсы. Параметр bond-rebalance-interval определяет, как часто OVS должен выполнять измерение нагрузки трафика (по умолчанию 10 секунд). Этот режим не требует какой-либо специальной настройки на коммутаторах.
`bond_mode=balance-tcp`	Этот режим выполняет балансировку нагрузки, принимая во внимание данные уровней 2 – 4 (например, MAC-адрес, IP-адрес и порт TCP). На коммутаторе должен быть настроен LACP. Этот режим похож на режим `mode=4 Linux Bond`. Всегда, когда это возможно, рекомендуется использовать этот режим.

Окончание таблицы 19

Параметр	Описание
`lacp=[active|passive|off]`	Управляет поведением протокола управления агрегацией каналов (LACP). На коммутаторе должен быть настроен протокол LACP. Если коммутатор не поддерживает LACP, необходимо использовать `bond_mode=balance-slb` или `bond_mode=active-backup`.
`other-config:lacp-fallback-ab=true`	Устанавливает поведение LACP для переключения на `bond_mode=active-backup` в качестве запасного варианта.
`other_config:lacp-time=[fast|slow]`	Определяет, с каким интервалом управляющие пакеты LACPDU отправляются по каналу LACP: каждую секунду (fast) или каждые 30 секунд (slow). По умолчанию slow.
`other_config:bond-detect-mode=[miimon|carrier]`	Режим определения состояния канала. По умолчанию carrier.
`other_config:bond-miimon-interval=100`	Устанавливает периодичность MII мониторинга в миллисекундах.
`other_config:bond_updelay=1000`	Задает время задержки в миллисекундах, перед тем как поднять линк при обнаружении восстановления канала.
`other_config:bond-rebalance-interval=10000`	Устанавливает периодичность выполнения измерения нагрузки трафика в миллисекундах (по умолчанию 10 секунд).

8.5.4.2.1. Агрегированный bond-интерфейс с фиксированным IP-адресом

Конфигурация с агрегированным bond-интерфейсом с фиксированным IP-адресом может использоваться как распределенная/общая сеть хранения. Преимущество будет заключаться в том, что вы получите больше скорости, а сеть будет отказоустойчивой (рис. 224).

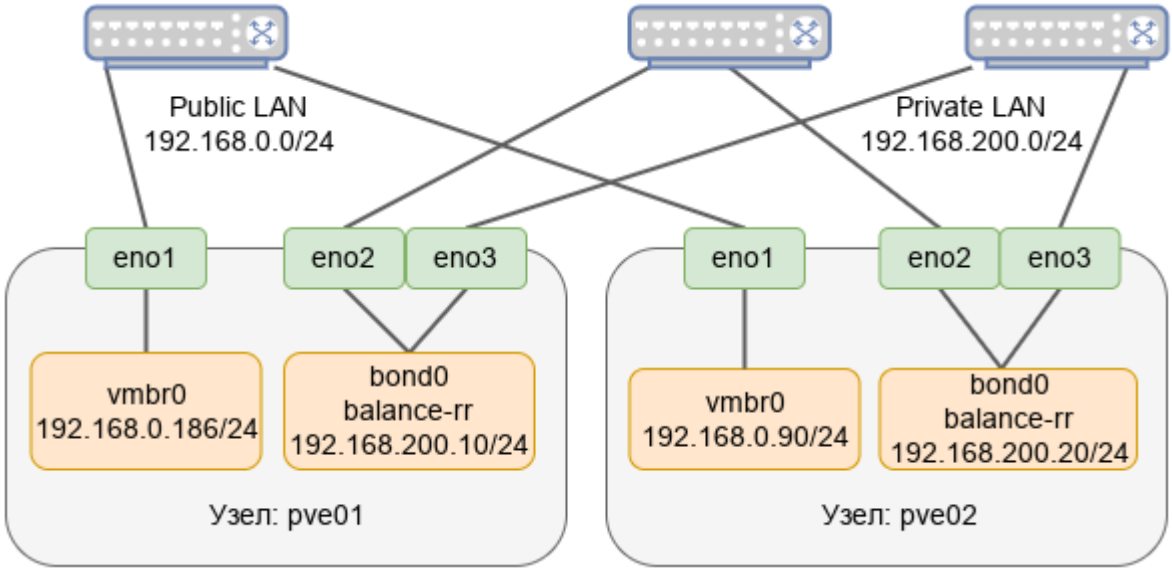

Рис. 224 – Агрегированный bond-интерфейс с фиксированным IP-адресом

8.5.4.2.1.1. Настройка в веб-интерфейсе

Для настройки Linux Bond необходимо выполнить следующие действия:

- перейти в раздел «Сеть», нажать на кнопку «Создать» и в выпадающем меню выбрать пункт «Linux Bond» (рис. 219);
- в открывшемся окне указать имя агрегированного соединения, в выпадающем списке «Режим» выбрать режим агрегации (в примере balance-rr), в поле «Устройства» указать сетевые интерфейсы, которые будут входить в объединение, в поле «IPv4/CIDR» ввести IP-адрес объединения и нажать на кнопку «Создать» (рис. 225).

Примечание. В зависимости от выбранного режима агрегации будут доступны разные поля.

Для применения изменений нажать на кнопку «Применить конфигурацию». Получившаяся конфигурация показана на рис. 226.

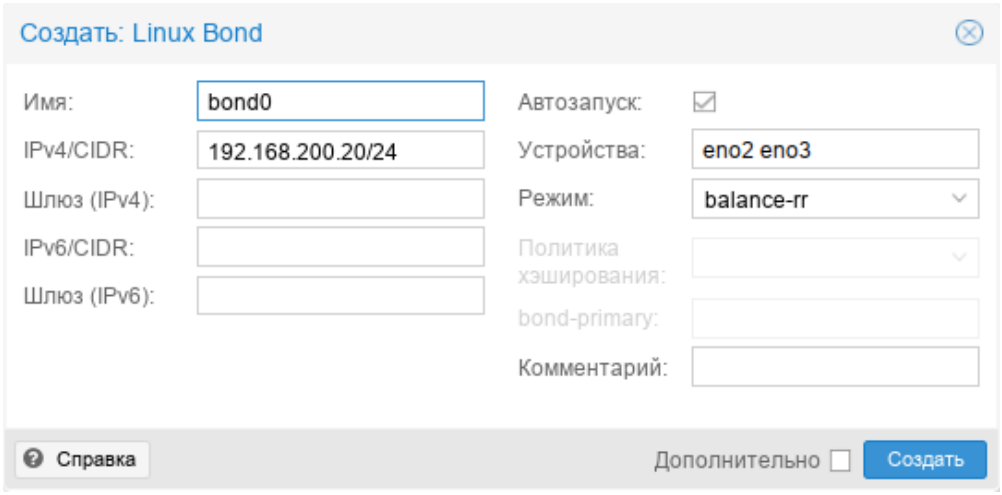

Рис. 225 – Редактирование параметров объединения bond0

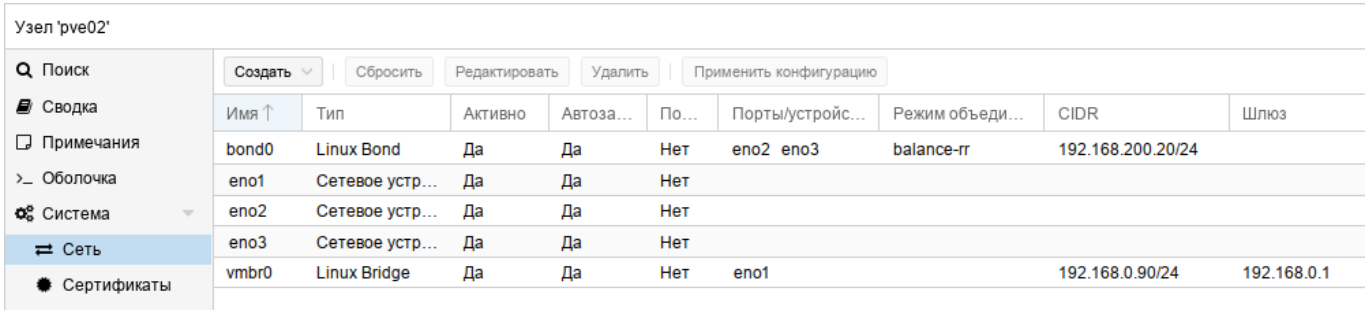

Рис. 226 – Агрегированный интерфейс с фиксированным IP-адресом

8.5.4.2.1.2. Настройка в командной строке

Для создания такой конфигурации, необходимо выполнить следующие действия:

1) создать Linux Bond bond0 на интерфейсах eno1 и eno2, выполнив следующие команды:

```
# mkdir /etc/net/ifaces/bond0
# rm -f /etc/net/ifaces/eno1/{i,r}*
# rm -f /etc/net/ifaces/eno2/{i,r}*
# cat <<EOF > /etc/net/ifaces/bond0/options
BOOTPROTO=static
CONFIG_WIRELESS=no
CONFIG_IPV4=yes
HOST='eno1 eno2'
ONBOOT=yes
TYPE=bond
BONDOPTIONS='miimon=100'
BONDMODE=0
EOF
```

где:

- `BONDMODE=1` – режим агрегации Round-robin;
- `HOST='eno1 eno2'` – интерфейсы, которые будут входить в объединение;
- `miimon=100` – определяет, как часто производится мониторинг MII (Media Independent Interface);

2) в файле `/etc/net/ifaces/bond0/ipv4address`, указать IP-адрес для интерфейса bond0:

```
# echo "192.168.200.20/24" > /etc/net/ifaces/bond0/ipv4address
```

3) перезапустить службу network, чтобы изменения вступили в силу:

```
# systemctl restart network
```

8.5.4.2.2. Агрегированный bond-интерфейс в качестве порта моста

Чтобы сделать гостевую сеть отказоустойчивой можно использовать bond напрямую в качестве порта моста (рис. 227).

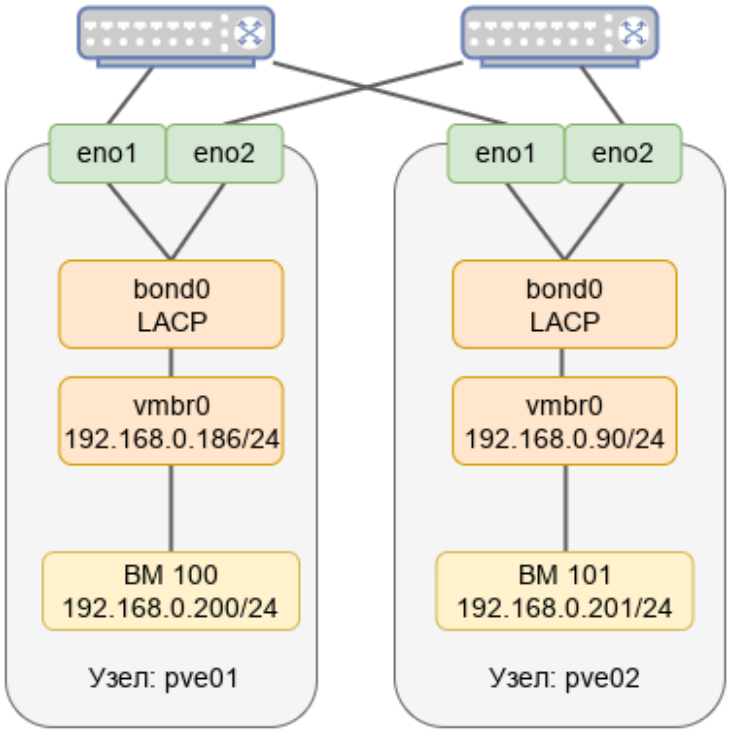

Рис. 227 – Агрегированный bond-интерфейс в качестве порта моста

8.5.4.2.2.1. Настройка в веб-интерфейсе

Для настройки Linux Bond необходимо выполнить следующие действия:

1) перейти в раздел «Сеть», выбрать существующий мост vmbr0 и нажать на кнопку «Редактировать» (рис. 228);
2) в открывшемся окне (рис. 229) удалить содержимое поля «Порты сетевого моста» и нажать на кнопку «ОК»;
3) нажать на кнопку «Создать» (рис. 219) и в выпадающем меню выбрать пункт «Linux Bond»;
4) в открывшемся окне в выпадающем списке «Режим» выбрать режим агрегации (в примере LACP), в поле «Устройства» указать сетевые интерфейсы, которые будут входить в объединение, в выпадающем списке «Политика хэширования» выбрать политику хэширования и нажать на кнопку «Создать» (рис. 230);
5) выбрать мост vmbr0 и нажать на кнопку «Редактировать»;
6) в открывшемся окне в поле «Порты сетевого моста» вписать значение bond0 и нажать на кнопку «ОК» (рис. 231);
7) для применения изменений нажать на кнопку «Применить конфигурацию»;
8) получившаяся конфигурация показана на рис. 232.

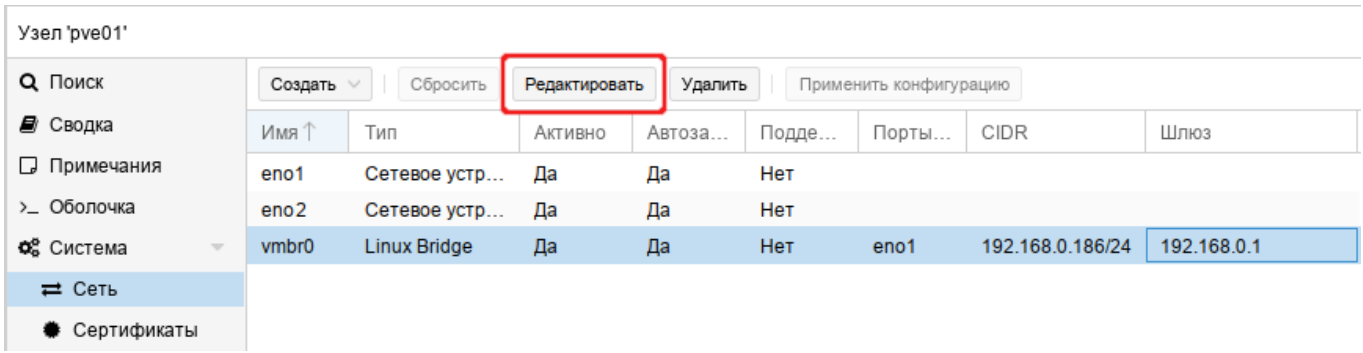

Рис. 228 – Мост vmbr0

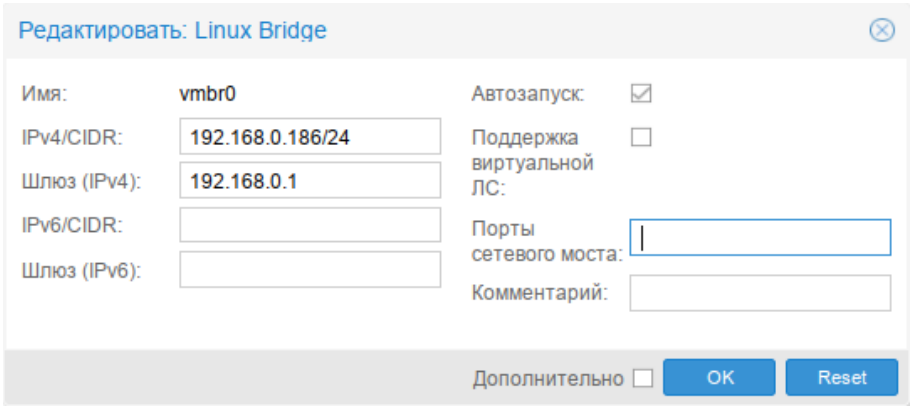

Рис. 229 – Редактирование параметров моста vmbr0

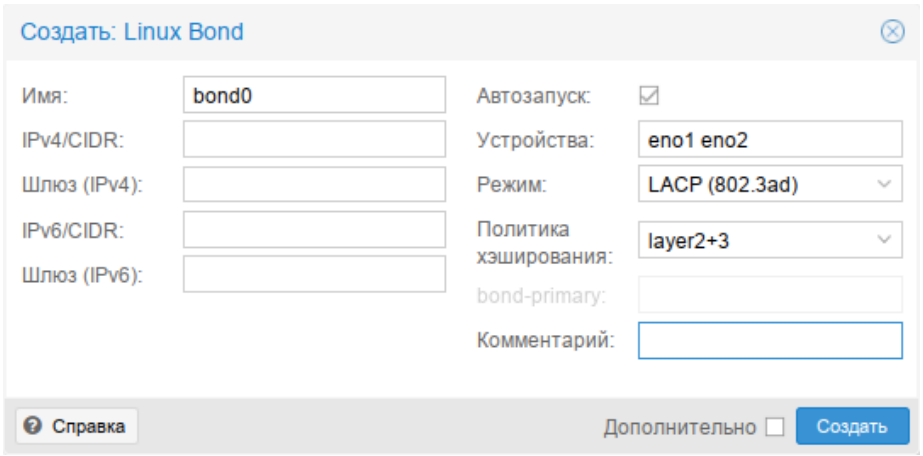

Рис. 230 – Редактирование параметров объединения bond0

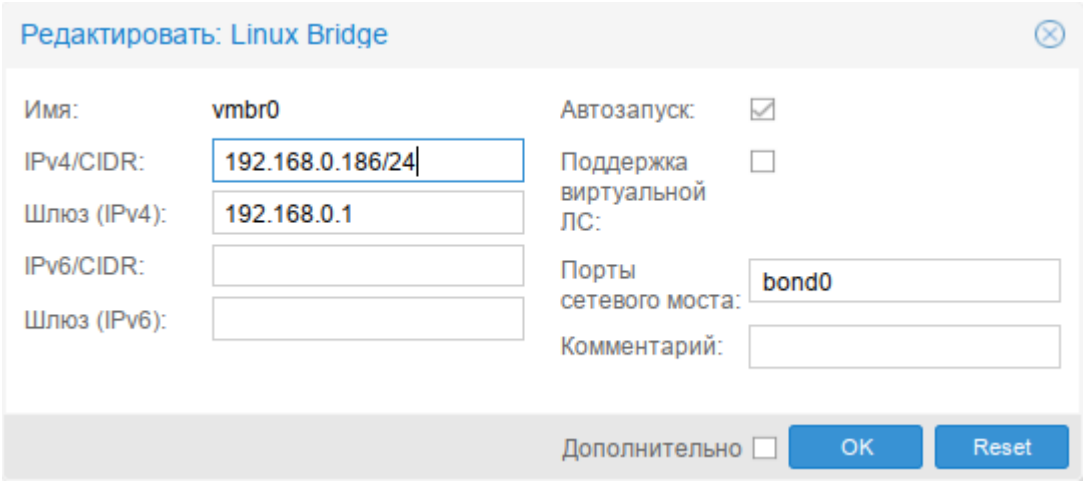

Рис. 231 – Редактирование параметров моста vmbr0

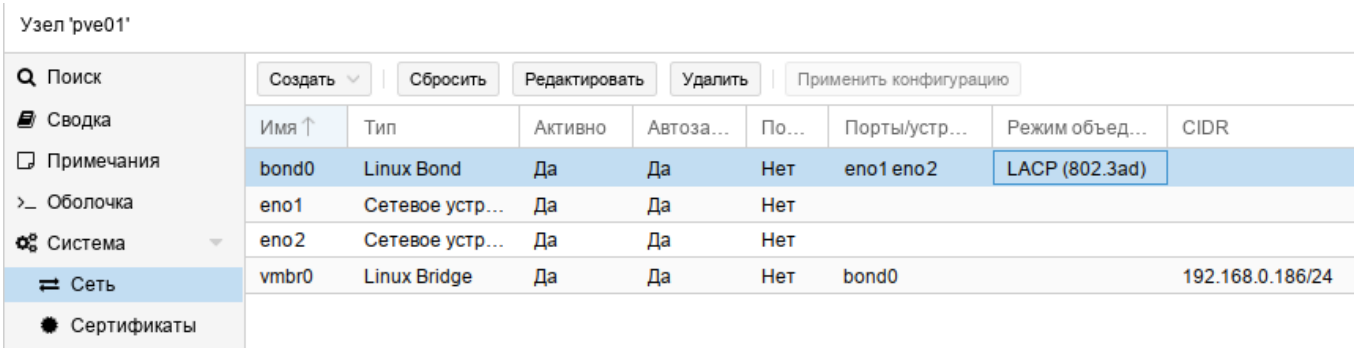

Рис. 232 – Агрегированный bond-интерфейс в качестве порта моста

Для настройки OVS Bond необходимо выполнить следующие действия:

1) перейти в раздел «Сеть», выбрать существующий мост vmbr0 и нажать на кнопку «Редактировать» (рис. 233);
2) в открывшемся окне удалить содержимое поля «Порты сетевого моста» и нажать на кнопку «ОК» (рис. 234);
3) нажать на кнопку «Создать» (рис. 219) и в выпадающем меню выбрать пункт «OVS Bond»;
4) в открывшемся окне указать имя агрегированного интерфейса, в выпадающем списке «Режим» выбрать режим агрегации, в поле «Устройства» указать сетевые интерфейсы, которые будут входить в объединение, в выпадающем списке «OVS Bridge» выбрать мост, в который должен добавиться созданный интерфейс и нажать на кнопку «Создать» (рис. 235);
5) для применения изменений нажать на кнопку «Применить конфигурацию»;
6) получившаяся конфигурация показана на рис. 236.

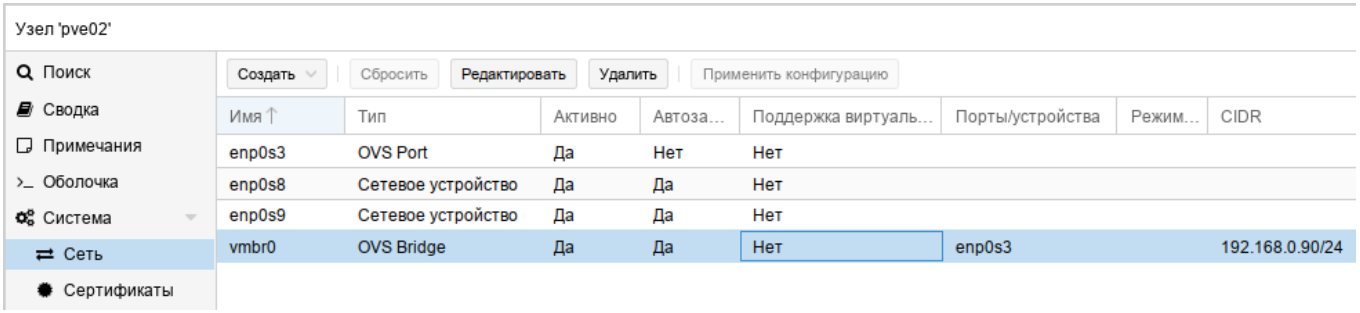

Рис. 233 – Мост vmbr0

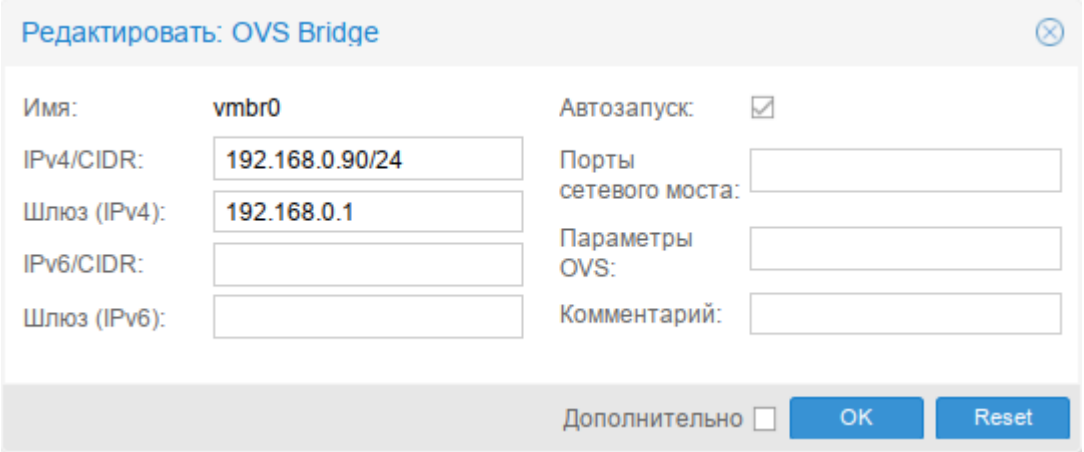

Рис. 234 – Редактирование параметров моста vmbr0

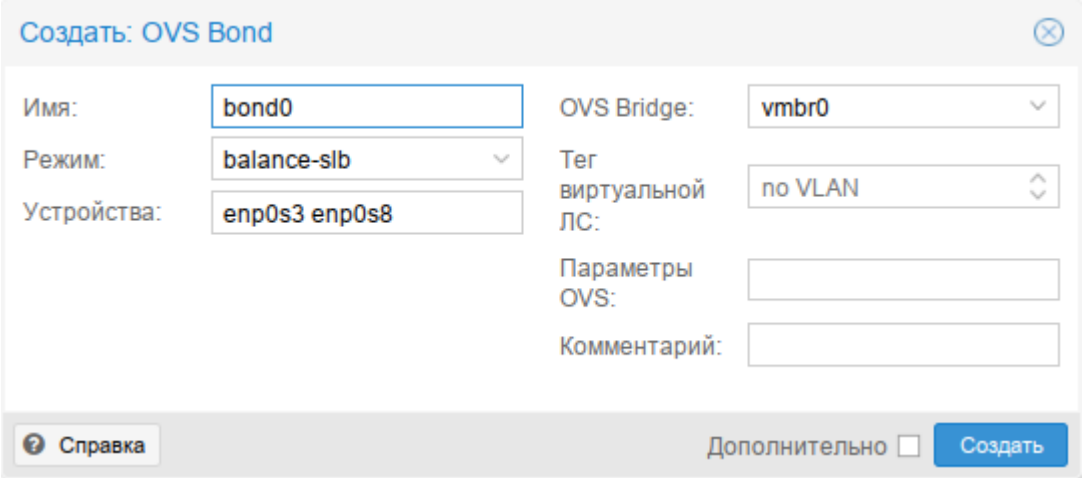

Рис. 235 – Редактирование параметров объединения bond0

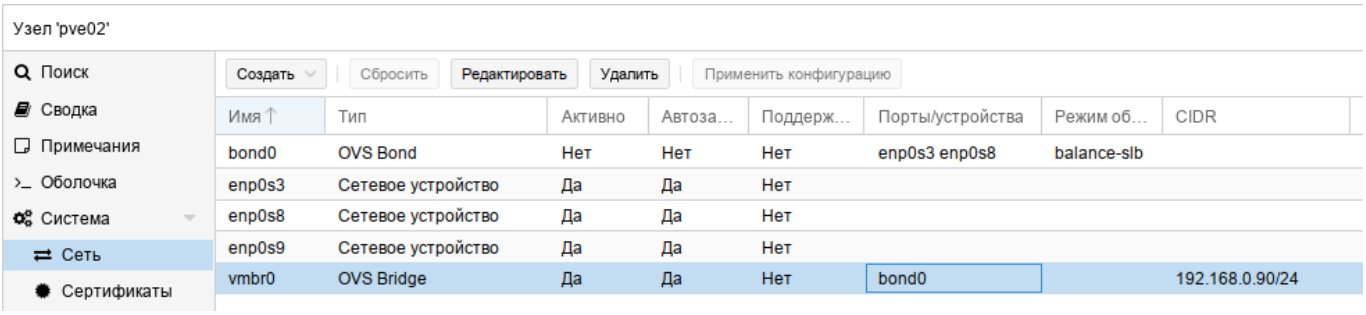

Рис. 236 – Агрегированный bond-интерфейс в качестве порта моста

8.5.4.2.2.2. Настройка в командной строке

Исходное состояние: мост vmbr0 на интерфейсе enp0s3. Необходимо создать агрегированный интерфейс bond0 (enp0s3 и enp0s8) и включить его в мост vmbr0.

Для создания Linux Bond, необходимо выполнить следующие действия:

1) создать агрегированный интерфейс bond0 на интерфейсах enp0s3 и enp0s8, выполнив следующие команды:

```
# mkdir /etc/net/ifaces/bond0
# cat <<EOF > /etc/net/ifaces/bond0/options
BOOTPROTO=static
CONFIG_WIRELESS=no
CONFIG_IPV4=yes
HOST='enp0s3 enp0s8'
ONBOOT=yes
TYPE=bond
BONDOPTIONS='xmit-hash-policy=layer2+3 lacp_rate=1 miimon=100'
BONDMODE=4
EOF
```

где:

- `BONDMODE=4` – режим агрегации LACP (802.3ad);
- `HOST='enp0s3 enp0s8'` – интерфейсы, которые будут входить в объединение;
- `xmit-hash-policy=layer2+3` – определяет режим выбора каналов;
- `lacp_rate=1` – определяет, что управляющие пакеты LACPDU отправляются по каналу LACP каждую секунду;
- `miimon=100` – определяет, как часто производится мониторинг MII (Media Independent Interface);

2) в настройках Ethernet-моста vmbr0 (файл `/etc/net/ifaces/vmbr0/options`) в опции HOST указать интерфейс bond0:

```
BOOTPROTO=static
BRIDGE_OPTIONS="stp_state 0"
CONFIG_IPV4=yes
HOST='bond0'
ONBOOT=yes
TYPE=bri
```

3) перезапустить службу network, чтобы изменения вступили в силу:

```
# systemctl restart network
```

Для создания OVS Bond, необходимо выполнить следующие действия:

1) начальная конфигурации:

```
# ovs-vsctl show
6b1add02-fb20-48e6-b925-260bf92fa889
    Bridge vmbr0
        Port enp0s3
            Interface enp0s3
        Port vmbr0
            Interface vmbr0
                type: internal
    ovs_version: "2.17.6"
```

2) привести настройки сетевого интерфейса enp0s3 (файл `/etc/net/ifaces/enp0s3/options`) к виду:

```
TYPE=eth
CONFIG_WIRELESS=no
BOOTPROTO=static
CONFIG_IPV4=yes
```

3) создать агрегированный интерфейс bond0 на интерфейсах enp0s3 и enp0s8, выполнив следующие команды:

```
# mkdir /etc/net/ifaces/bond0
# cat <<EOF > /etc/net/ifaces/bond0/options
BOOTPROTO=static
BRIDGE=vmbr0
CONFIG_IPV4=yes
HOST='enp0s3 enp0s8'
OVS_OPTIONS='other_config:bond-miimon-interval=100
bond_mode=balance-slb'
TYPE=ovsbond
EOF
```

где:

- `bond_mode=balance-slb` – режим агрегации;
- `HOST='enp0s3 enp0s8'` – интерфейсы, которые будут входить в объединение;
- `BRIDGE=vmbr0` – мост, в который должен добавиться созданный интерфейс;
- `other_config:bond-miimon-interval=100` – определяет, как часто производится мониторинг MII (Media Independent Interface).

В опции HOST настроек моста vmbr0 (файл `/etc/net/ifaces/vmbr0/options`) указать bond0:

```
BOOTPROTO=static
CONFIG_IPV4=yes
HOST=bond0
ONBOOT=yes
TYPE=ovsbr
```

- перезапустить службу network, чтобы изменения вступили в силу:

```
# systemctl restart network
```

- проверка конфигурации:

```
# ovs-vsctl show
6b1add02-fb20-48e6-b925-260bf92fa889
    Bridge vmbr0
        Port bond0
            Interface enp0s3
            Interface enp0s8
        Port vmbr0
            Interface vmbr0
                type: internal
    ovs_version: "2.17.6"
```

8.5.5. Настройка VLAN

Виртуальные сети (VLAN) являются сетевым стандартом на основе 802.1q для создания логических разделов на одном и том же сетевом интерфейсе для изоляции обмена множества сетей.

Примечание. На стороне физического коммутатора порт должен быть настроен как trunk, от него должен приходить тэгированный трафик 802.1q. Если на коммутаторе сделана агрегация портов (Portchannel или Etherchannel), то параметр `Trunk` выставляется на это новом интерфейсе.

8.5.5.1. Мост с поддержкой VLAN

Если используется Linux Bridge, то для возможности использования тегов VLAN в настройках ВМ, необходимо включить поддержку VLAN для моста. Для этого в веб-интерфейсе в настройках моста следует установить отметку в поле «Поддержка виртуальной ЛС» (рис. 237).

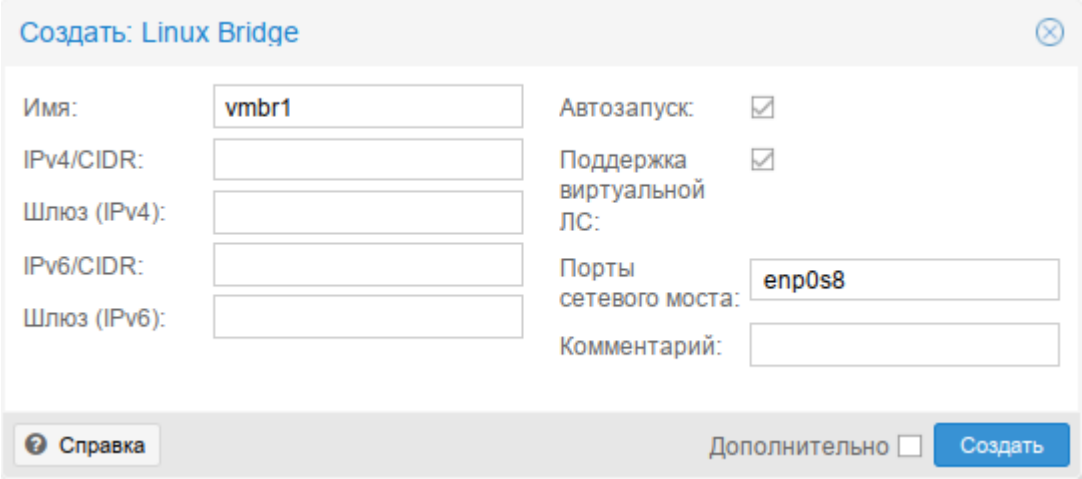

Рис. 237 – Настройки моста Linux Bridge

Если используется OVS Bridge, то никаких дополнительных настроек не требуется.

Тег VLAN можно указать в настройках сетевого интерфейса при создании ВМ (рис. 238), либо отредактировав параметры сетевого устройства.

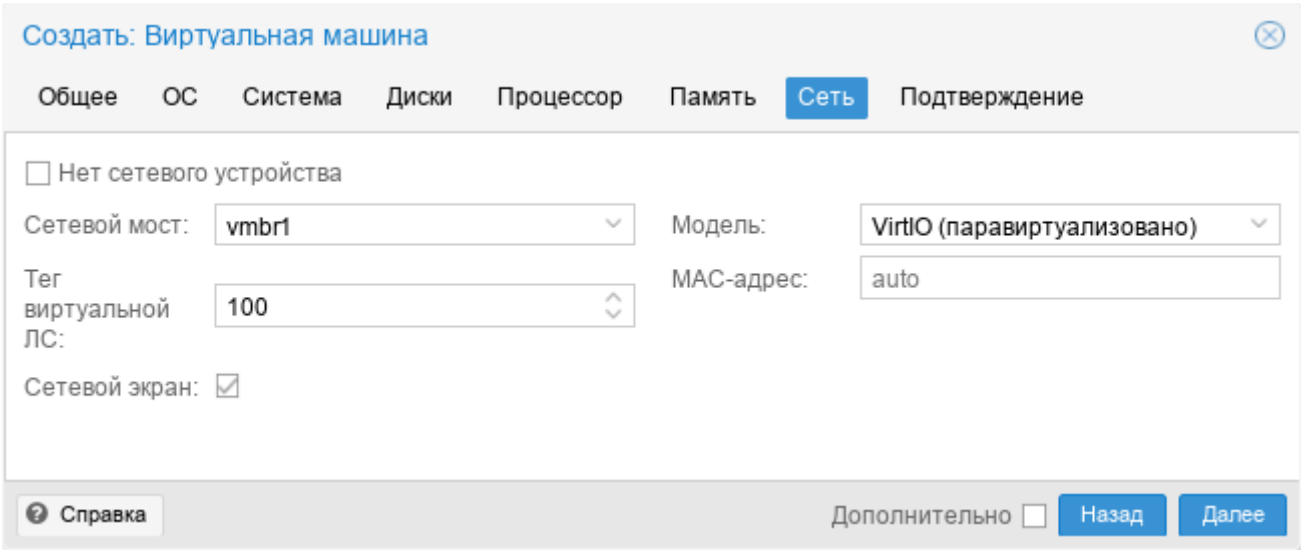

Рис. 238 – Настройки сетевого интерфейса ВМ

8.5.5.2. Мост на VLAN

Можно создать конфигурацию VLAN `<интерфейс>.<vlan tag>` (например, enp0s8.100), этот VLAN включить в мост Linux Bridge и указывать этот мост в настройках сетевого интерфейса ВМ.

Для создания такой конфигурации, необходимо выполнить следующие действия:

- настроить VLAN с ID 100 на интерфейсе enp0s8, выполнив следующие команды (в опции HOST нужно указать тот интерфейс, на котором будет настроен VLAN):

```
# mkdir /etc/net/ifaces/enp0s8.100
# cat <<EOF > /etc/net/ifaces/enp0s8.100/options
BOOTPROTO=static
CONFIG_WIRELESS=no
CONFIG_IPV4=yes
HOST=enp0s8
ONBOOT=yes
TYPE=vlan
VID=100
EOF
```

- настроить Ethernet-мост vmbr1, выполнив следующие команды (в опции HOST нужно указать VLAN-интерфейс):

```
# mkdir /etc/net/ifaces/vmbr1
# cat <<EOF > /etc/net/ifaces/vmbr1/options
BOOTPROTO=static
CONFIG_WIRELESS=no
CONFIG_IPV4=yes
HOST='enp0s8.100'
ONBOOT=yes
TYPE=bri
EOF
```

- в файле `/etc/net/ifaces/vmbr1/ipv4address`, если это необходимо, можно указать IP-адрес для интерфейса моста:

```
# echo "192.168.10.3/24" > /etc/net/ifaces/vmbr1/ipv4address
```

- перезапустить службу network, чтобы изменения вступили в силу:

```
# systemctl restart network
```

Теперь в настройках сетевого интерфейса ВМ можно указать сетевой мост vmbr1 (рис. 239). Трафик через этот интерфейс будет помечен тегом 100.

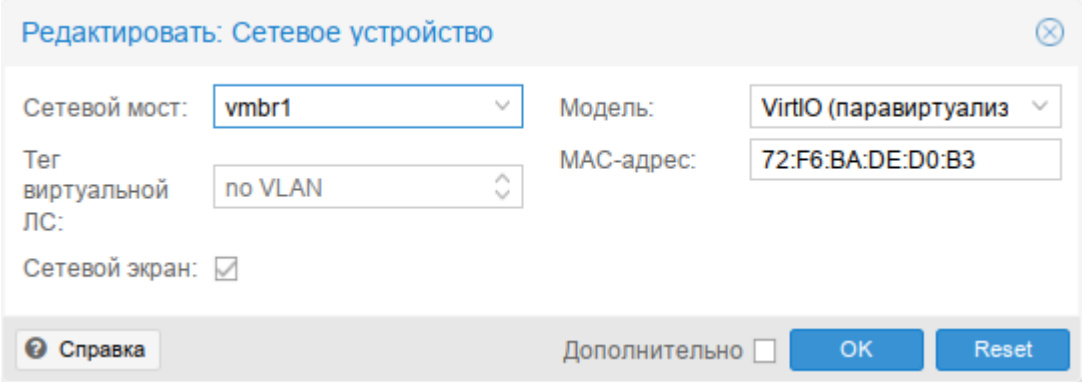

Рис. 239 – Настройки сетевого интерфейса ВМ

8.6. 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-образы» для загрузки ISO-образов (рис. 240) или на вкладку «Шаблоны контейнеров» для загрузки шаблонов LXC;
3) для загрузки образа (шаблона) с локального компьютера следует нажать на кнопку «Отправить». В открывшемся окне нажать на кнопку «Выбрать файл», выбрать файл с ISO-образом и нажать на кнопку «Отправить» (рис. 241). 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 на кнопку «Загрузить по URL-адресу». В открывшемся окне указать ссылку на образ (шаблон), нажать на кнопку «Запрос URL-адреса», для того чтобы получить метаинорфмацию о файле, нажать на кнопку «Загрузка» для старта загрузки файла в хранилище (рис. 242).

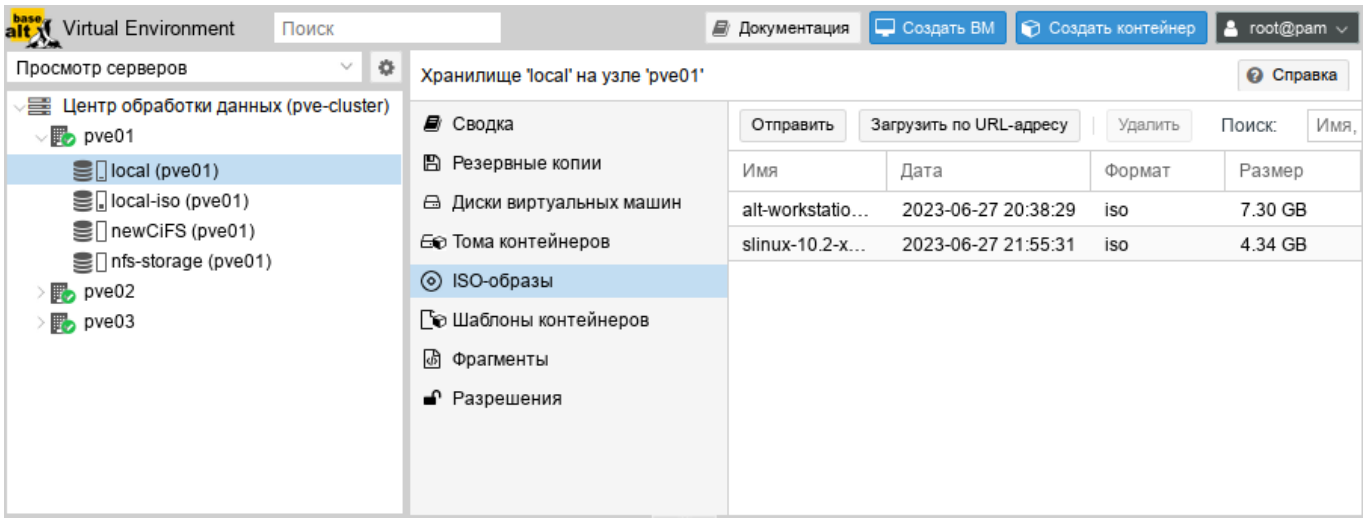

Рис. 240 – Локальное хранилище. Вкладка «ISO-образы»

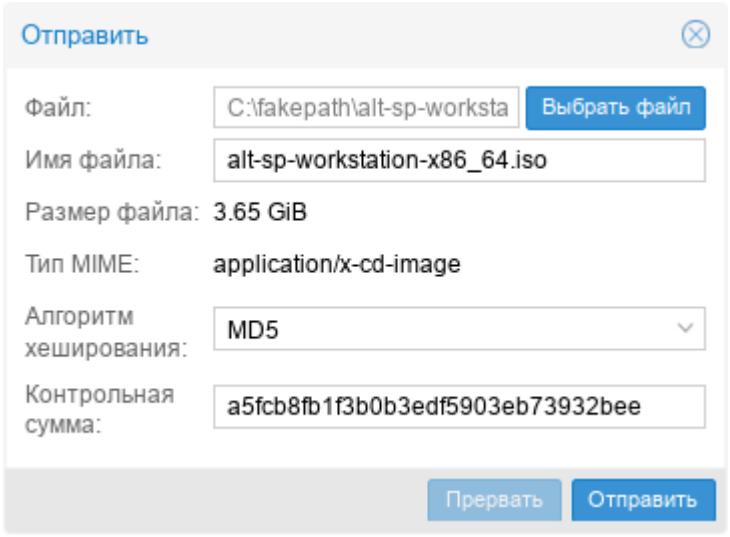

Рис. 241 – Выбор образа

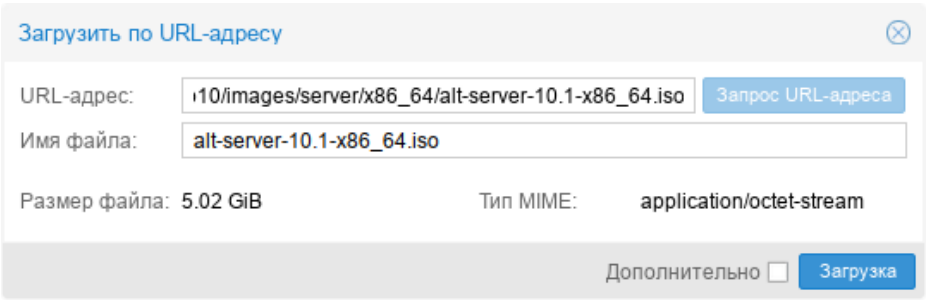

Рис. 242 – Выбор образа для загрузки файла с сервера

Для удаления ISO-образа или шаблона LXC следует выбрать файл из списка в хранилище (рис. 240) и нажать на кнопку «Удалить».

PVE предоставляет базовые шаблоны для некоторых дистрибутивов Linux. Эти шаблоны можно загрузить в веб-интерфейсе (кнопка «Шаблоны») или в командной строке (утилита `pveam`).

Загрузка базового шаблона в веб-интерфейсе:

- запустить обновление списка доступных шаблонов (например, на вкладке «Оболочка»):

```
# pveam update
```

- выбрать хранилище;
- перейти на вкладку «Шаблоны контейнеров» и нажать на кнопку «Шаблоны» (рис. 243);
- в открывшемся окне выбрать шаблон и нажать на кнопку «Загрузка» (рис. 244).

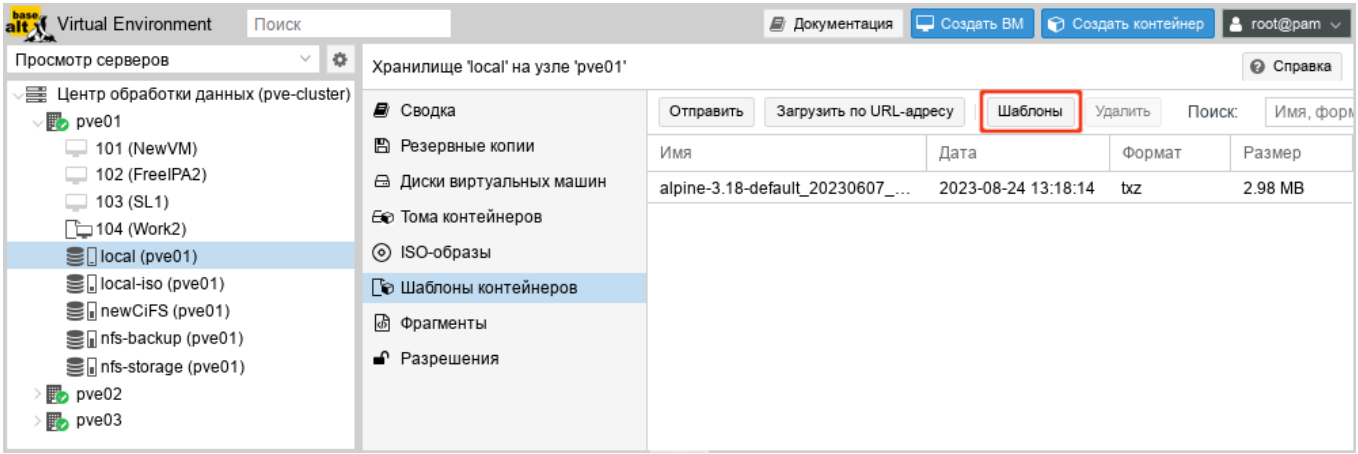

Рис. 243 – Вкладка «Шаблоны контейнеров»

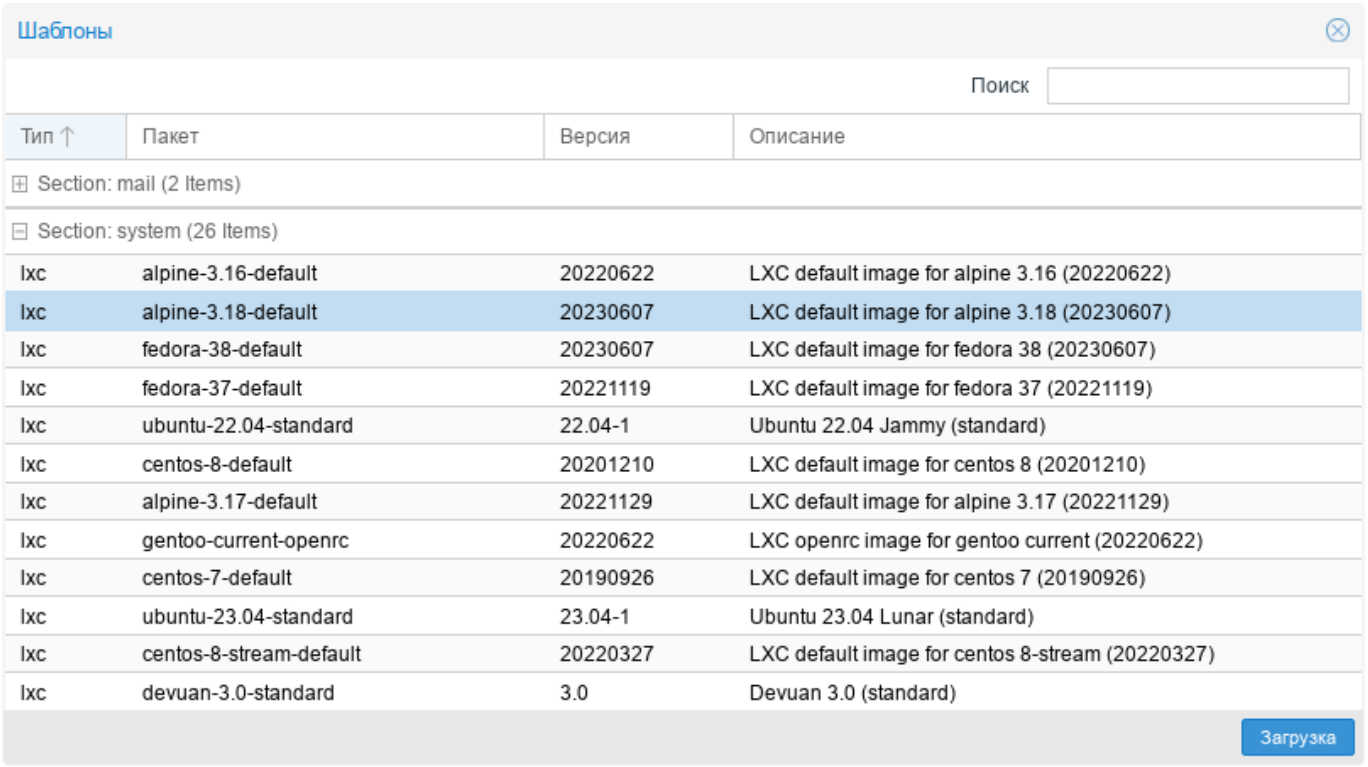

Рис. 244 – Выбор шаблона для загрузки

Загрузка базового шаблона в консоли:

- запустить обновление списка доступных шаблонов:

```
# pveam update
update successful
```

- просмотреть список доступных шаблонов:

```
# pveam available
mail          proxmox-mailgateway-7.3-standard_7.3-1_amd64.tar.zst
mail          proxmox-mailgateway-8.0-standard_8.0-1_amd64.tar.zst
system        almalinux-8-default_20210928_amd64.tar.xz
system        almalinux-9-default_20221108_amd64.tar.xz
system        alpine-3.16-default_20220622_amd64.tar.xz
…
```

- загрузить шаблон в хранилище `local`:

```
# pveam download local almalinux-9-default_20221108_amd64.tar.xz
```

- просмотреть список загруженных шаблонов в хранилище `local`:

```
# pveam list local
NAME                                                    SIZE
local:vztmpl/almalinux-9-default_20221108_amd64.tar.xz      97.87MB
```

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 20 показаны каталоги для локального хранилища. В таблице 21 показаны каталоги для всех других хранилищ.

Т а б л и ц а 20 – 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 21 – Каталоги общих хранилищ

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

8.7. Виртуальные машины на базе KVM

8.7.1. 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. 240).

Для создания ВМ необходимо нажать на кнопку «Создать ВМ», расположенную в правом верхнем углу веб-интерфейса PVE (рис. 245).

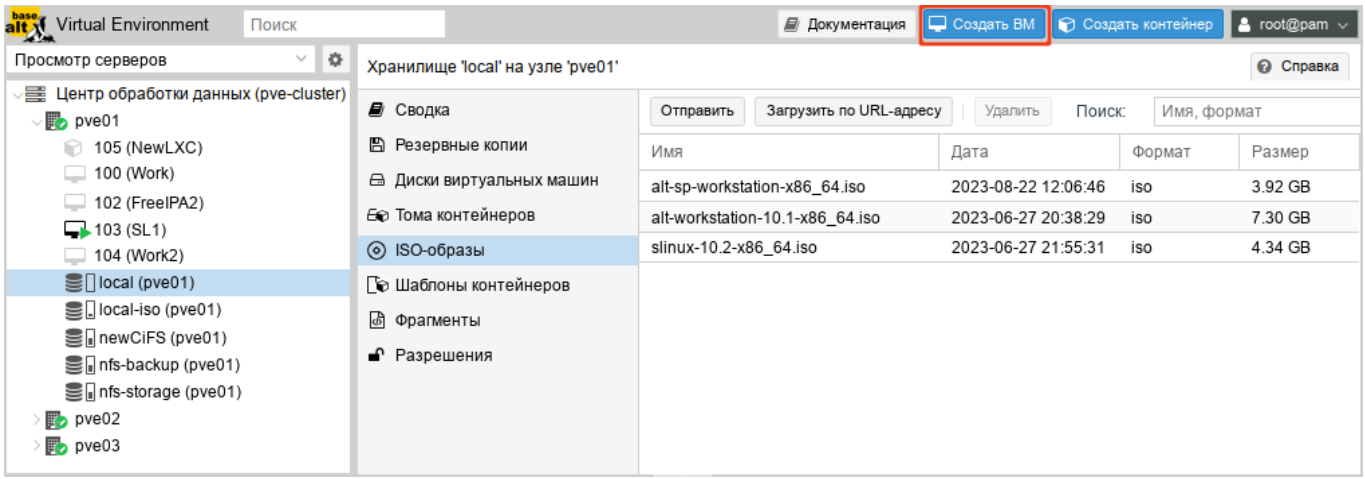

Рис. 245 – Кнопка «Создать ВМ»

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

На вкладке «Общее» необходимо указать (рис. 246):

- «Узел» – 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» – текстовая строка названия ВМ;
- «Пул ресурсов» – логическая группа ВМ. Чтобы иметь возможность выбора, пул должен быть предварительно создан.

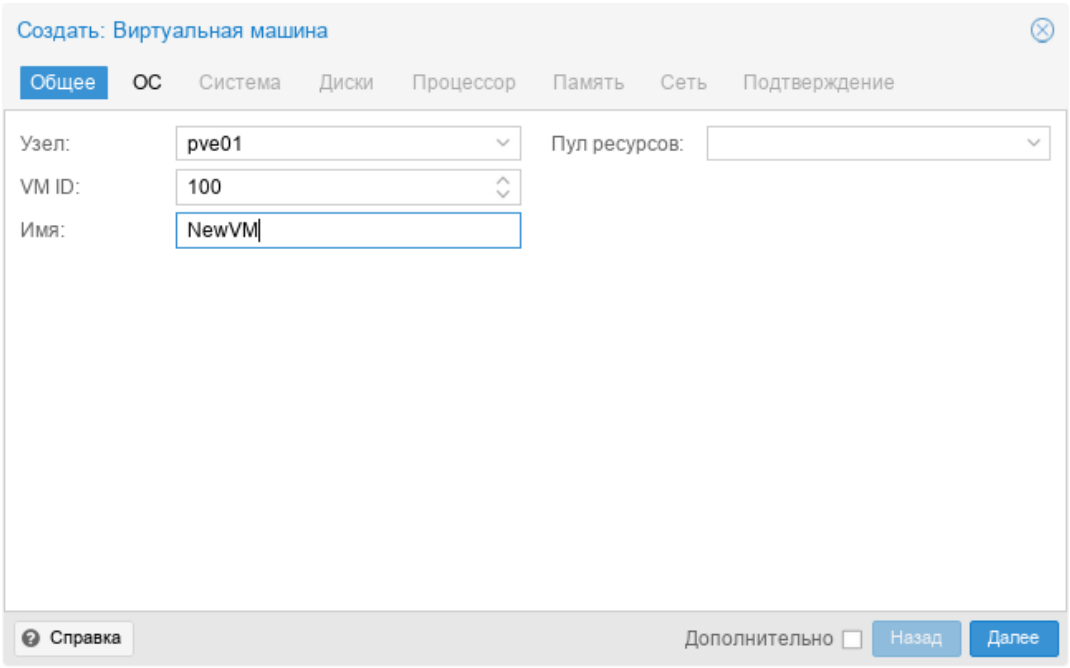

Рис. 246 – Вкладка «Общее»

П р и м е ч а н и е . Настроить диапазон, из которого выбираются новые VM ID при создании ВМ или контейнера можно, выбрав на вкладке «Центр обработки данных» → «Параметры» пункт «Следующий свободный диапазон ID виртуальных машин» (рис. 247). Установка нижнего значения («Нижний предел») равным верхнему («Верхний предел») полностью отключает автоподстановку VM ID.

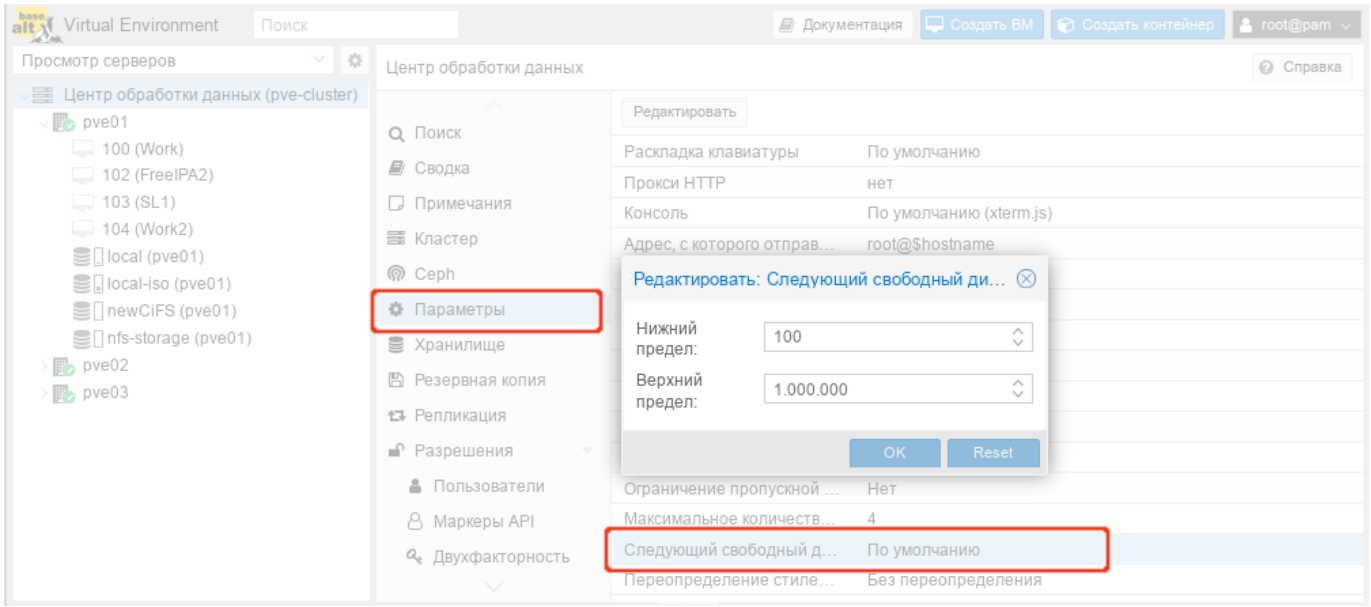

Рис. 247 – Настройка диапазона VM ID

На вкладке «ОС» необходимо указать источник установки ОС и тип ОС (рис. 248).

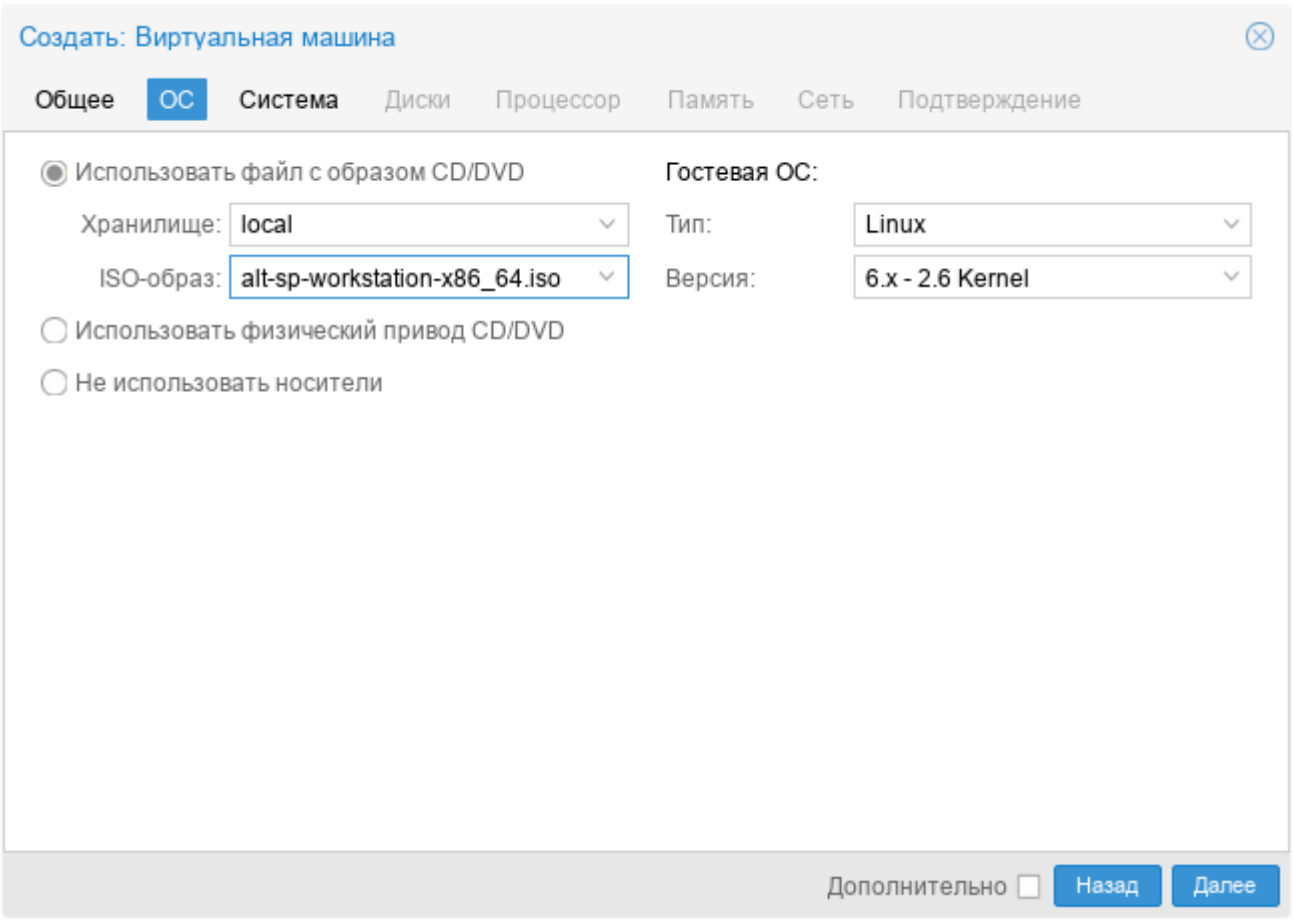

Рис. 248 – Вкладка «ОС»

В качестве источника установки ОС можно указать:

- «Использовать файл с образом CD/DVD» – использовать уже загруженный в хранилище ISO-образ (рис. 249);
- «Использовать физический привод CD/DVD» – использовать физический диск хоста PVE;
- «Не использовать носители» – не использовать ISO-образ или физический носитель.

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

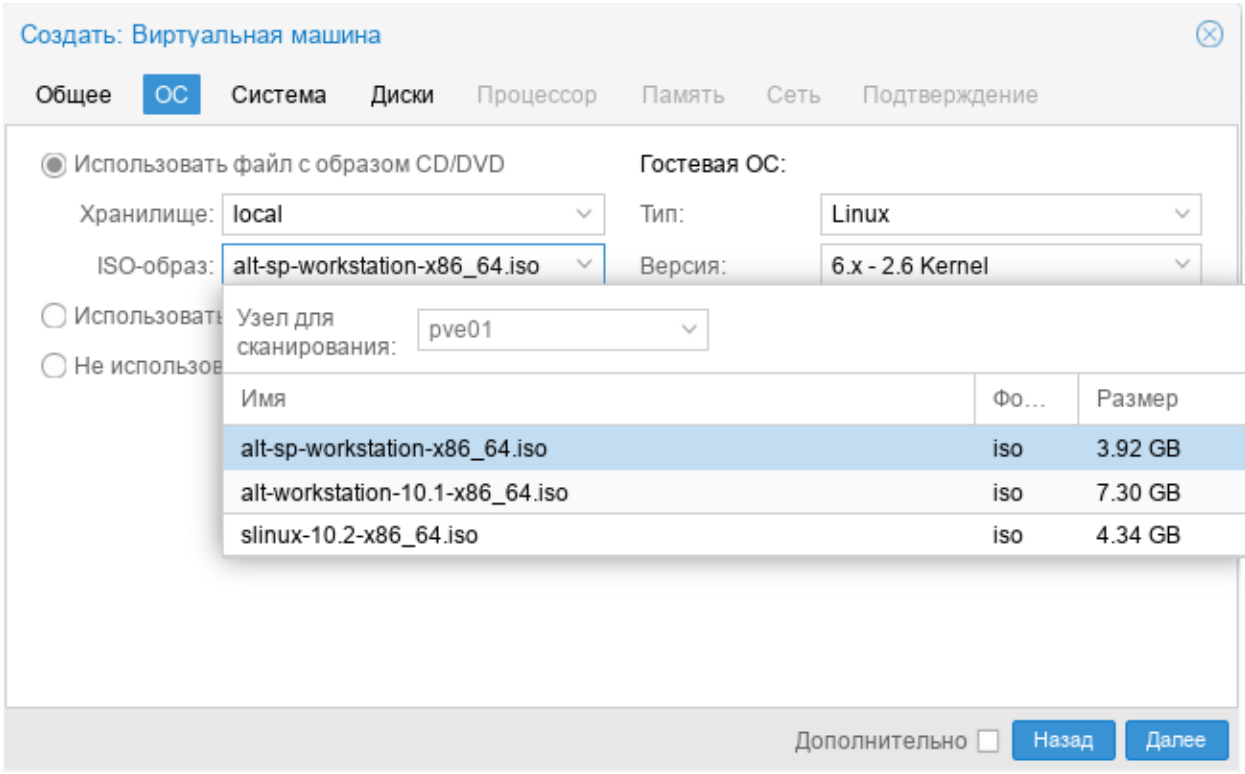

Рис. 249 – Выбор ISO-образа

На следующем этапе (вкладка «Система») можно выбрать видеокарту, контроллер SCSI, указать, нужно ли использовать агент QEMU (рис. 250).

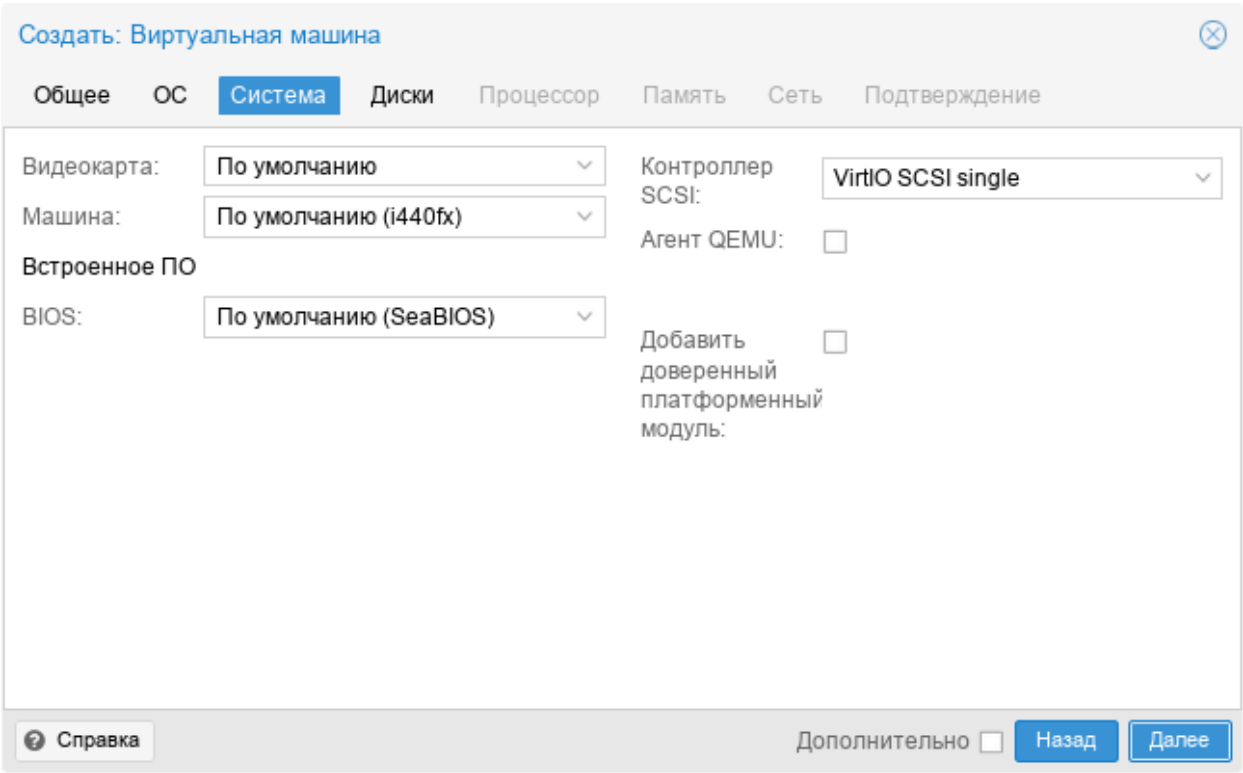

Рис. 250 – Вкладка «Система»

Подробнее о выборе видеокарты см. п. 8.7.5.2 «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. 251) для сохранения порядка загрузки, должен быть добавлен диск EFI (см. «BIOS и UEFI»).

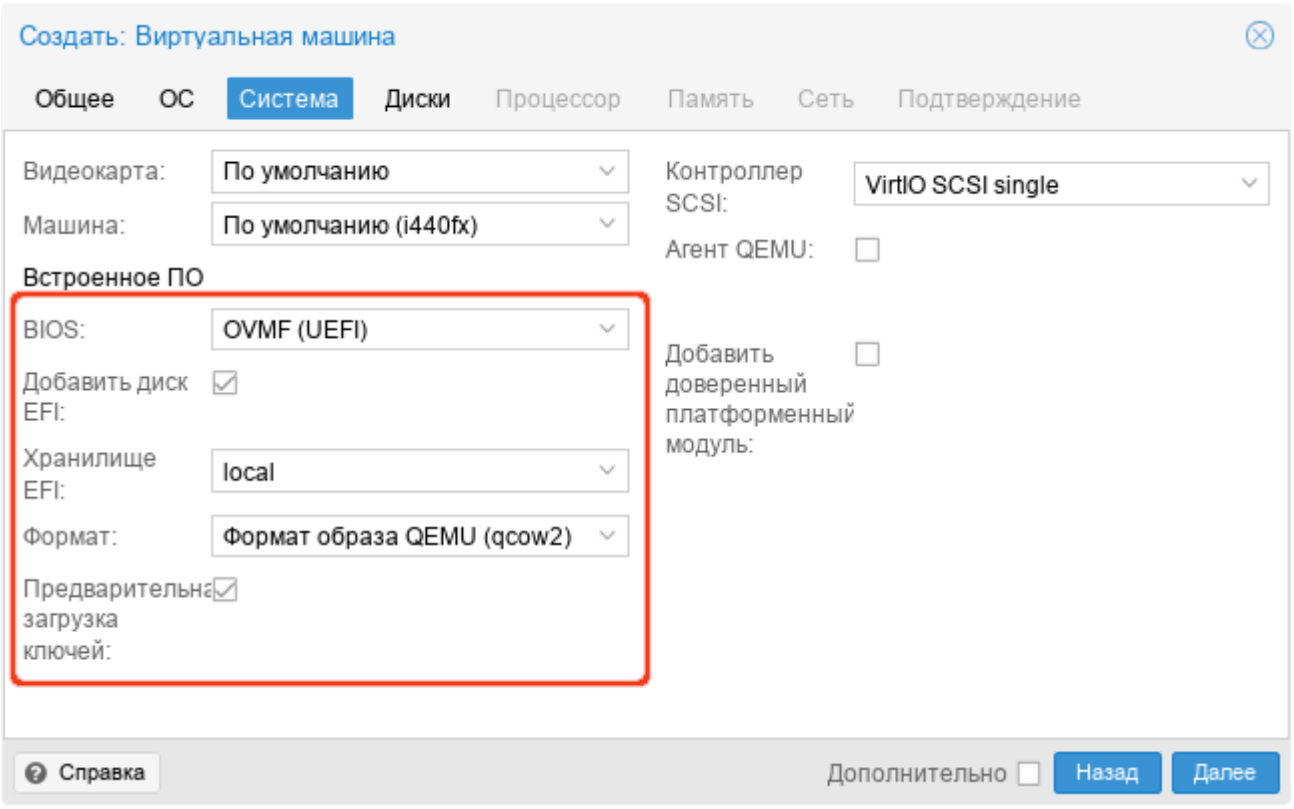

Рис. 251 – Выбор прошивки OVMF

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» содержит следующие настройки (рис. 252):

- «Шина/Устройство» – тип устройства виртуального диска. Допустимые значения: «IDE», «SATA», «VirtIO Block» и «SCSI» (по умолчанию). Можно также указать идентификатор устройства;
- «Хранилище» – выбор хранилища для размещения виртуального диска (выбор хранилища определяет возможный формат образа диска);
- «Размер диска» (GiB) – размер виртуального диска в Гбайт;

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

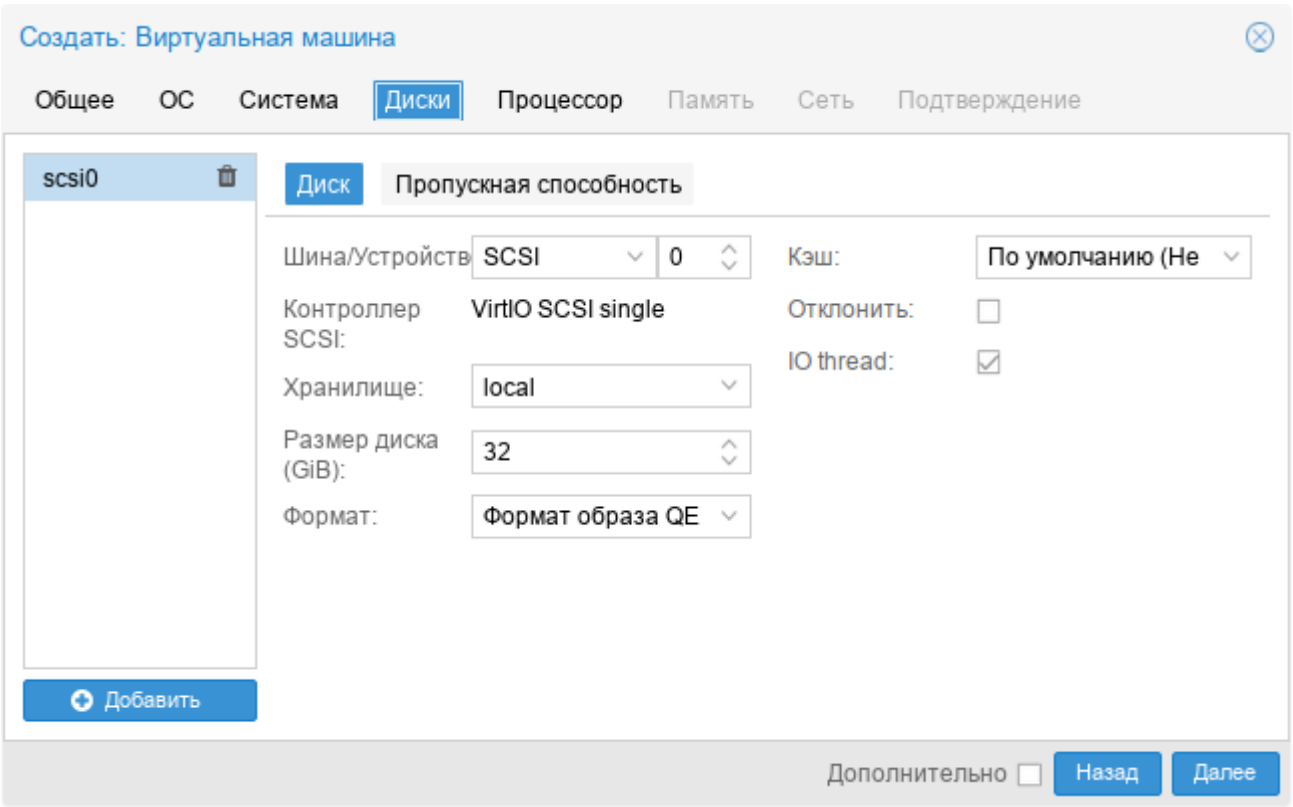

Рис. 252 – Вкладка «Жесткий диск»

В мастере создания ВМ можно добавить несколько дисков (рис. 253) (кнопка «Добавить»).

Максимально можно добавить: 31 диск SCSI, 16 – VirtIO, 6 – SATA, 4 – IDE.

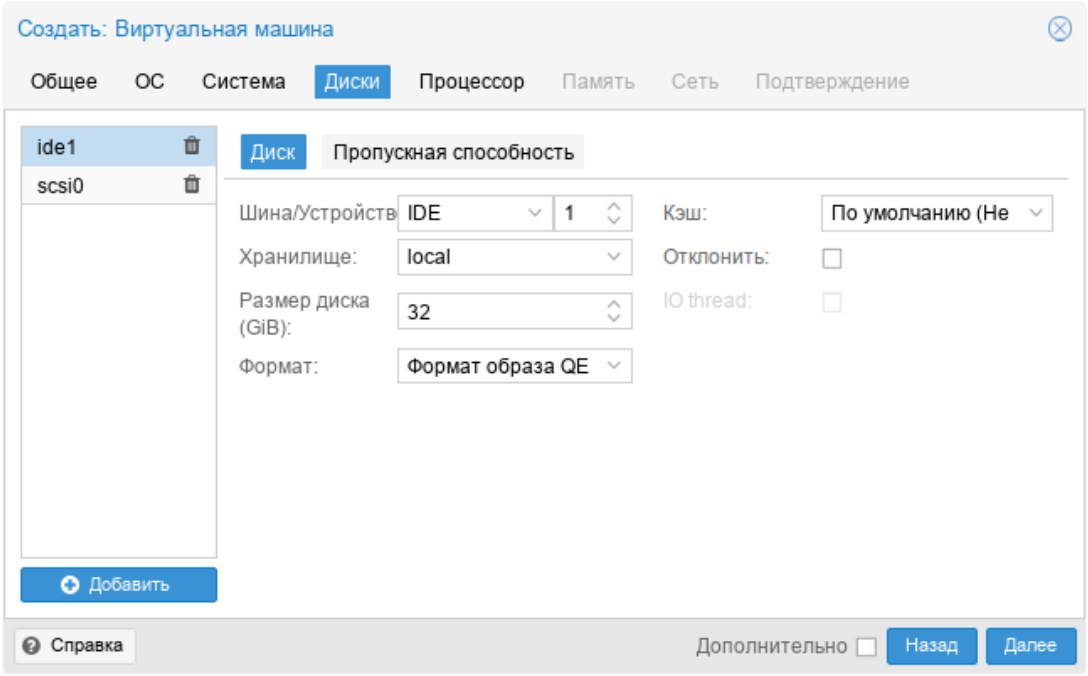

Рис. 253 – Вкладка «Жесткий диск». Создание нескольких дисков

В разделе «Пропускная способность» (рис. 254) можно задать максимальную скорость чтения/записи с диска (в мегабайтах в секунду или в операциях в секунду).

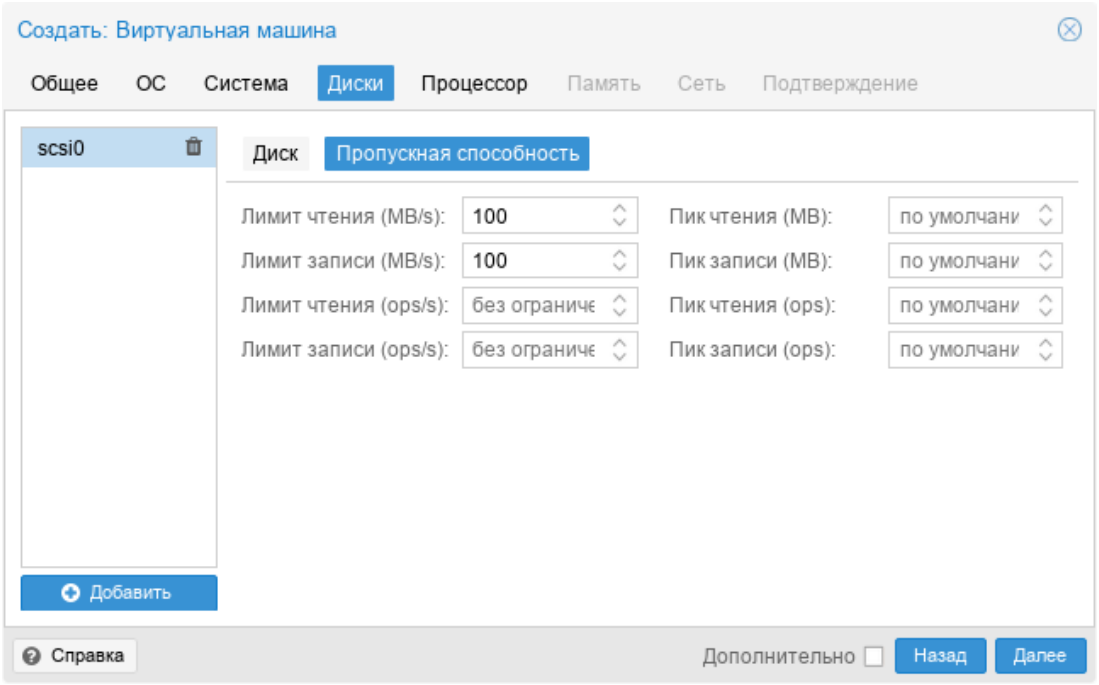

Рис. 254 – Скорость чтения/записи с диска

Примечание. SCSI и VirtIO дискам может быть добавлен атрибут read-only (рис. 255) (отметка «Только для чтения»).

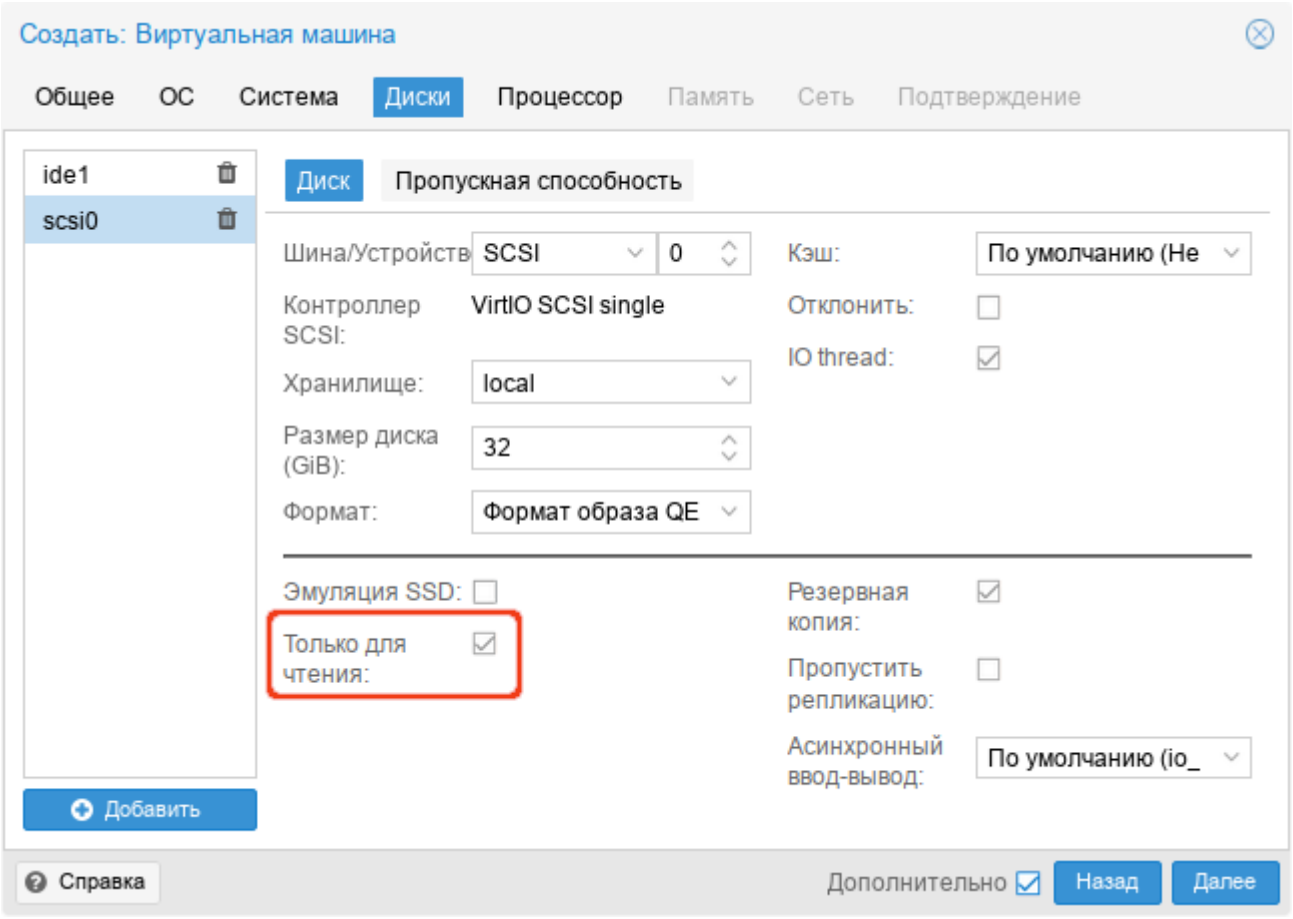

Рис. 255 – Отметка «Только для чтения»

На следующем этапе настраивается процессор (CPU) (рис. 256):

- «Сокеты» – число сокетов ЦПУ для ВМ;
- «Ядра» – число ядер для ВМ;
- «Тип» – тип процессора.

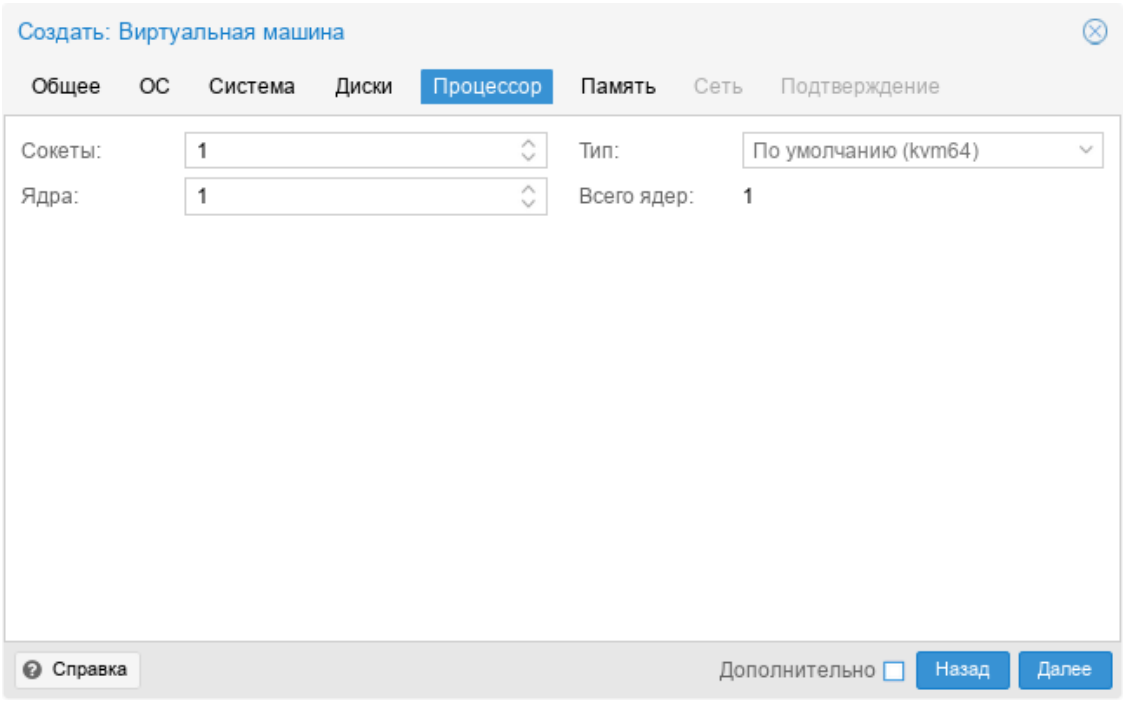

Рис. 256 – Вкладка «Процессор»

На вкладке «Память» (рис. 257) необходимо указать объем оперативной памяти, выделяемой ВМ.

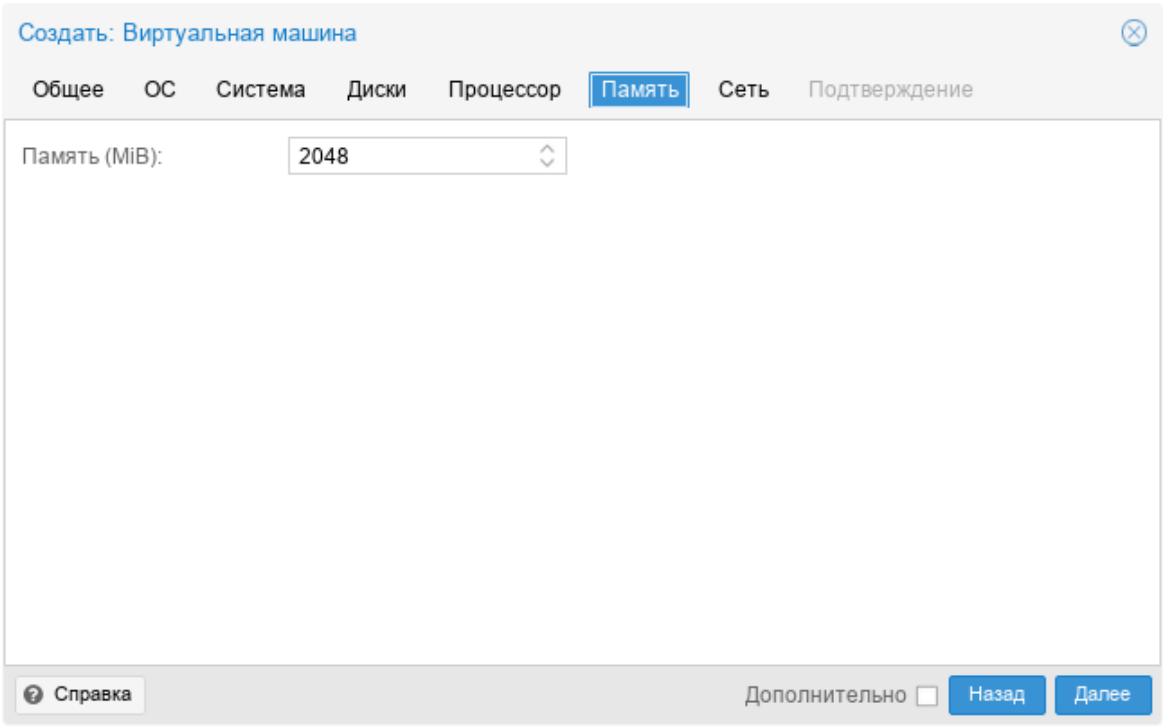

Рис. 257 – Вкладка «Память»

Вкладка «Сеть» содержит следующие настройки (рис. 258):

- «Нет сетевого устройства» – выбор данного параметра пропускает шаг настройки сетевой среды;
- «Сетевой мост» – 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;
- «Тег виртуальной ЛС» – применяется для установки идентификатора VLAN для данного виртуального интерфейса;
- «Сетевой экран» – разрешает использование для ВМ встроенных межсетевых экранов;
- «Модель» – тип драйвера сетевого устройства. Для максимальной сетевой производительности ВМ следует выбрать пункт «VirtIO (паравиртуализированно)»;
- «MAC-адрес» – по умолчанию PVE автоматически создает уникальный MAC-адрес для сетевого интерфейса. Если есть такая необходимость, можно ввести пользовательский MAC-адрес вручную.

Последняя вкладка «Подтверждение» отобразит все введенные или выбранные значения для ВМ (рис. 259). Для создания ВМ следует нажать на кнопку «Готово». Если необходимо внести изменения в параметры ВМ, можно перейти по вкладкам назад. Если отметить пункт «Запуск после создания» ВМ будет запущена сразу после создания.

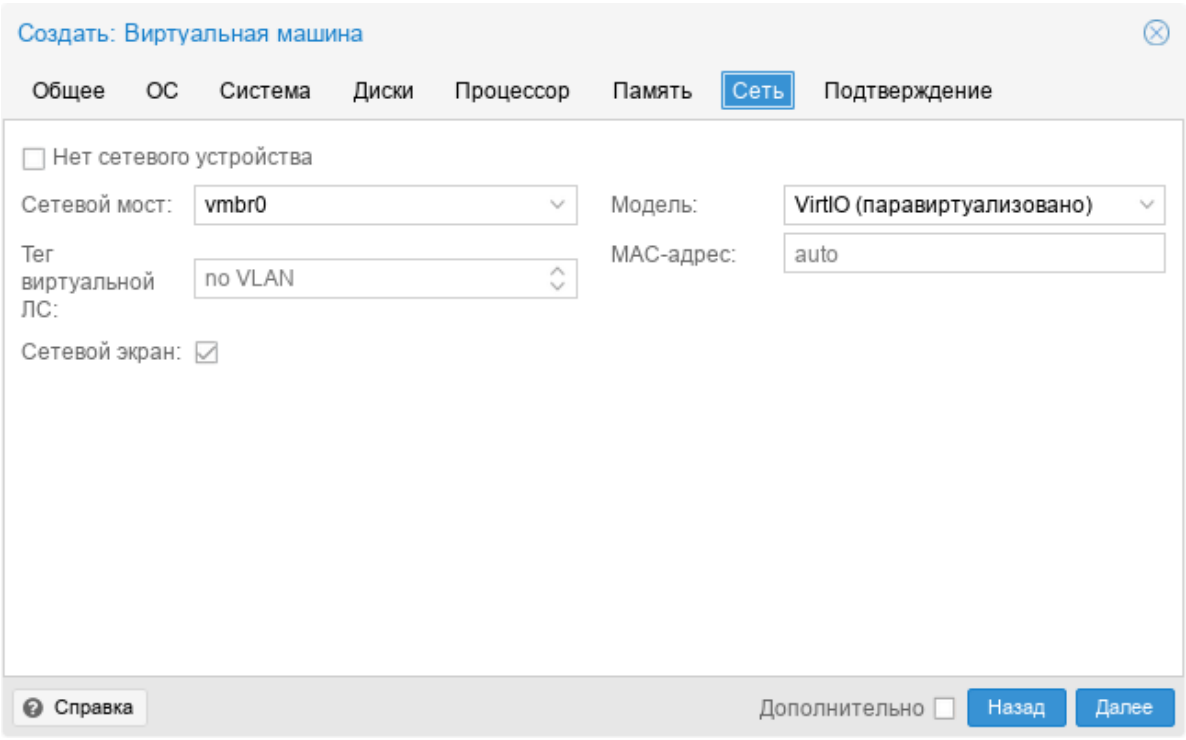

Рис. 258 – Вкладка «Сеть»

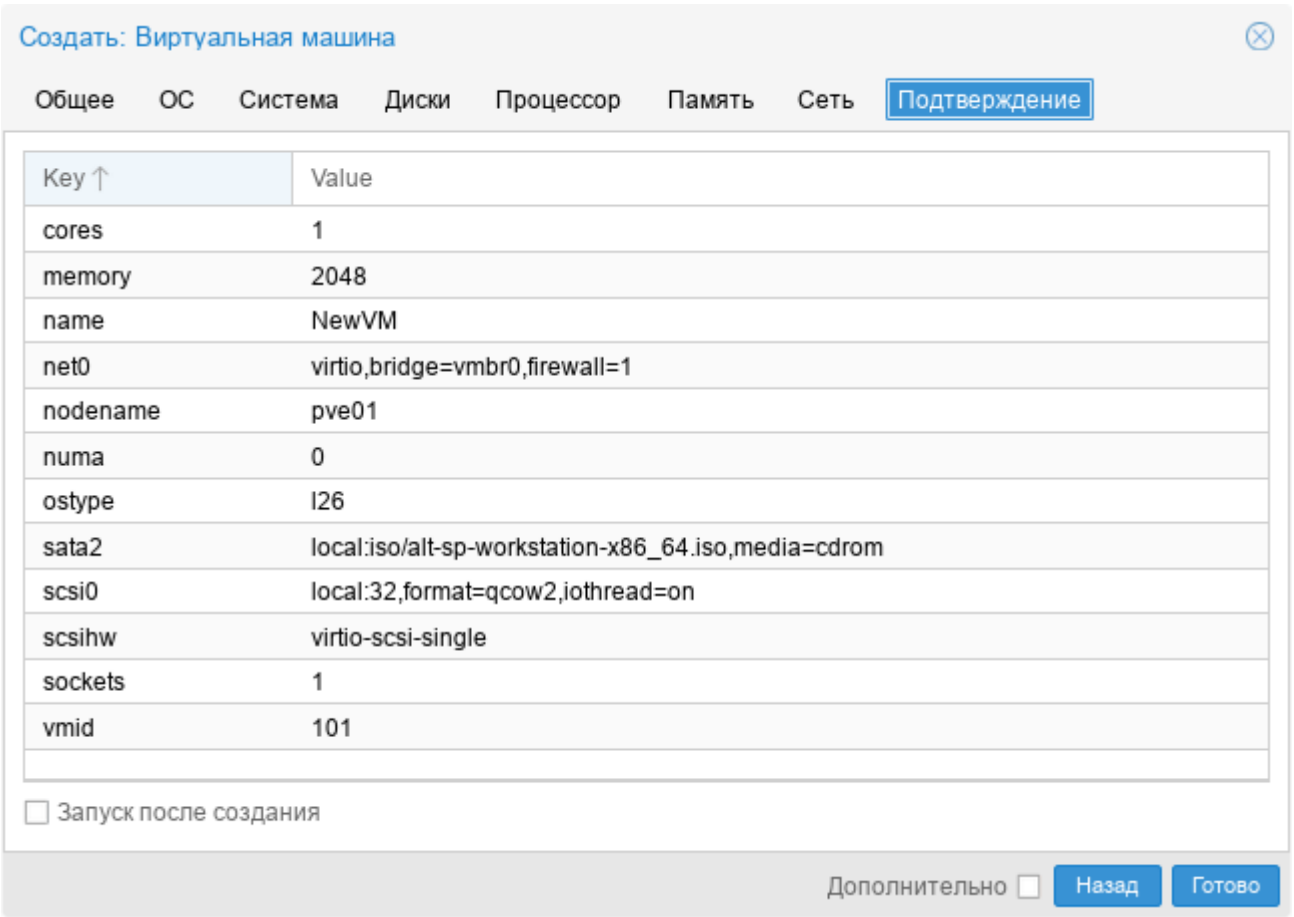

Рис. 259 – Вкладка «Подтверждение»

8.7.2. Запуск и остановка ВМ

8.7.2.1. Изменение состояния ВМ в веб-интерфейсе

Запустить ВМ можно, выбрав в контекстном меню ВМ пункт «Запуск» (рис. 260), либо нажав на кнопку «Запуск» (рис. 261).

Запущенная ВМ будет обозначена зеленой стрелкой на значке ВМ.

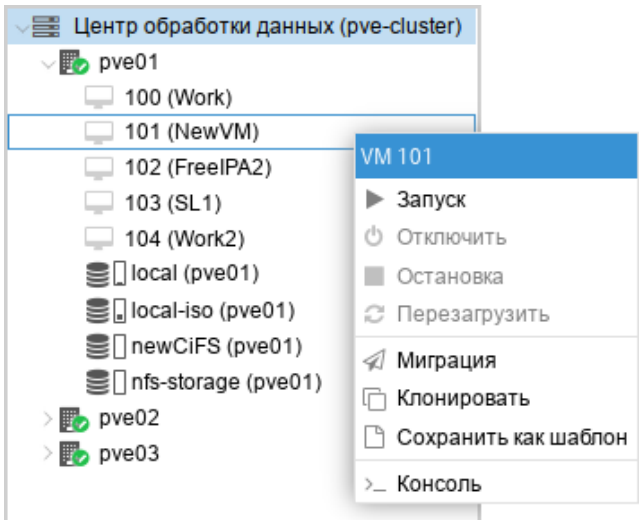

Рис. 260 – Контекстное меню ВМ

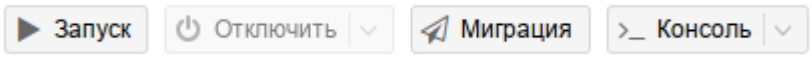

Рис. 261 – Кнопки управления состоянием ВМ

Запустить ВМ также можно, нажав кнопку «Start Now» в консоли гостевой машины (рис. 262).

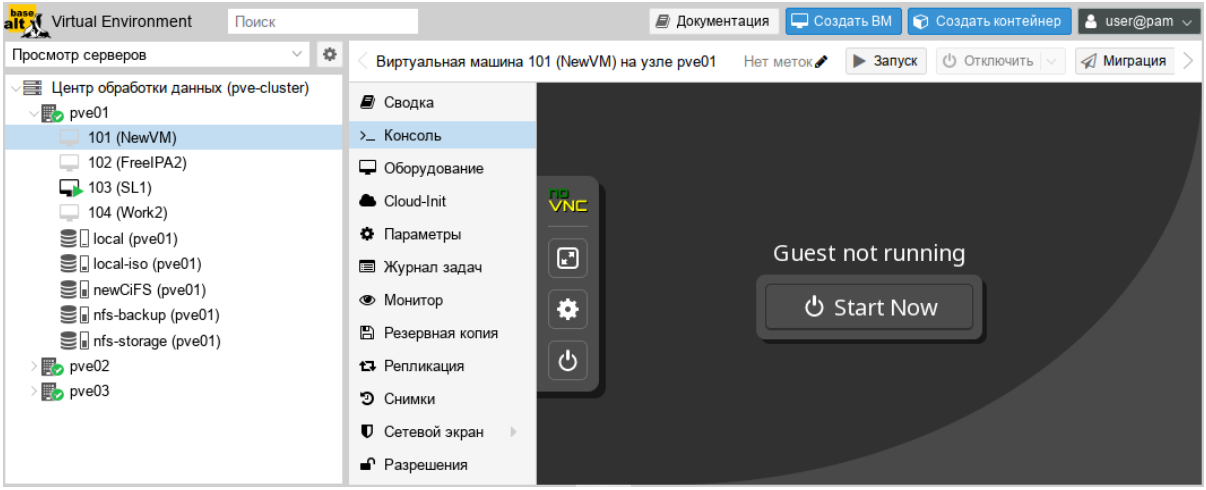

Рис. 262 – Кнопка «Start Now» в консоли ВМ

Для запущенной ВМ доступны следующие действия (рис. 263):

- «Приостановить» – перевод ВМ в спящий режим;
- «Гибернация» – перевод ВМ в ждущий режим;
- «Отключить» – выключение ВМ;
- «Остановка» – остановка ВМ, путем прерывания ее работы;
- «Перезагрузить» – перезагрузка ВМ.

8.7.2.2. 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
```

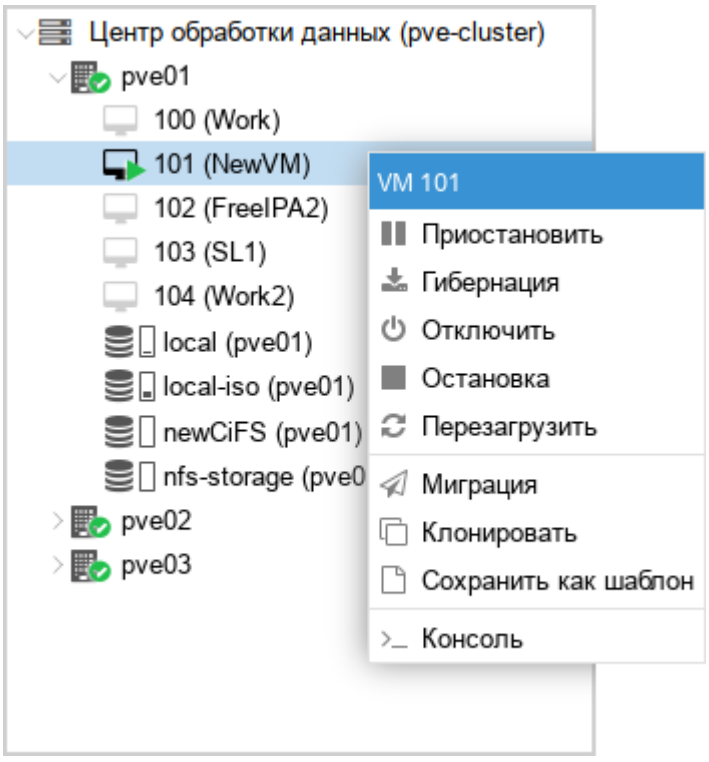

Рис. 263 – Контекстное меню запущенной ВМ

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. 264) (опция «Порядок запуска и отключения» на вкладке «Параметры» требуемой ВМ):

- «Порядок запуска и отключения» – 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» – определяет интервал (в секундах) между запуском этой ВМ и последующими запусками ВМ;
- «Задержка отключения» – 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.

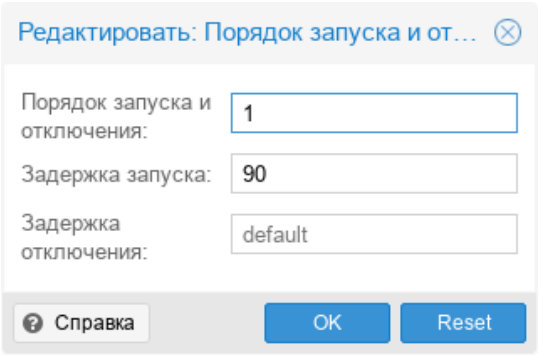

Рис. 264 – Настройка порядка запуска и отключения ВМ

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 отключения» 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.

8.7.3. 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 Гбайт, в хранилище local-lvm:

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

8.7.4. 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 разрешения 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 «Экран» ВМ должно быть установленно значение VirtIO, SPICE (qxl) (см. п. 8.7.5.2 «Настройки дисплея»).

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

Для доступа к ВМ следует выбрать ее в веб-интерфейсе, нажать на кнопку «Консоль» и в выпадающем меню выбрать нужную консоль (рис. 265).

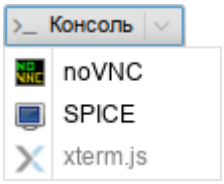

Рис. 265 – Кнопка «Консоль»

Консоль noVNC также можно запустить, выбрав вкладку «Консоль» для ВМ (рис. 266).

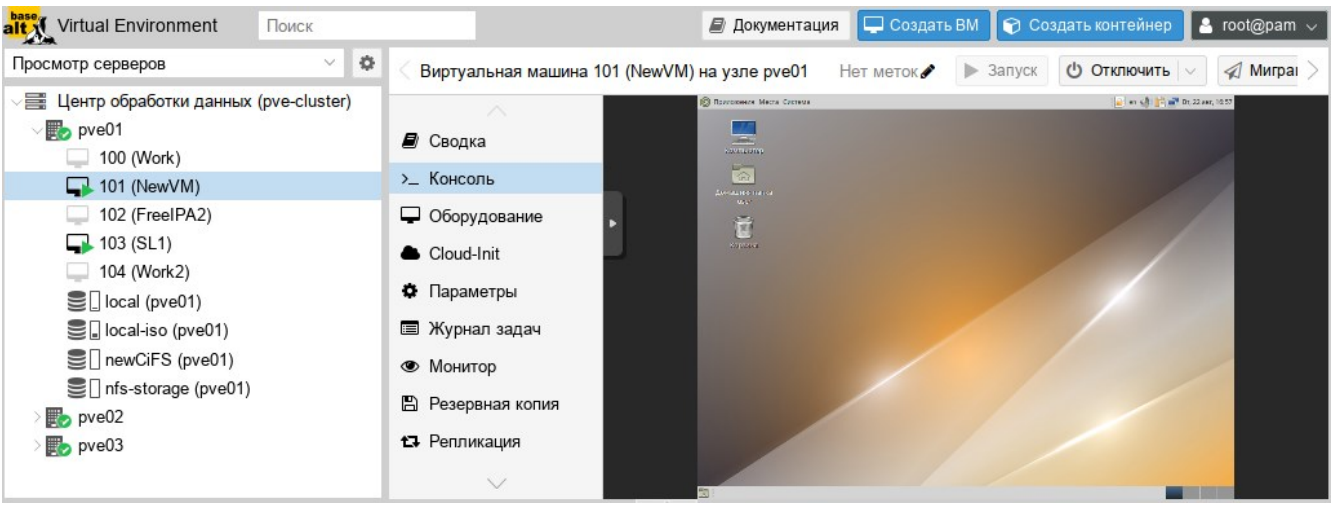

Рис. 266 – Консоль noVNC

Если нужен независимый от веб-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).

8.7.5. 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» (рис. 267). На этой вкладке следует выбрать ресурс и нажать на кнопку «Редактировать» для выполнения изменений.

Примечание. В случаях, когда изменение не может быть выполнено в горячем режиме, оно будет зарегистрировано как ожидающее изменение (рис. 268) (выделяется цветом). Такие изменения будут применены только после перезагрузки ВМ.

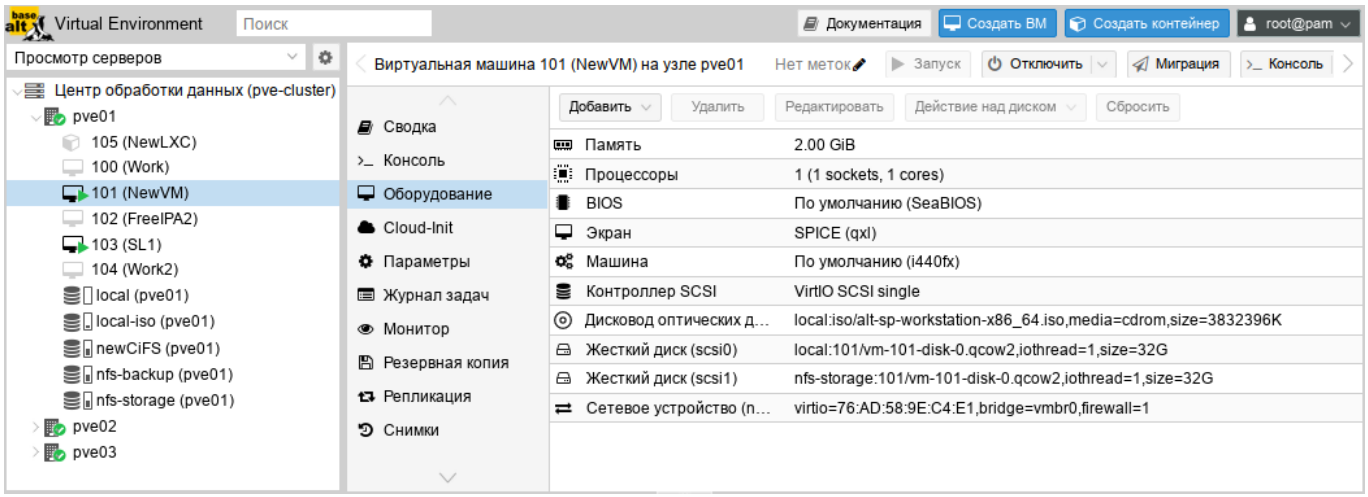

Рис. 267 – Оборудование ВМ

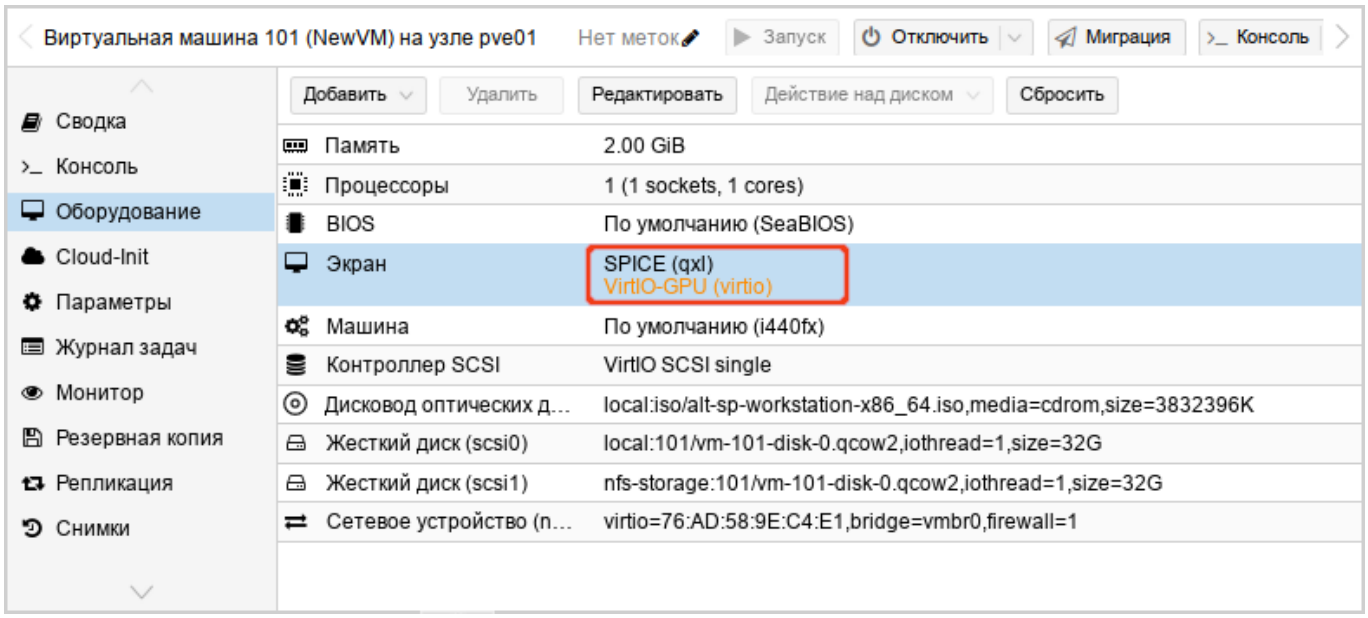

Рис. 268 – Изменения, которые будут применены после перезапуска ВМ

8.7.5.1. 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.

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 Гбайт:

```
# 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». После выбора образа диска станут доступными кнопки (рис. 269): «Добавить», «Отключить», «Редактировать», «Изменить размер», «Переназначить владельца», «Переместить хранилище».

8.7.5.1.1. Добавление виртуального диска в ВМ

Для добавления образа виртуального диска к ВМ необходимо:

1) перейти на вкладку «Оборудование» (рис. 269);
2) нажать на кнопку «Добавить» и выбрать в выпадающем списке пункт «Жесткий диск» (рис. 270);
3) указать параметры жесткого диска (рис. 271) и нажать на кнопку «Добавить».

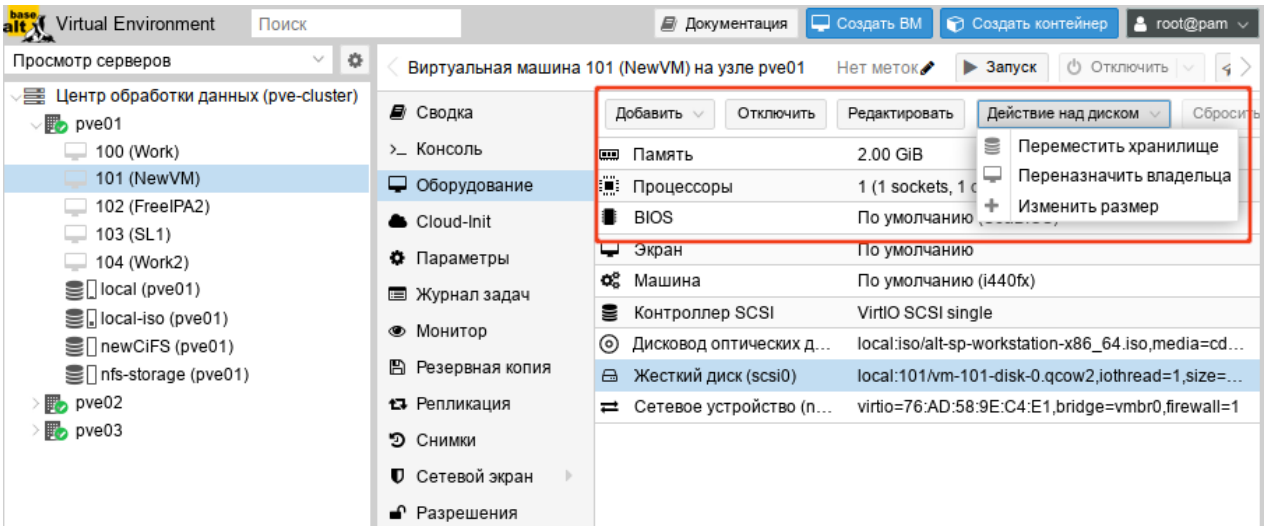

Рис. 269 – Вкладка «Оборудование». Управление образом виртуального диска

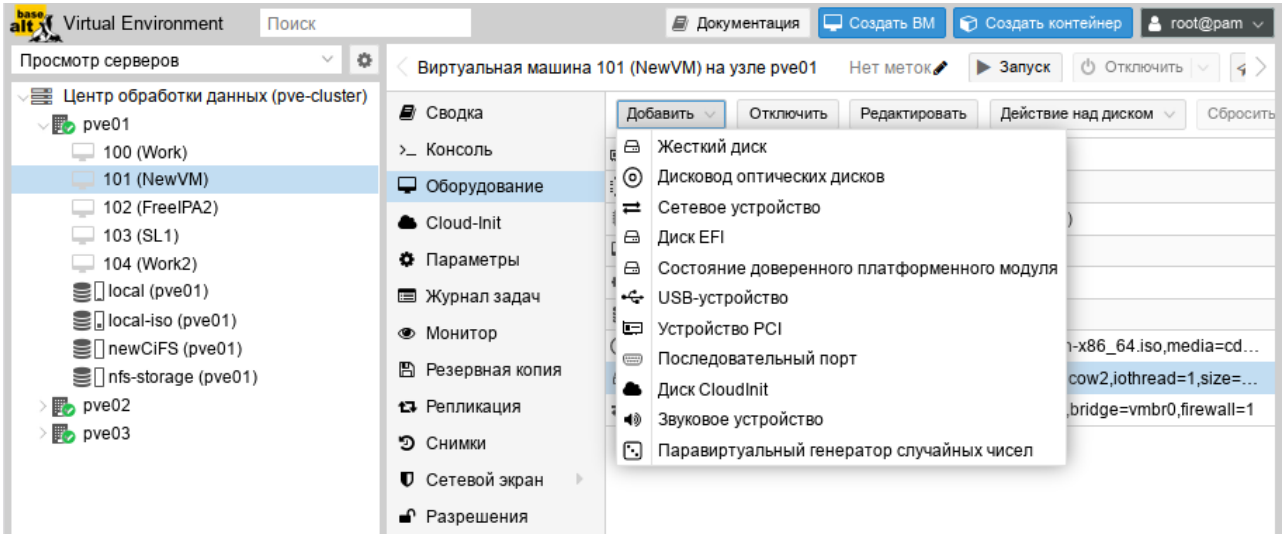

Рис. 270 – Кнопка «Добавить» → «Жесткий диск»

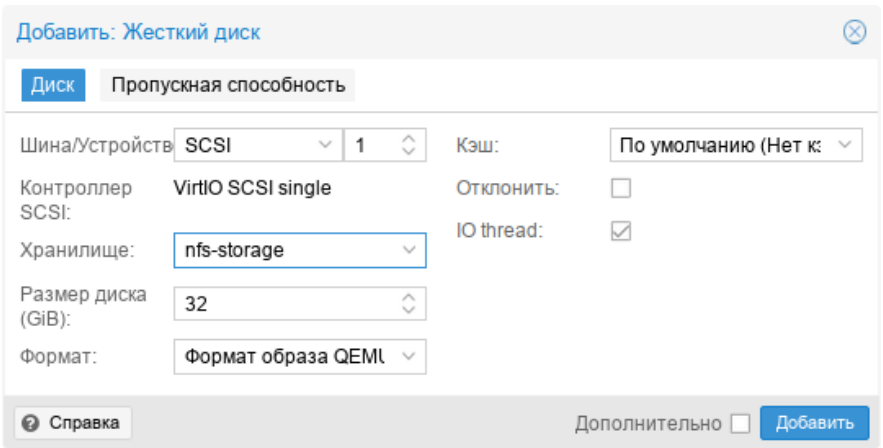

Рис. 271 – Опции добавления жесткого диска

8.7.5.1.2. Удаление образа виртуального диска

Для удаления образа виртуального диска необходимо:

1) перейти на вкладку «Оборудование» (рис. 269);
2) выбрать образ диска ВМ;
3) нажать на кнопку «Отключить»;
4) в окне подтверждения нажать на кнопку «Да» 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» (рис. 272).

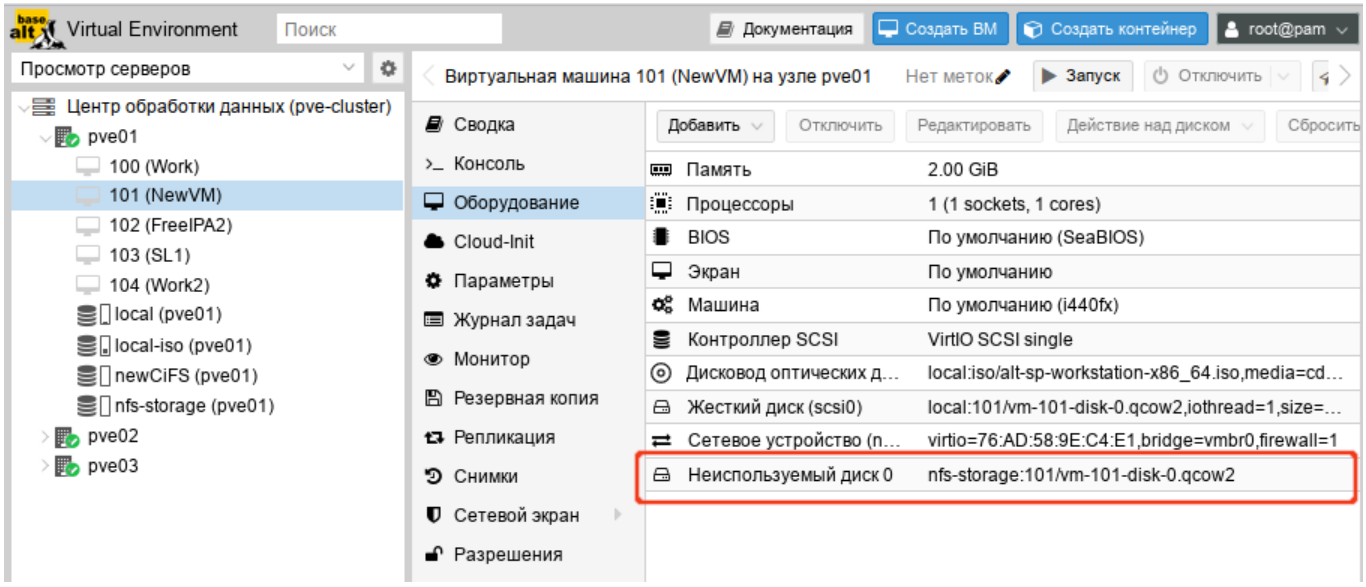

Рис. 272 – «Неиспользуемый диск»

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

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

1) выбрать неиспользуемый диск;
2) нажать на кнопку «Редактировать»;
3) в открывшемся диалоговом окне (рис. 273) изменить, если это необходимо, параметры «Шина/Устройство»;
4) нажать на кнопку «Добавить» для повторного подключения образа диска.

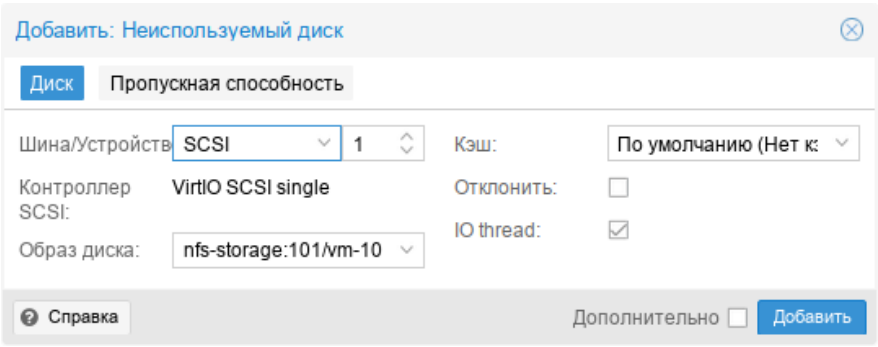

Рис. 273 – Подключение неиспользуемого диска

8.7.5.1.3. 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» (рис. 269);
2) выбрать образ виртуального диска;
3) нажать на кнопку «Действие над диском» → «Изменить размер»;
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 Гбайт, для изменения размера диска до 30 Гбайт следует ввести число 10 (рис. 274);
5) нажать на кнопку «Изменить размер диска» для завершения изменения размера.

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 Гбайт:

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

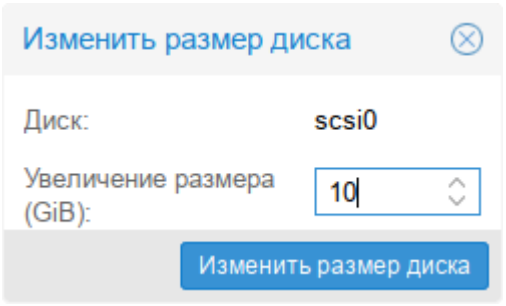

Рис. 274 – Изменение размера диска

8.7.5.1.4. 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» (рис. 269);
2) выбрать образ диска, который необходимо переместить;
3) нажать на кнопку «Действие над диском» → «Переместить хранилище»;
4) в открывшемся диалоговом окне (рис. 275) в выпадающем меню «Целевое хранилище» выбрать хранилище-получатель, место, куда будет перемещен образ виртуального диска;
5) в выпадающем меню «Формат» выбрать формат образа диска. Этот параметр полезен для преобразования образа диска из одного формата в другой;
6) отметить, если это необходимо, пункт «Удалить источник» для удаления образа диска из исходного хранилища после его перемещения в новое хранилище;
7) нажать на кнопку «Переместить диск».

Команда перемещения образа диска в другое хранилище:

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

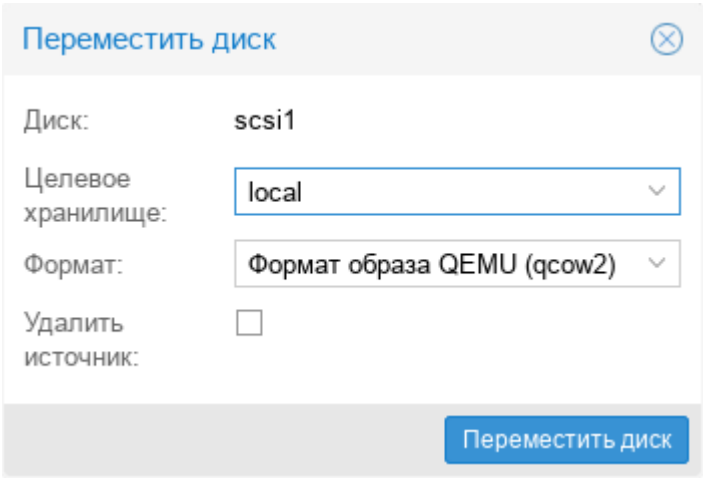

Рис. 275 – Диалоговое окно перемещения диска

8.7.5.1.5. Переназначение диска другой ВМ

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

Для переназначения образа диска другой ВМ необходимо:

1) перейти на вкладку «Оборудование» (рис. 269);
2) выбрать образ диска, который необходимо переназначить;
3) нажать на кнопку «Действие над диском» → «Переназначить владельца»;
4) в открывшемся диалоговом окне (рис. 276) в выпадающем «Целевой гость» выбрать целевую ВМ (место, куда будет перемещен образ виртуального диска);
5) выбрать нужные параметры в выпадающем меню «Шина/Устройство»;
6) нажать на кнопку «Переназначить диск».

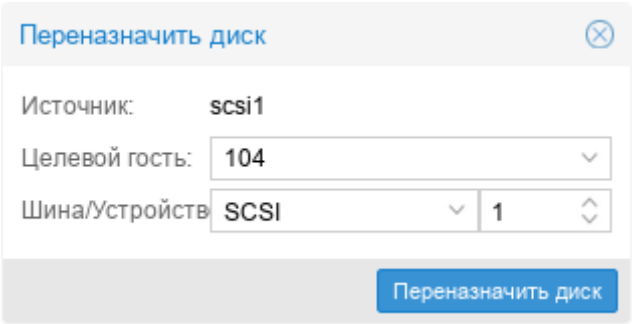

Рис. 276 – Диалоговое окно переназначения диска

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
```

8.7.5.2. Настройки дисплея

QEMU может виртуализировать разные типы оборудования VGA (рис. 277), 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.

Примечания:

1. Для типов дисплеев «VirtIO» и «VirGL» по умолчанию включена поддержка SPICE.
2. 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)»). Увеличение объема памяти может обеспечить более высокое разрешение внутри ВМ, особенно при использовании SPICE/QXL.

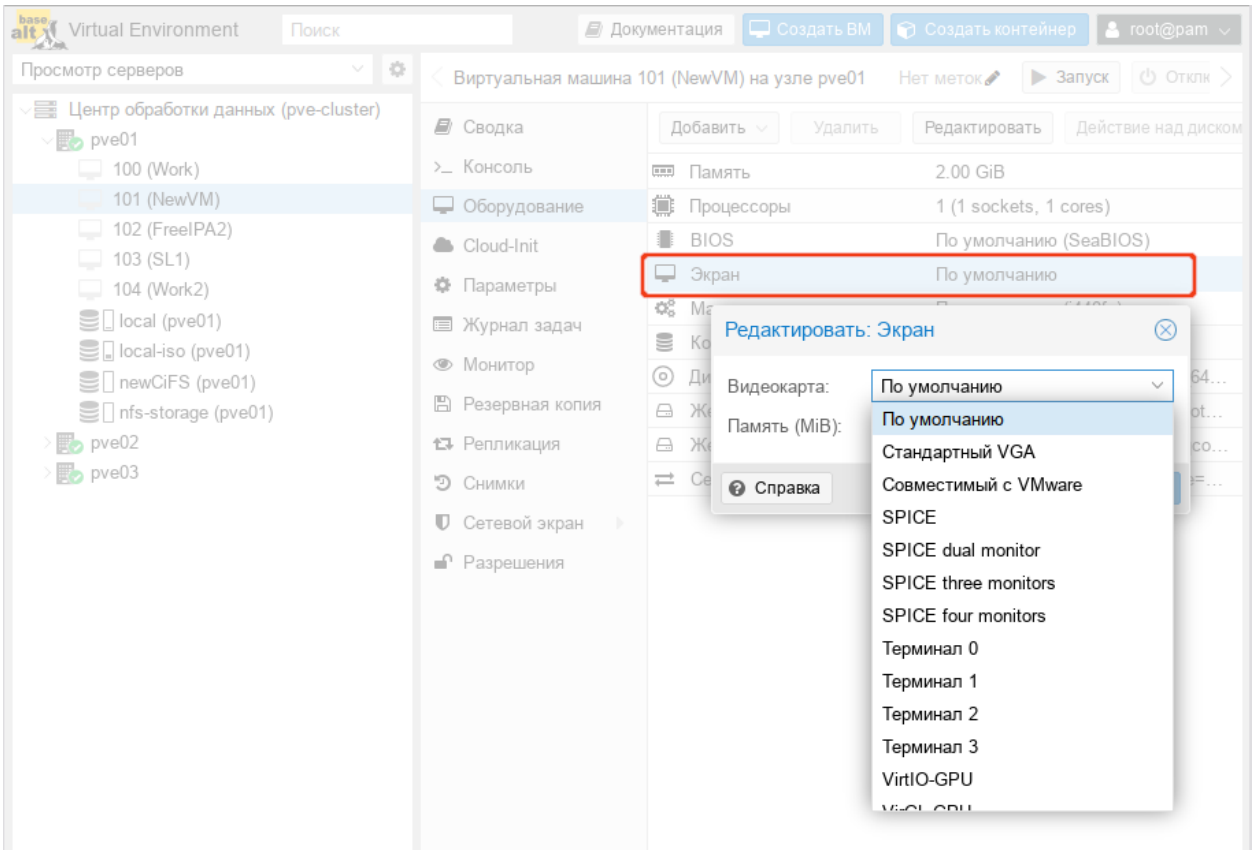

Рис. 277 – PVE. Настройки дисплея

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, объем 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.

8.7.5.3. 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. 278) (пункт «Улучшения SPICE» в разделе «Параметры» ВМ);
- в командной строке:

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

Примечание. Чтобы использовать дополнительные функции SPICE, для параметра «Экран» ВМ должно быть установлено значение SPICE (qxl).

8.7.5.3.1. Общий доступ к папкам (Folder Sharing)

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

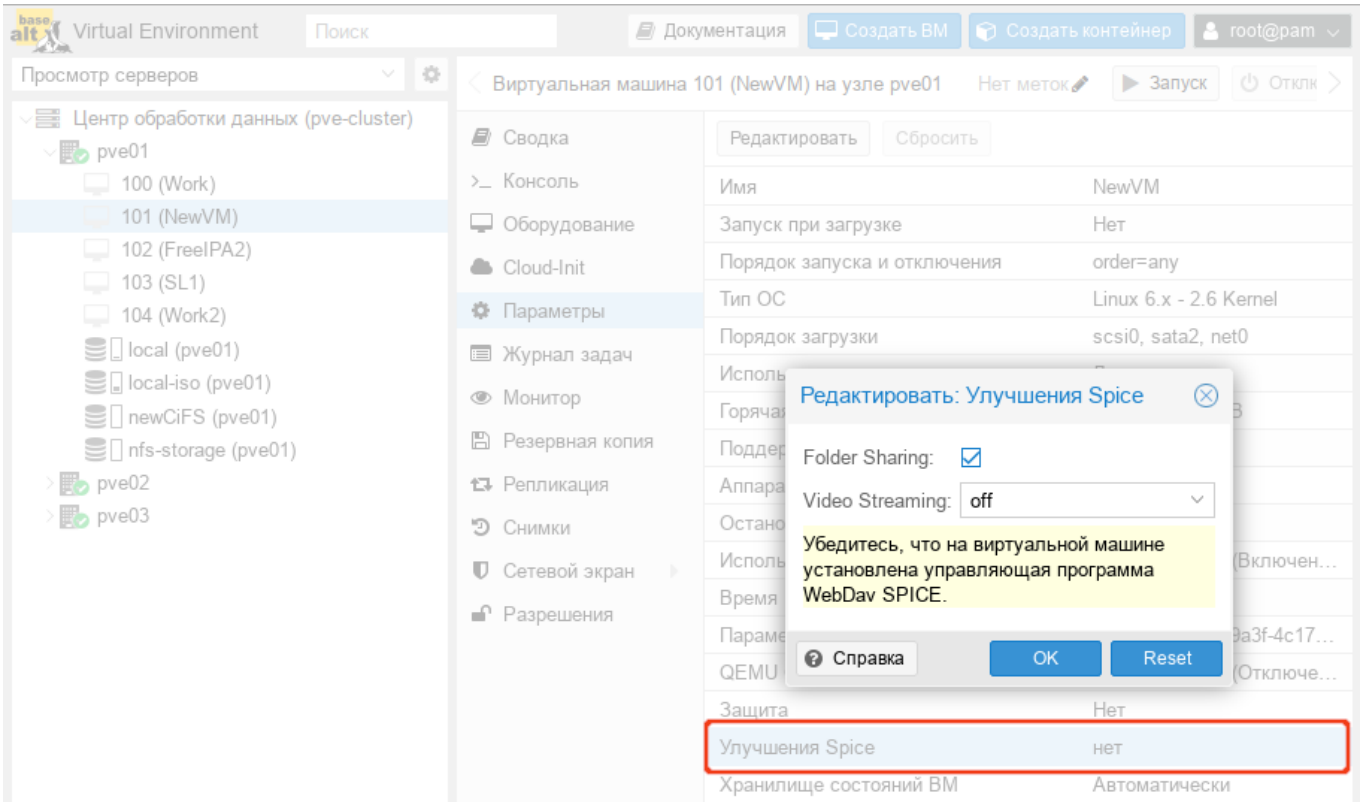

Рис. 278 – PVE. Дополнительные функции SPICE

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. 279).

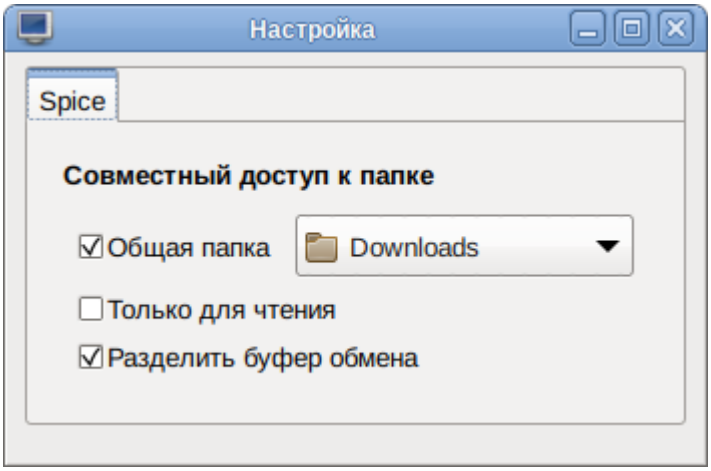

Рис. 279 – Совместный доступ к папке

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 веб-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/`).

8.7.5.3.2. Потоковое видео (Video Streaming)

Если потоковое видео включено, 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.

8.7.5.4. Проброс USB

Для проброса USB-устройства в ВМ необходимо:

1) перейти на вкладку «Оборудование» (рис. 269);

2) нажать на кнопку «Добавить» и выбрать в выпадающем списке пункт «USB-устройство» (рис. 280);

3) откроется окно добавления устройства, в котором можно выбрать режим проброса:

- «Порт Spice» – сквозная передача SPICE USB (рис. 281) (позволяет пробросить USB-устройство с клиента SPICE);
- «Использовать устройство USB по номеру» – проброс в ВМ конкретного USB-устройства (рис. 282). USB-устройство можно выбрать в выпадающем списке «Выберите устройство» или указать вручную, указав `<ID-производителя>:<ID-устройства>` (можно получить из вывода команды `lsusb`);
- «Использовать порт USB» – проброс конкретного порта (рис. 283) (в ВМ будет проброшено любое устройство, вставленное в этот порт). USB-порт можно выбрать в выпадающем списке «Выберите порт» или ввести вручную, указав `<Номер_шины>:<Путь_к_порту>` (можно получить из вывода команды `lsusb`);

4) нажать на кнопку «Добавить»;

5) остановить и запустить ВМ (перезагрузки недостаточно).

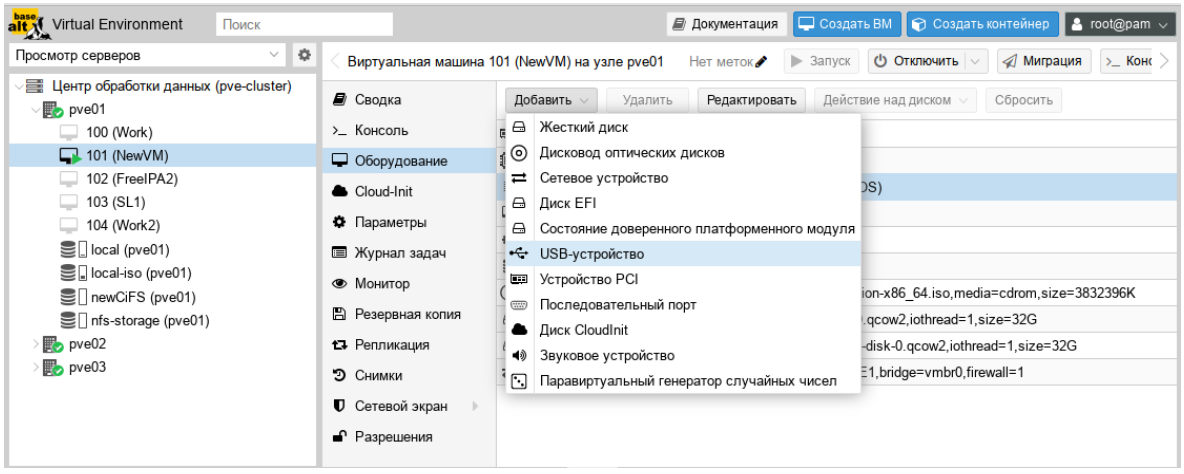

Рис. 280 – Кнопка «Добавить» → «Устройство USB»

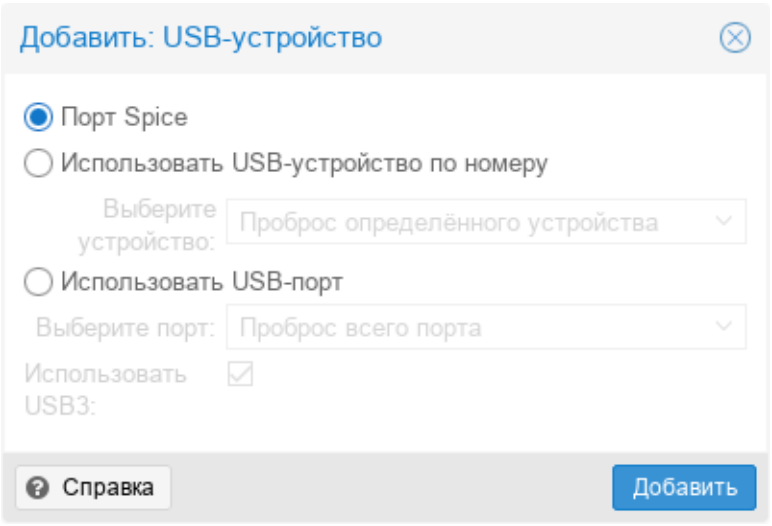

Рис. 281 – Порт Spice

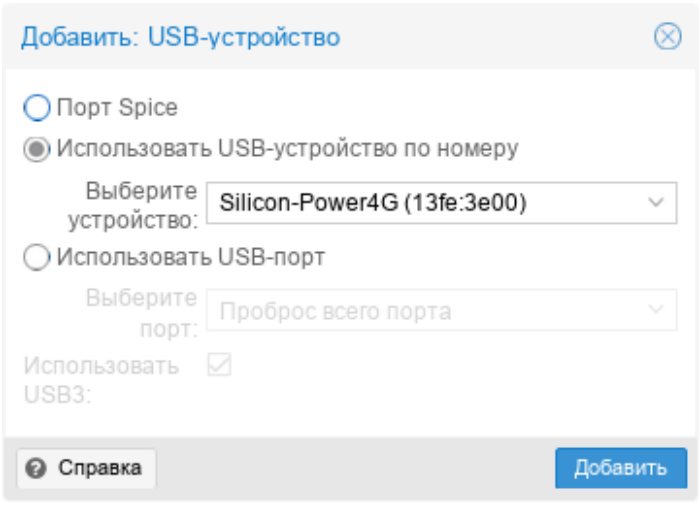

Рис. 282 – Использовать устройство USB по номеру

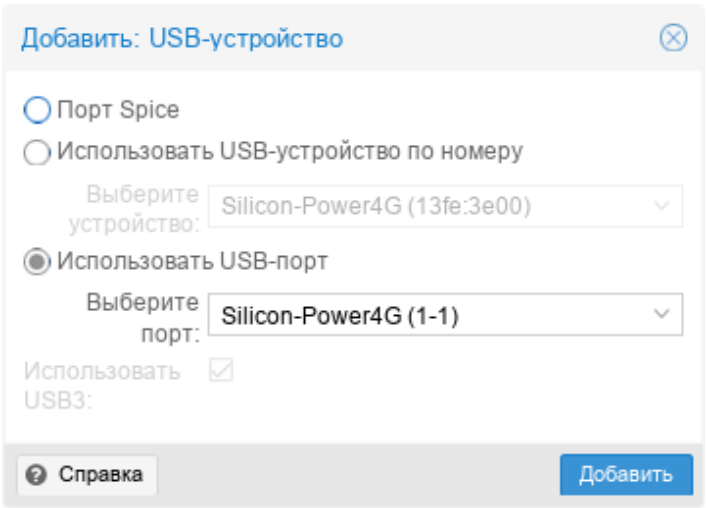

Рис. 283 – Использовать порт USB

Примечание. Список подключенных к ВМ и хосту USB-устройств можно получить, введя на вкладке «Монитор» соответственно команды `info usb` или `info usbhost` (рис. 284).

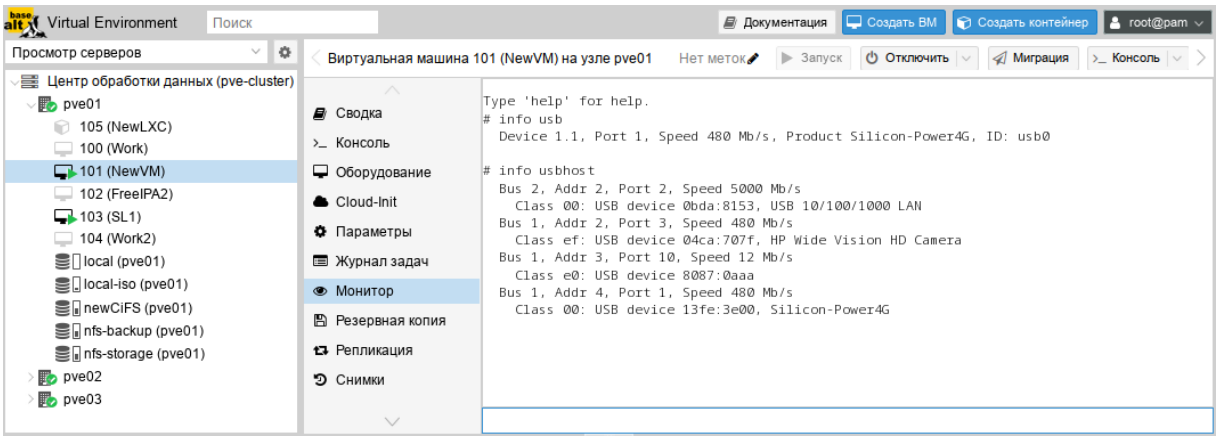

Рис. 284 – Список подключенных к ВМ и хосту USB-устройств

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.

8.7.5.5. 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:

- поменять тип прошивки на UEFI (рис. 285);
- добавить в конфигурацию ВМ диск EFI (рис. 286).

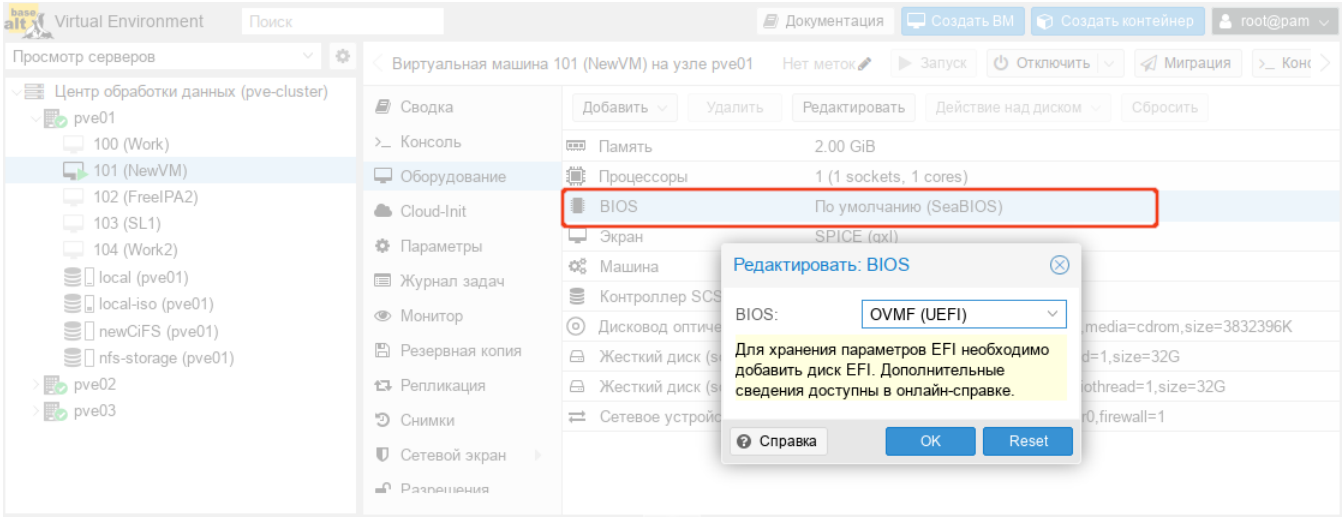

Рис. 285 – PVE. Настройка BIOS

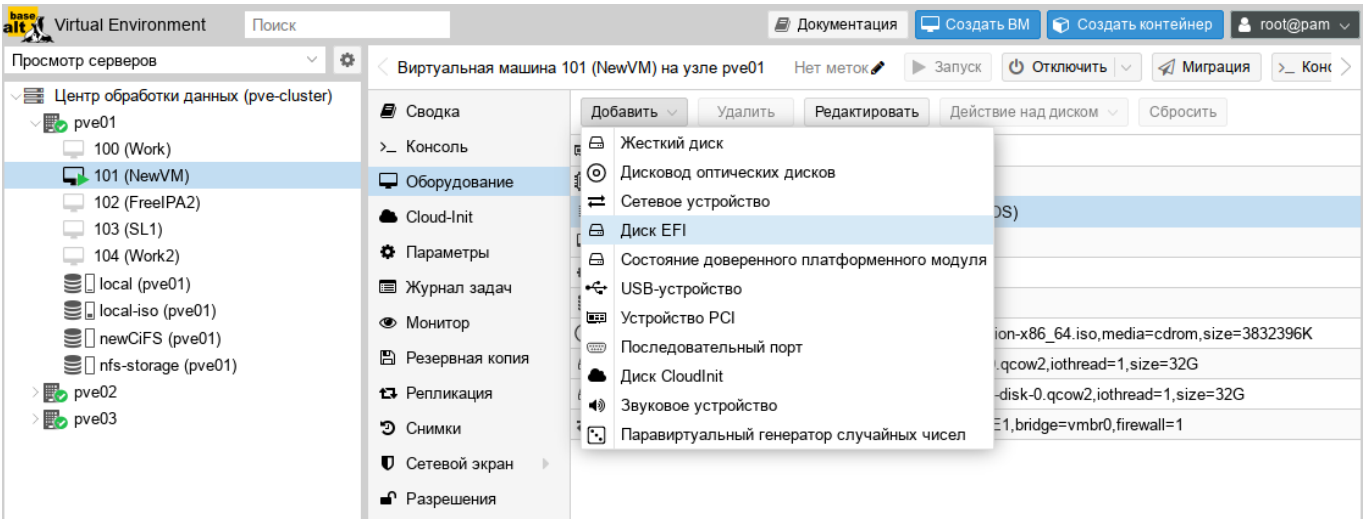

Рис. 286 – PVE. Добавление диска EFI

Команда создания диска EFI:

```
# qm set <vm_id> -efidisk0 <storage>:1,format=<format>,efitype=4m,pre-enrolled-keys=1
```

где:

- `<storage>` – хранилище, в котором будет размещен 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 efidisk поставляться с предварительно загруженными ключами безопасной загрузки для конкретного дистрибутива и Microsoft Standard Secure Boot. Включает безопасную загрузку по умолчанию.

8.7.5.6. Доверенный платформенный модуль (TPM)

TPM (англ. Trusted Platform Module) – спецификация, описывающая криптопроцессор, в котором хранятся криптографические ключи для защиты информации, а также обобщенное 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» → «Состояние доверенного платформенного модуля») показано на рис. 287.

Команда добавления TPM:

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

где:

- `<storage>` – хранилище, в которое будет помещен модуль;
- `<version>` – версия (1.2 или 2.0).

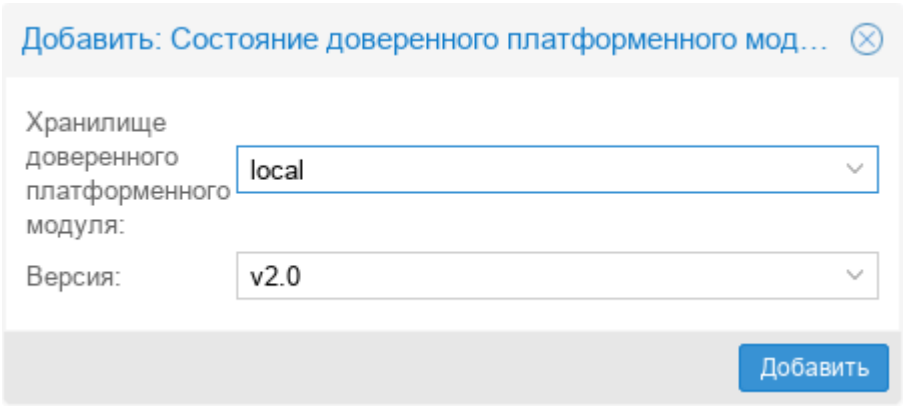

Рис. 287 – PVE. Добавление TPM в веб-интерфейсе

8.7.5.7. 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 проброс PCI(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. 288). 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 отметки «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
```

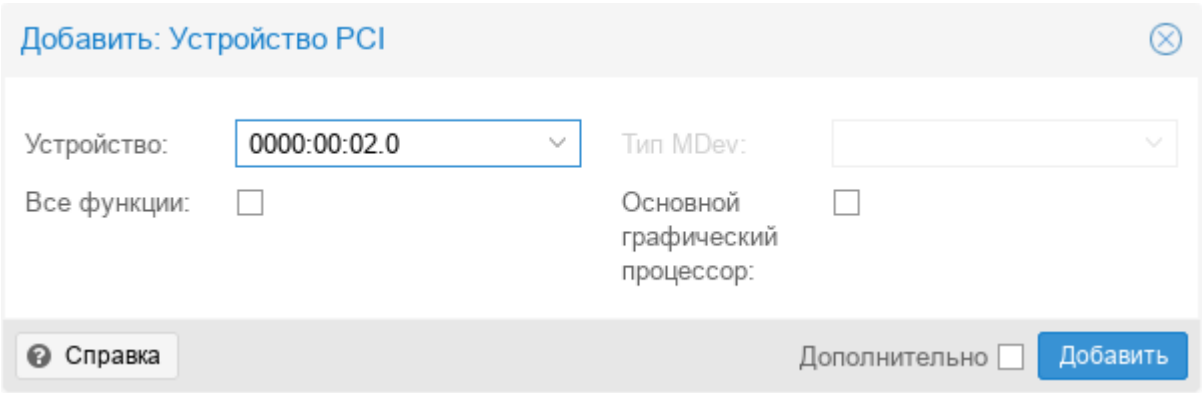

Рис. 288 – PVE. Добавление устройства PCI

8.7.6. 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;sata2;net0
cores: 1
memory: 2048
meta: creation-qemu=7.2.10,ctime=1692701248
name: NewVM
net0: virtio=76:AD:58:9E:C4:E1,bridge=vmbr0,firewall=1
numa: 0
ostype: l26
sata2: local:iso/alt-sp-workstation-
x86_64.iso,media=cdrom,size=3832396K
scsi0: local:101/vm-101-disk-0.qcow2,iothread=1,size=32G
scsi1: nfs-storage:101/vm-101-disk-0.qcow2,iothread=1,size=32G
scsihw: virtio-scsi-single
smbios1: uuid=547b268e-9a3f-4c17-8dff-b0dc20c39e58
sockets: 1
spice_enhancements: foldersharing=1
usb0: host=13fe:3e00
vga: qxl
vmgenid: f631f900-b5b3-4802-a300-7bfad377cd3a
```

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=7.2.10,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).

8.8. Создание и настройка контейнера LXC

8.8.1. Создание контейнера в графическом интерфейсе

Перед созданием контейнера можно загрузить шаблоны LXC в хранилище (см. п. 8.6).

Для создания контейнера необходимо нажать на кнопку «Создать контейнер», расположенную в правом верхнем углу веб-интерфейса PVE (рис. 289). Будет запущен диалог «Создать: Контейнер LXC» (рис. 290), который предоставляет графический интерфейс для настройки контейнера.

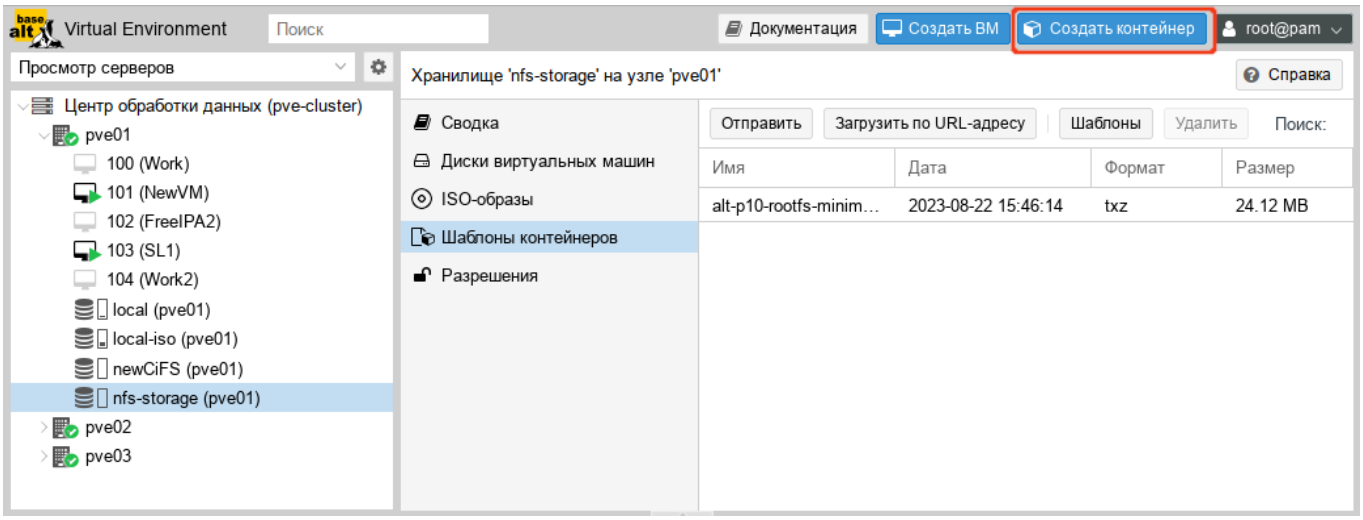

Рис. 289 – Кнопка «Создать контейнер»

На первой вкладке «Общее» необходимо указать (рис. 290):

- «Узел» – узел назначения для данного контейнера;
- «CT ID» – идентификатор контейнера в численном выражении;
- «Имя хоста» – 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);
- «Вложенность» – определяет возможность запуска контейнер в контейнере;
- «Пул ресурсов» – логическая группа контейнеров. Чтобы иметь возможность выбора, пул должен быть предварительно создан;
- «Пароль» – пароль для данного контейнера;
- «Открытый SSH ключ» – ssh ключ.

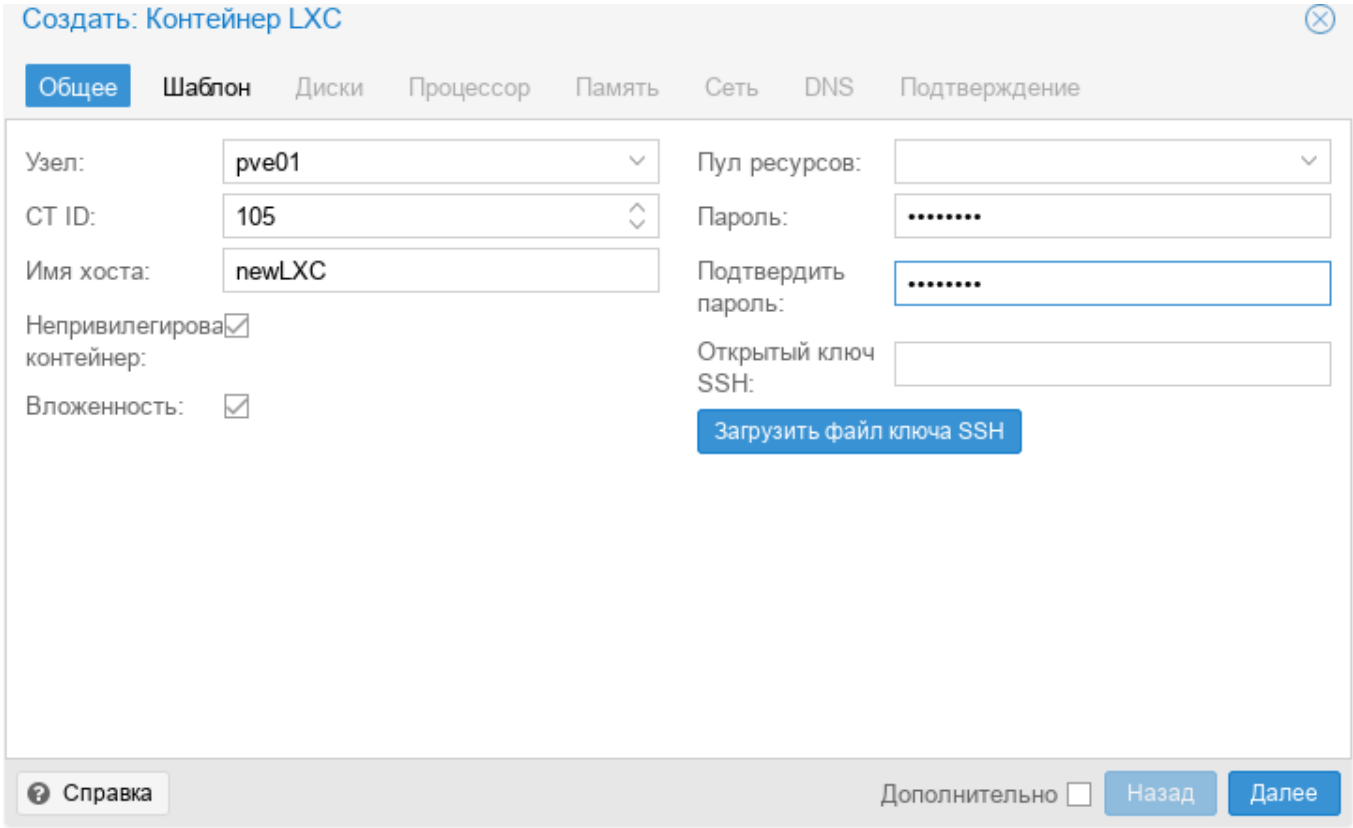

Рис. 290 – Вкладка «Общее» диалога создания контейнера

На вкладке «Шаблон» следует выбрать (рис. 291):

- «Хранилище» – хранилище, в котором хранятся шаблоны LXC;
- «Шаблон» – шаблон контейнера.

На вкладке «Диски» определяется хранилище, где будут храниться диски контейнера (рис. 292). Здесь также можно определить размер виртуальных дисков (не следует выбирать размер диска менее 4 Гбайт).

На вкладке «Процессор» определяется количество ядер процессора, которые будут выделены контейнеру (рис. 293).

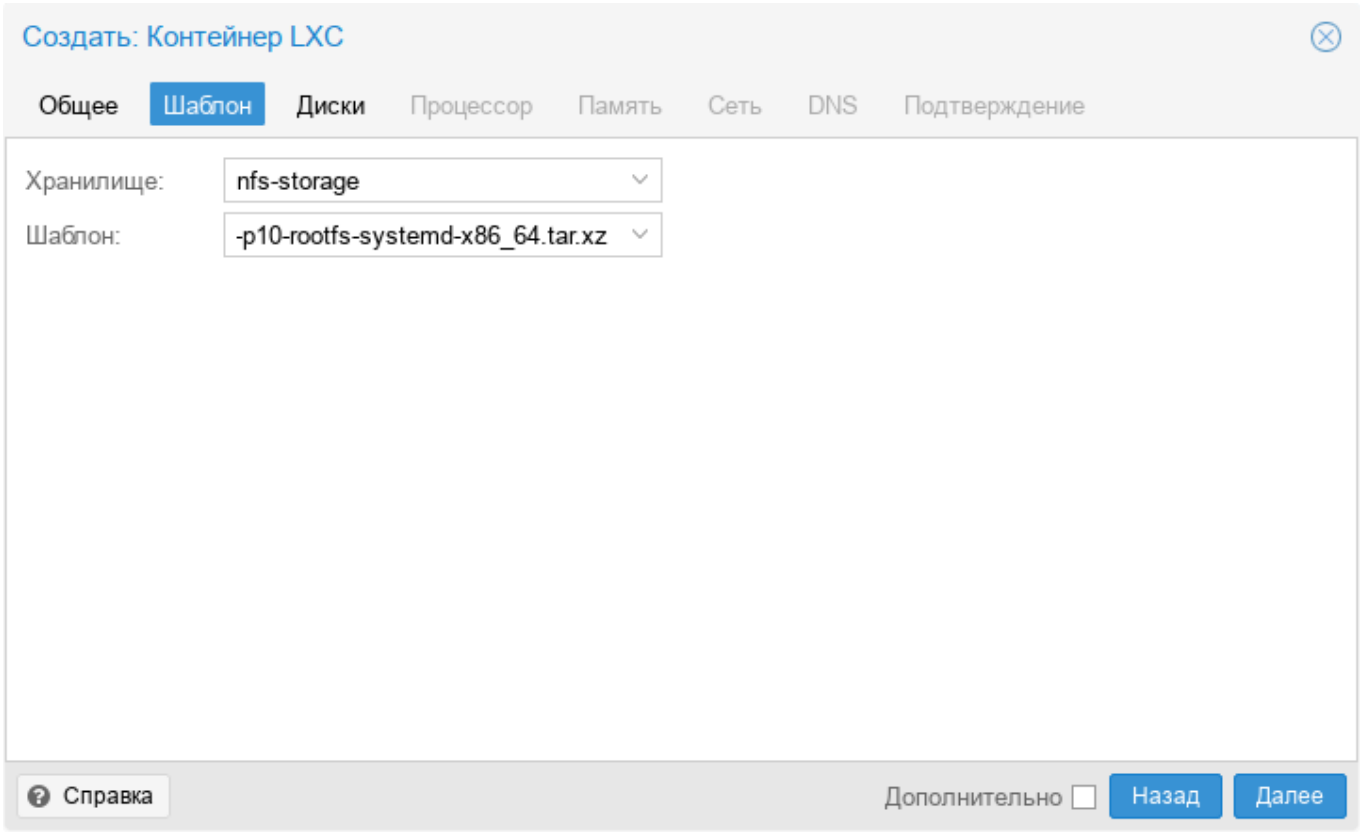

Рис. 291 – Вкладка «Шаблон» диалога создания контейнера

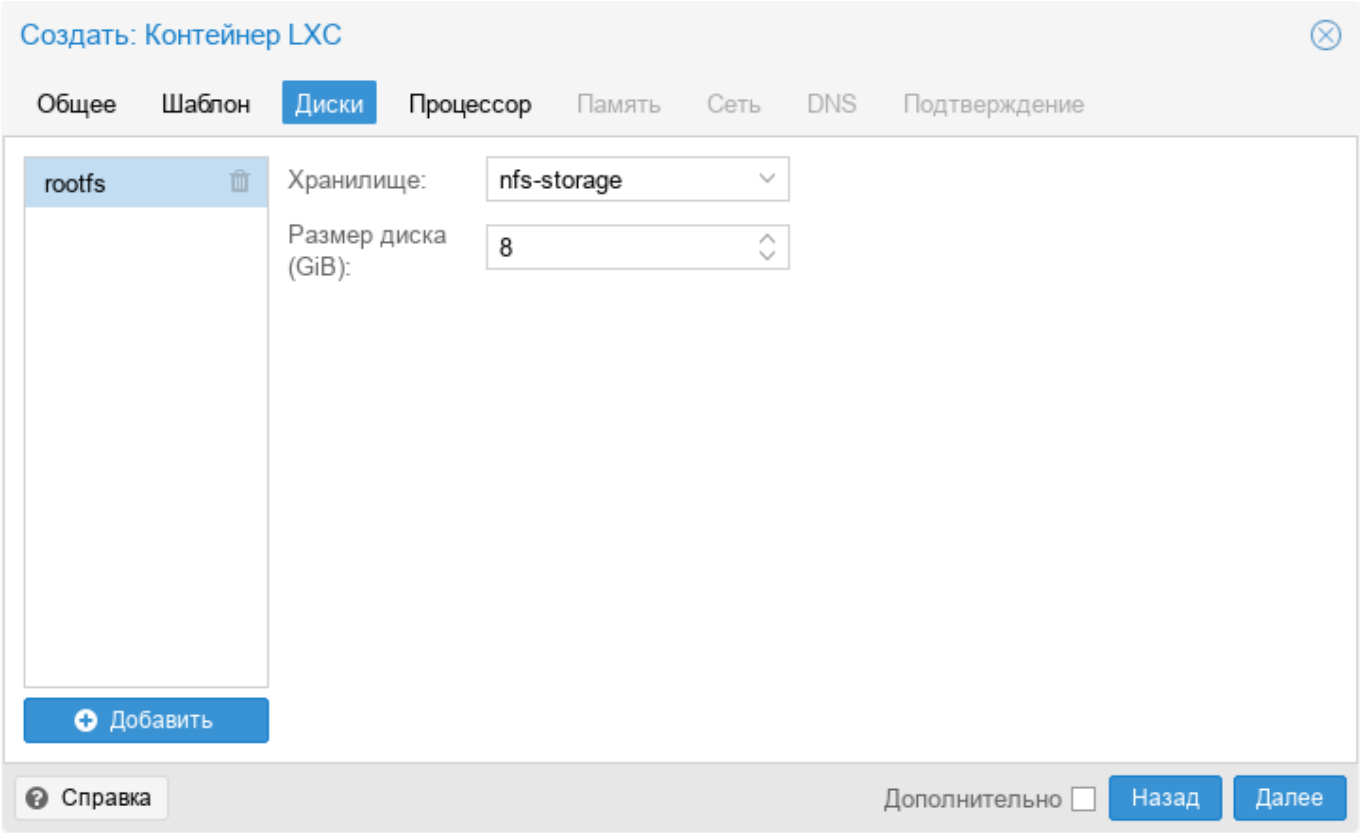

Рис. 292 – Вкладка «Диски» диалога создания контейнера

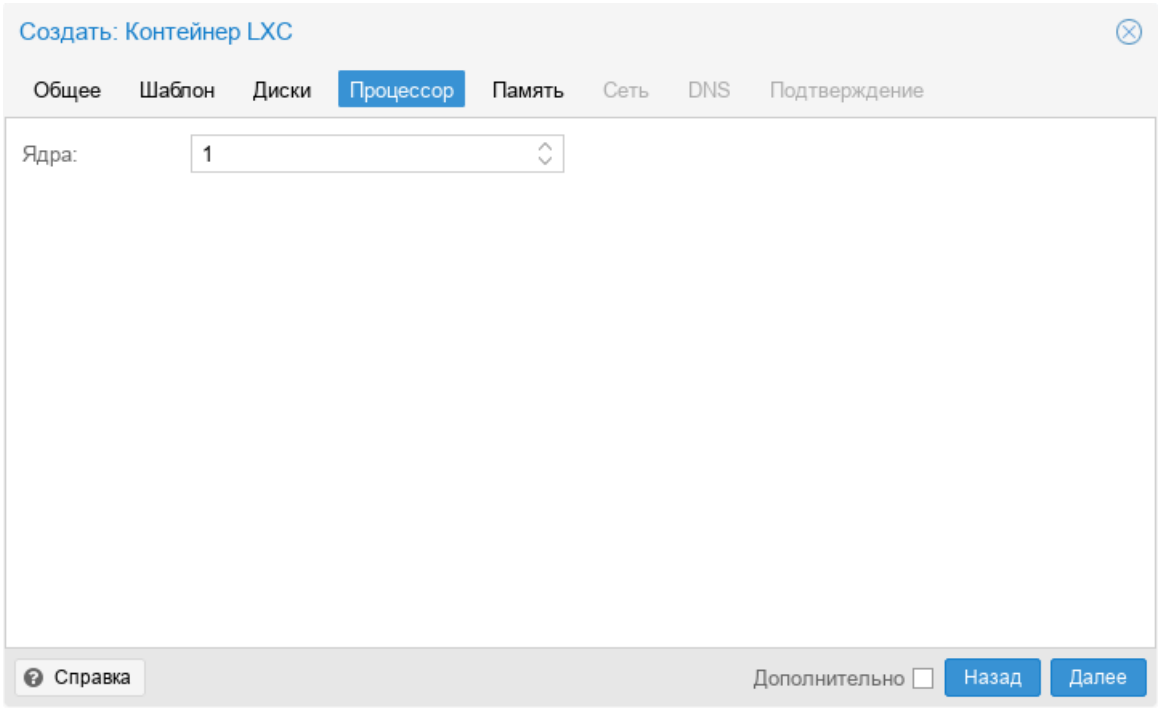

Рис. 293 – Вкладка «Процессор» диалога создания контейнера

На вкладке «Память» настраиваются (рис. 294):

- «Память» (MiB) – выделяемая память в Мебибайтах;
- «Подкачка» (MiB) – выделяемое пространство подкачки в Мебибайтах.

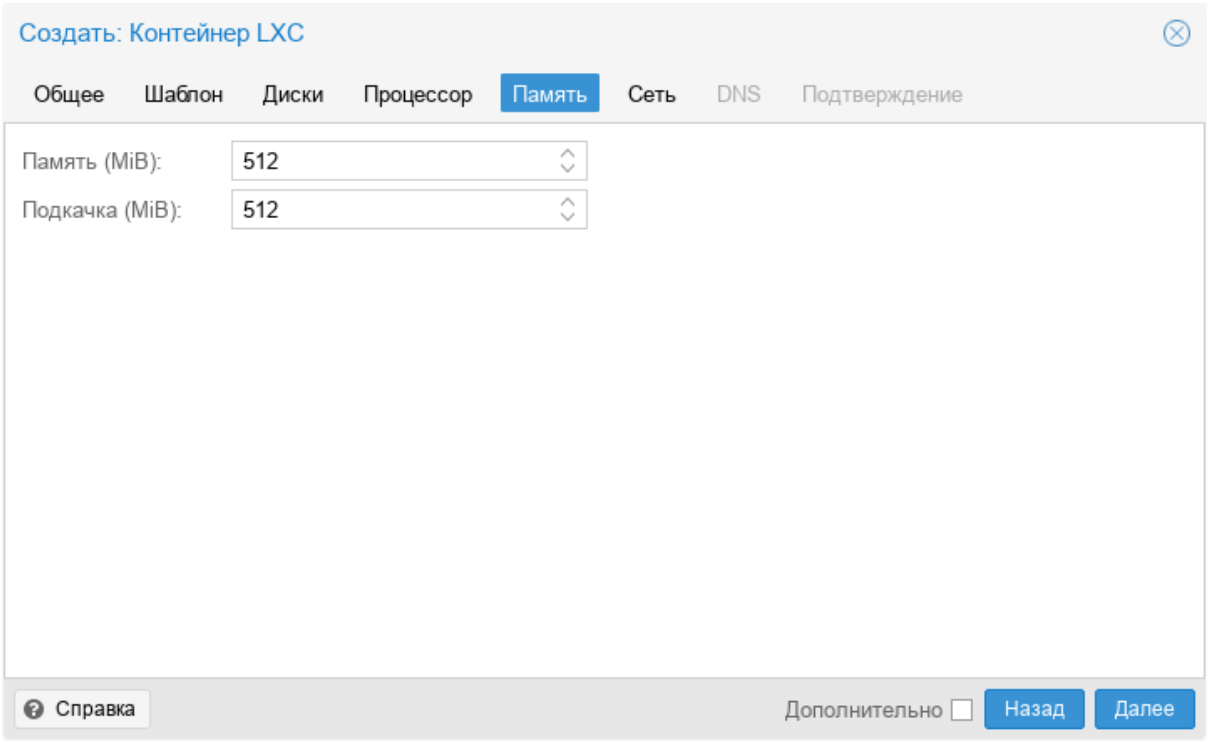

Рис. 294 – Вкладка «Память» диалога создания контейнера

Вкладка «Сеть» включает следующие настройки (рис. 295):

- «Имя» – определяет, как будет именоваться виртуальный сетевой интерфейс внутри контейнера (по умолчанию eth0);
- «MAC-адрес» – можно задать определенный MAC-адрес, необходимый для приложения в данном контейнере (по умолчанию, все MAC-адреса для виртуальных сетевых интерфейсов назначаются автоматически);
- «Сетевой мост» – выбор виртуального моста, к которому будет подключаться данный интерфейс (по умолчанию vmbr0);
- «Тег виртуальной ЛС» – применяется для установки идентификатора VLAN для данного виртуального интерфейса;
- «Сетевой экран» – 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).

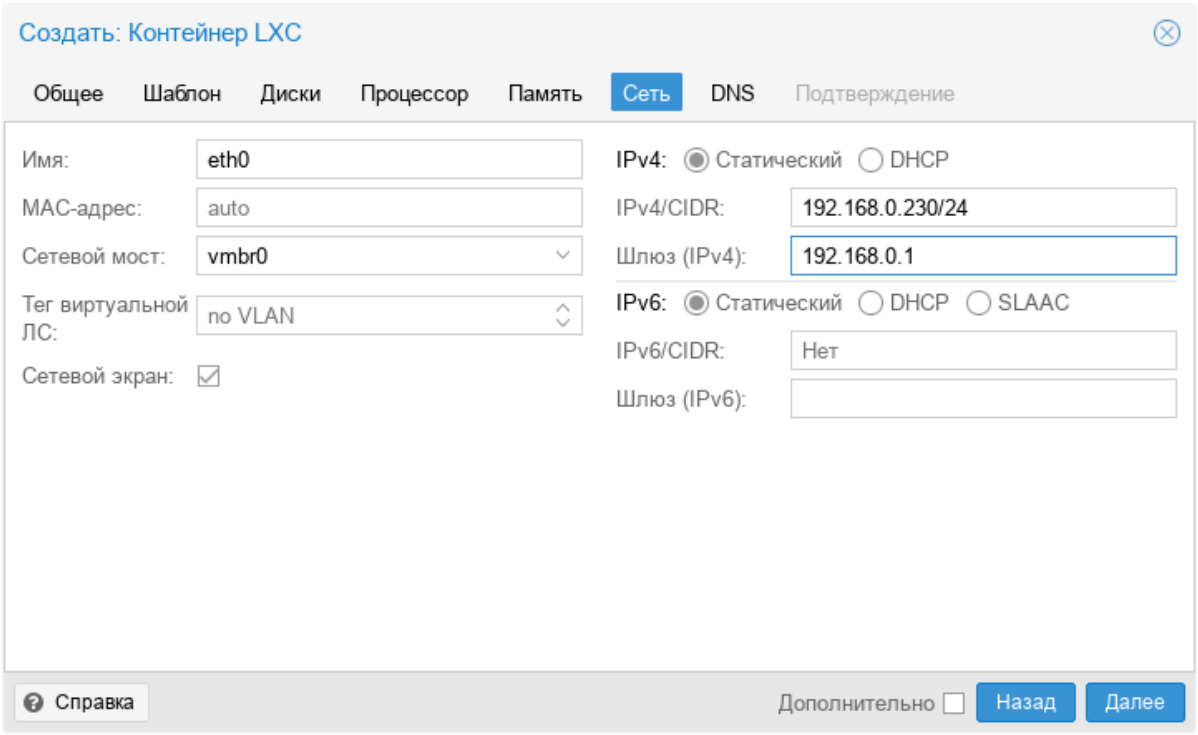

Рис. 295 – Вкладка «Сеть» диалога создания контейнера

Вкладка «DNS» содержит настройки (рис. 296):

- «Домен DNS» – имя домена (по умолчанию используются параметры хост системы);
- «Серверы DNS» – IP-адреса серверов DNS (по умолчанию используются параметры хост системы).

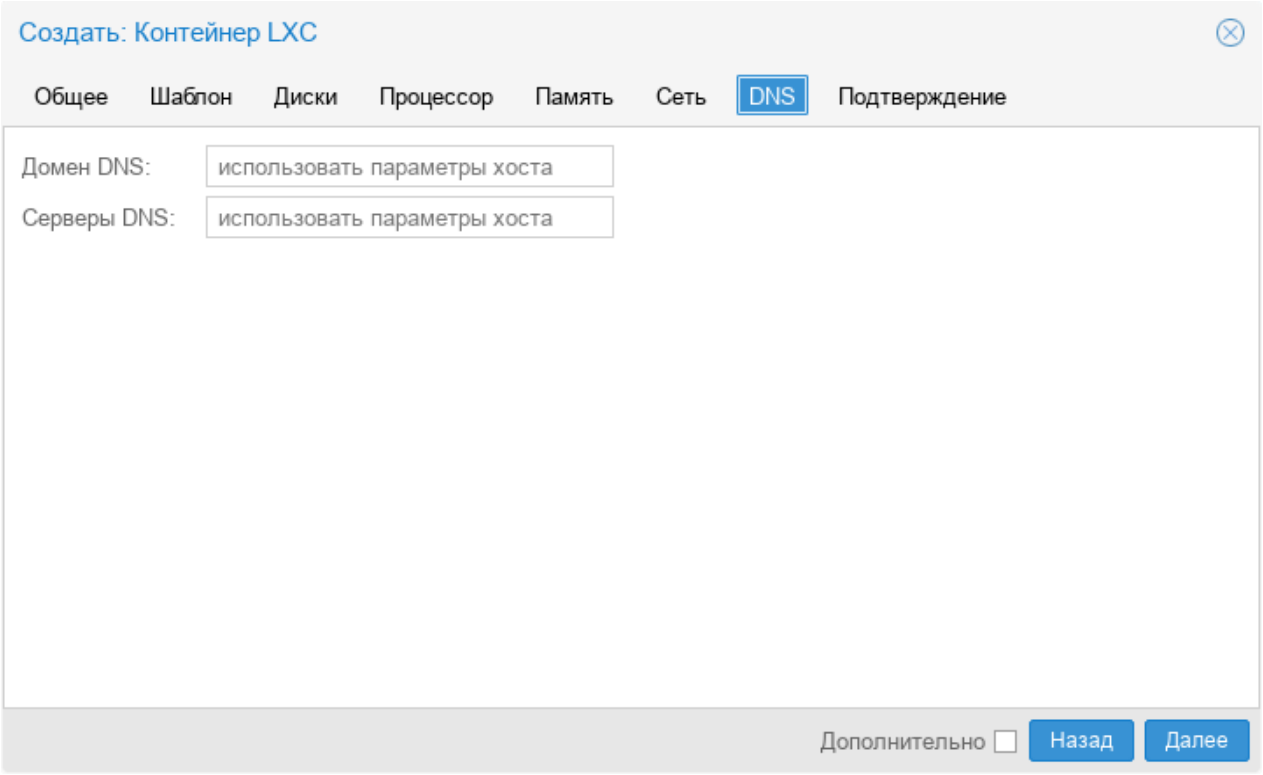

Рис. 296 – Вкладка «DNS» диалога создания контейнера

Во вкладке «Подтверждение» отображаются все введенные или выбранные значения для данного контейнера (рис. 297). Для создания контейнера необходимо нажать на кнопку «Готово». Если необходимо внести изменения в параметры контейнера, можно перейти по вкладкам назад.

Если отметить пункт «Запуск после создания» контейнер будет запущен сразу после создания.

После нажатия кнопки «Готово» во вкладке «Подтверждение», диалог настройки закрывается и в веб-браузере открывается новое окно, которое предлагает возможность наблюдать за построением PVE контейнера LXC из шаблона (рис. 298).

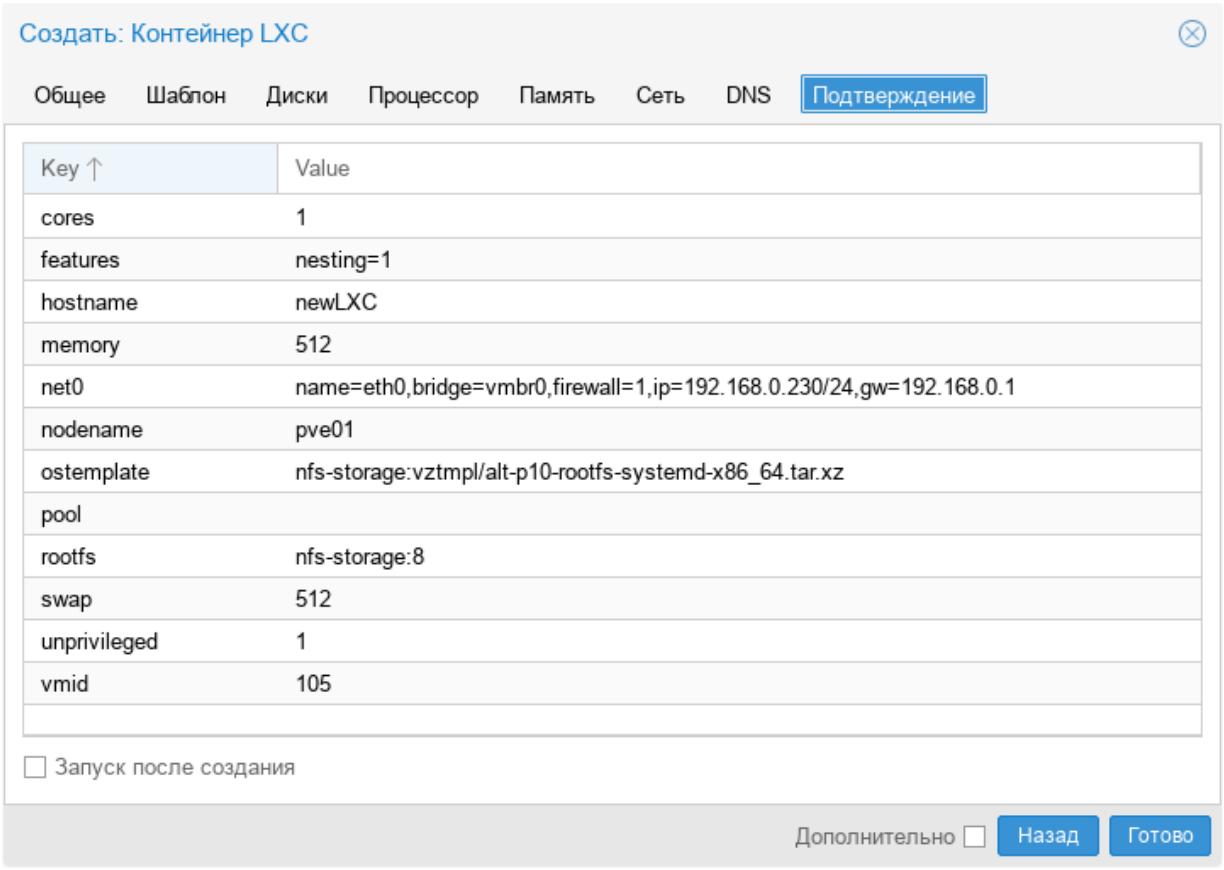

Рис. 297 – Вкладка «Подтверждение» диалога создания контейнера

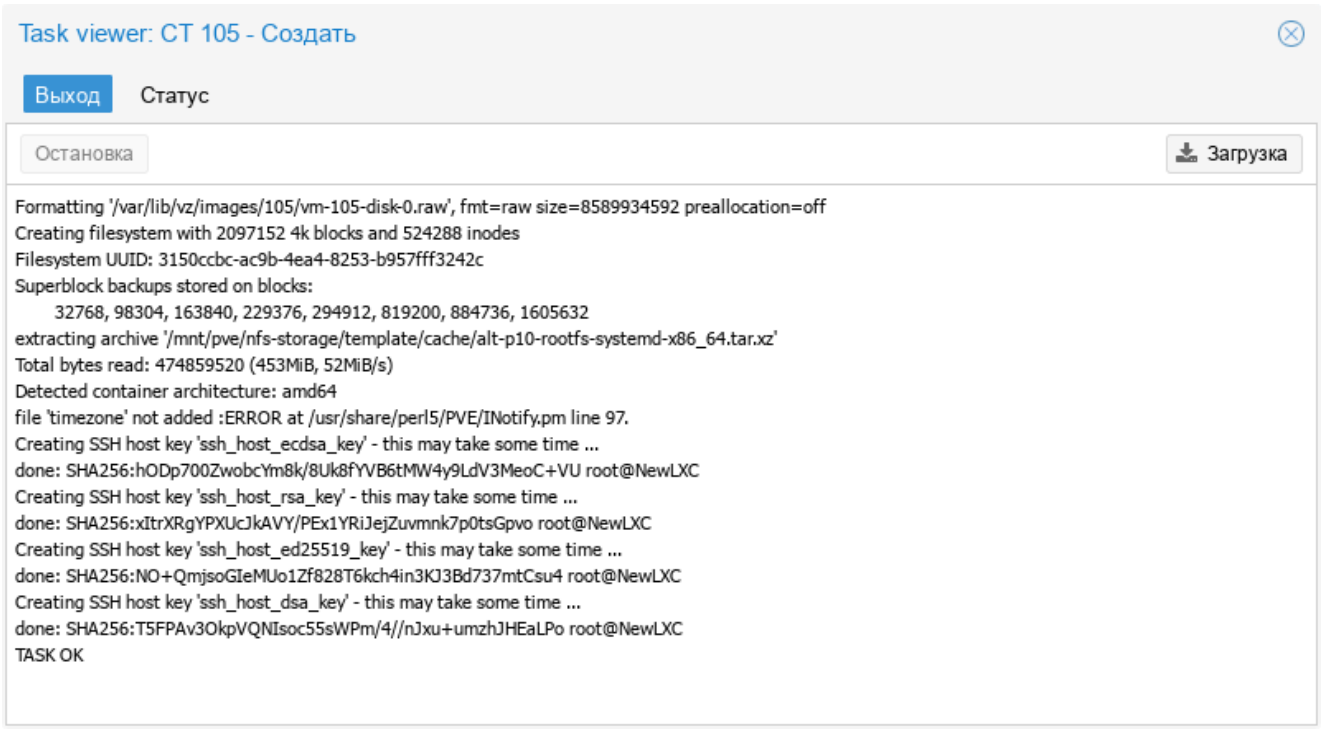

Рис. 298 – Создание контейнера

8.8.2. 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
```

8.8.3. 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.

8.8.3.1. 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. 299):

- «Ресурсы» (оперативная память, подкачка, количество ядер ЦПУ, размер диска);
- «Сеть»;
- «DNS»;
- «Параметры».

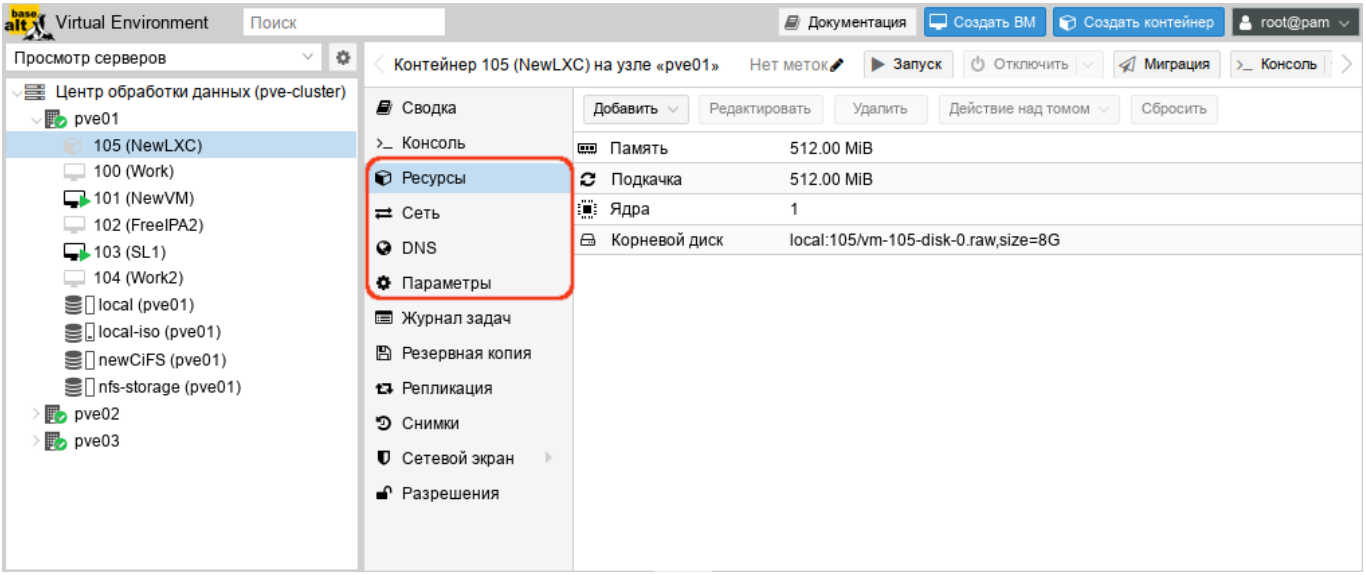

Рис. 299 – Изменений настроек контейнера в веб-интерфейсе PVE

Для редактирования ресурсов следует выполнить следующие действия:

- в режиме просмотра по серверам выбрать контейнер;
- перейти на вкладку «Ресурсы»;
- выбрать элемент для изменения: «Память», «Подкачка» или «Ядра», и нажать на кнопку «Редактировать»;
- в открывшемся диалоговом окне ввести нужные значения и нажать на кнопку «ОК».

Если необходимо изменить размер диска контейнера, например, увеличить до 18 Гбайт вместо предварительно созданного 8 Гбайт, нужно выбрать элемент «Корневой диск», нажать на кнопку «Действие над томом» → «Изменить размер», в открывшемся диалоговом окне ввести значение увеличения размера диска (рис. 300) и нажать на кнопку «Изменить размер диска».

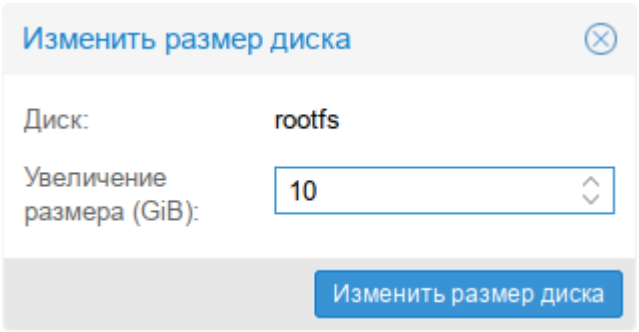

Рис. 300 – Изменение размера диска

Для перемещения образа диска в другое хранилище, нужно выбрать элемент «Корневой диск», нажать на кнопку «Действие над томом» → «Переместить хранилище», в открывшемся окне (рис. 301) в выпадающем меню «Целевое хранилище» выбрать хранилище-получатель, отметить, если это необходимо, пункт «Удалить источник» для удаления образа диска из исходного хранилища и нажать на кнопку «Переместить том».

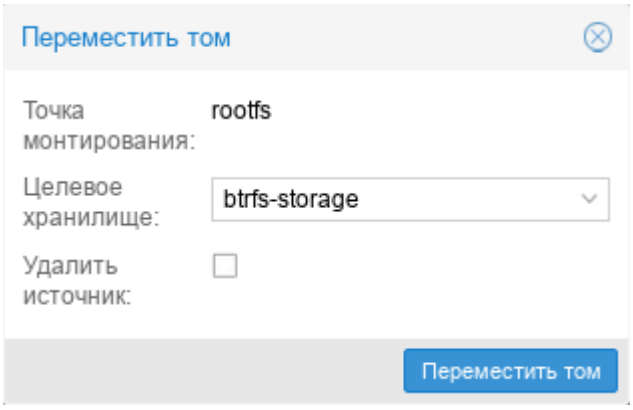

Рис. 301 – Диалоговое окно перемещения тома

Для изменения сетевых настроек контейнера необходимо:

- в режиме просмотра по серверам выбрать контейнер;
- перейти на вкладку «Сеть». На экране отобразятся все настроенные для контейнера виртуальные сетевые интерфейсы (рис. 302);
- выбрать интерфейс и нажать на кнопку «Редактировать» (рис. 303);
- после внесения изменений нажать на кнопку «ОК».

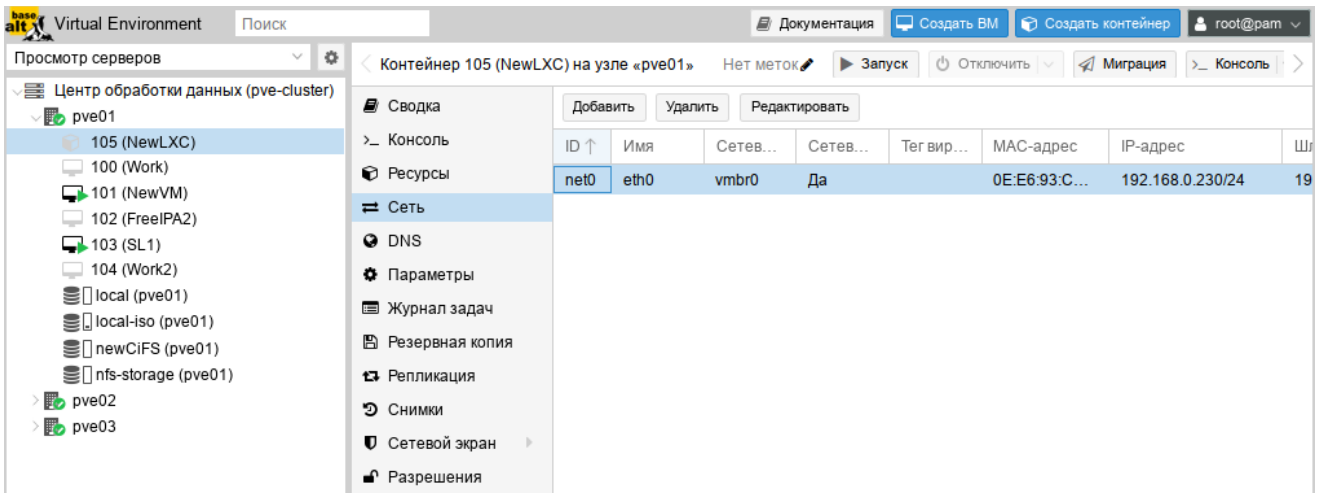

Рис. 302 – Виртуальные сетевые интерфейсы контейнера

На вкладке «Параметры» можно отредактировать разные настройки контейнера (рис. 304), например, «Режим консоли»:

- «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`.

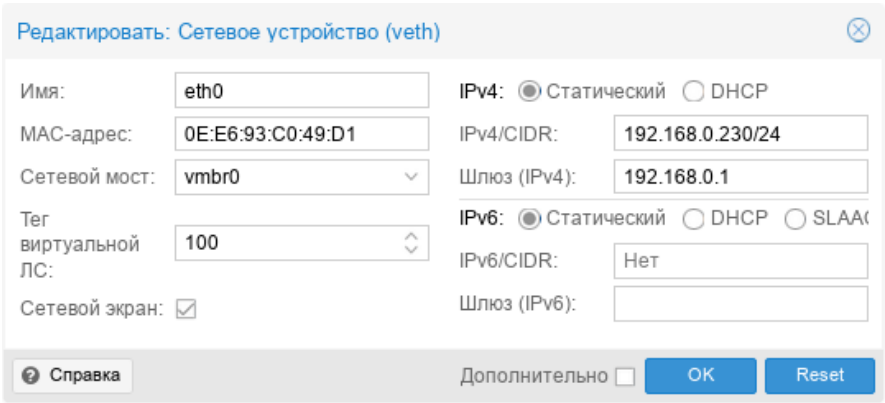

Рис. 303 – Изменение сетевых настроек контейнера

Примечание. В случаях, когда изменение не может быть выполнено в горячем режиме, оно будет зарегистрировано как ожидающее изменение (выделяется цветом, см. рис. 305). Такие изменения будут применены только после перезапуска контейнера.

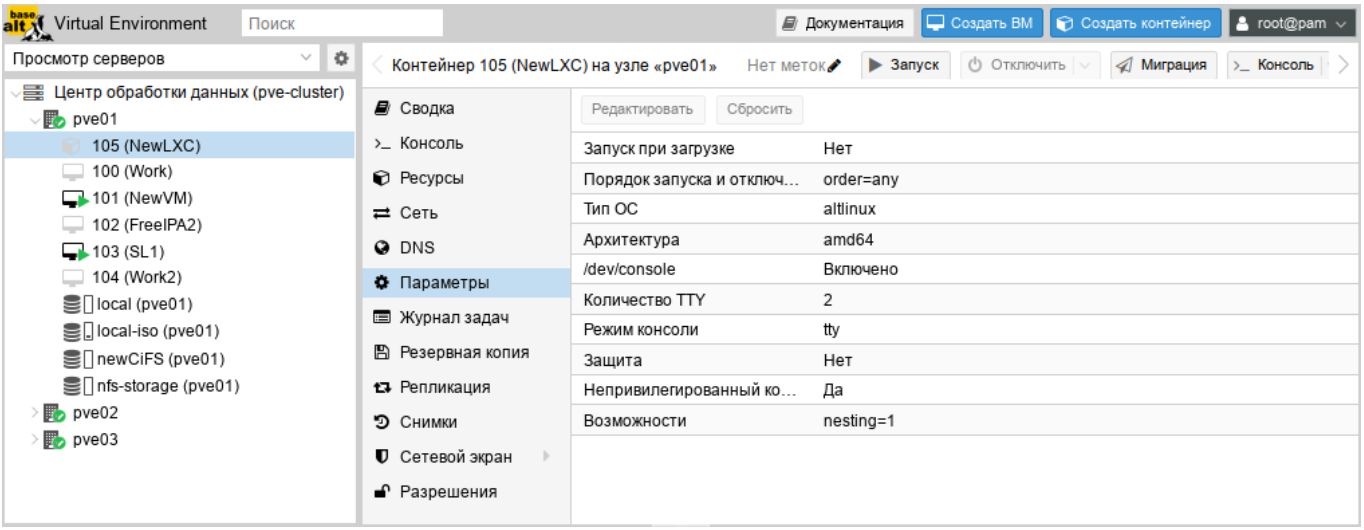

Рис. 304 – Изменение настроек контейнера. Вкладка «Параметры»

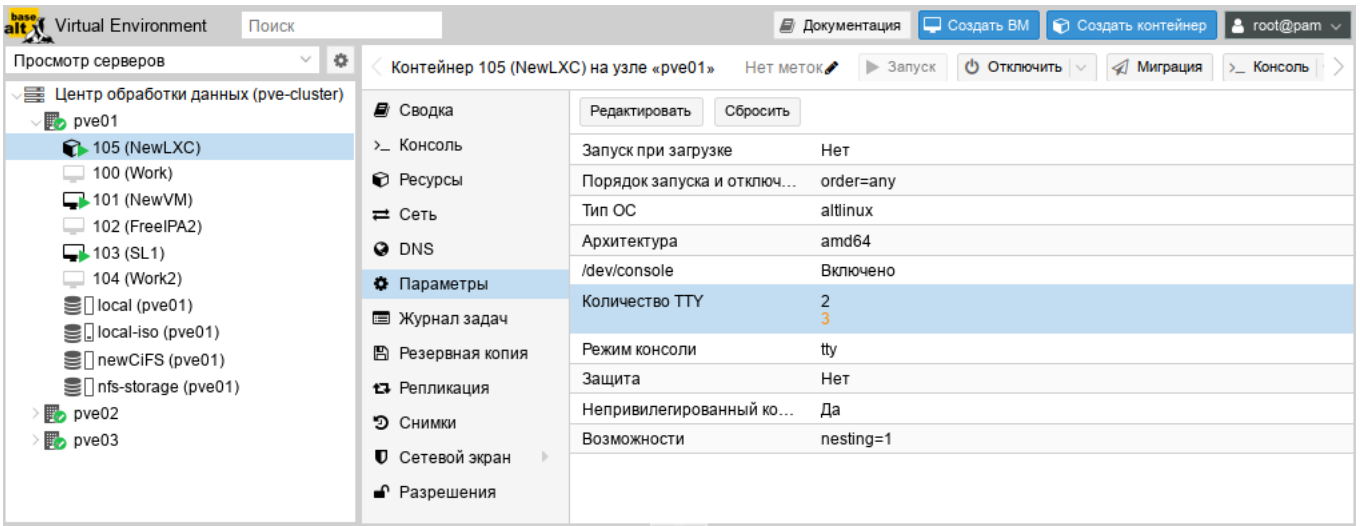

Рис. 305 – Изменения, которые будут применены после перезапуска контейнера

8.8.3.2. 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 Гбайт:

```
# 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>
```

8.8.3.3. 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 файле 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
```

8.8.4. Запуск и остановка контейнеров

8.8.4.1. 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. 306).

Запустить контейнер можно, выбрав в контекстном меню контейнера пункт «Запуск» (рис. 306), либо нажав кнопку «Запуск» (рис. 307).

Запущенный контейнер будет обозначен зеленой стрелкой на значке контейнера.

Для запущенного контейнера доступны следующие действия (рис. 308):

- «Отключить» – остановка контейнера;
- «Остановка» – остановка контейнера, путем прерывания его работы;
- «Перезагрузить» – перезапуск контейнера.

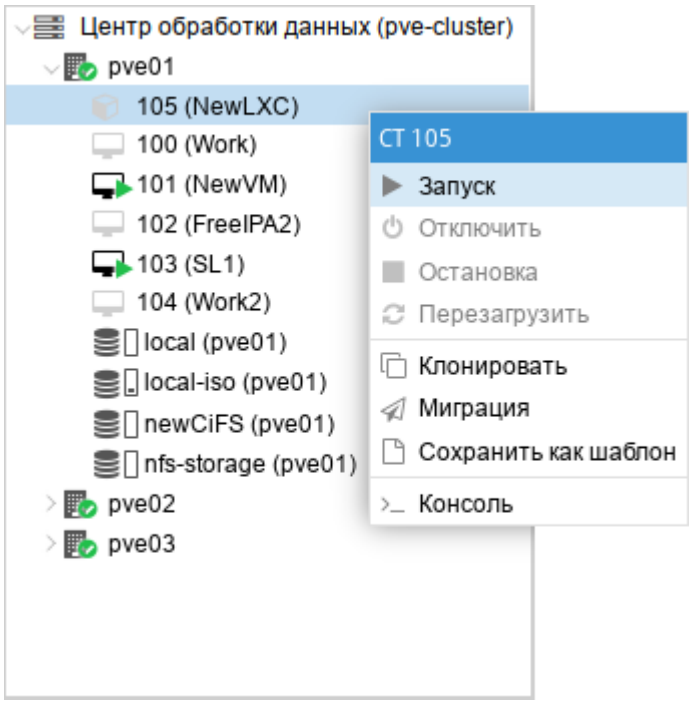

Рис. 306 – Контекстное меню контейнера

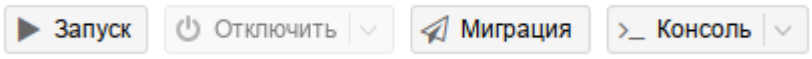

Рис. 307 – Кнопки управления состоянием контейнера

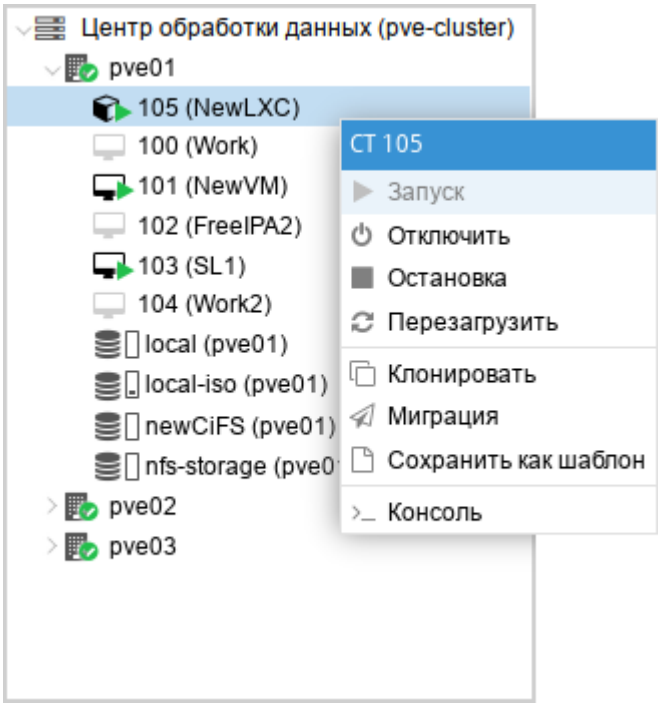

Рис. 308 – Контекстное меню запущенного контейнера

8.8.4.2. 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
```

8.8.5. 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 на кнопку «Консоль» и в выпадающем меню выбрать нужную консоль (рис. 309).

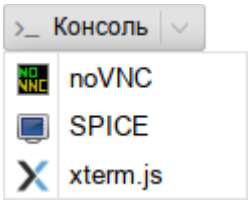

Рис. 309 – Кнопка «Консоль»

Консоль также можно запустить, выбрав вкладку «Консоль» для контейнера (рис. 310).

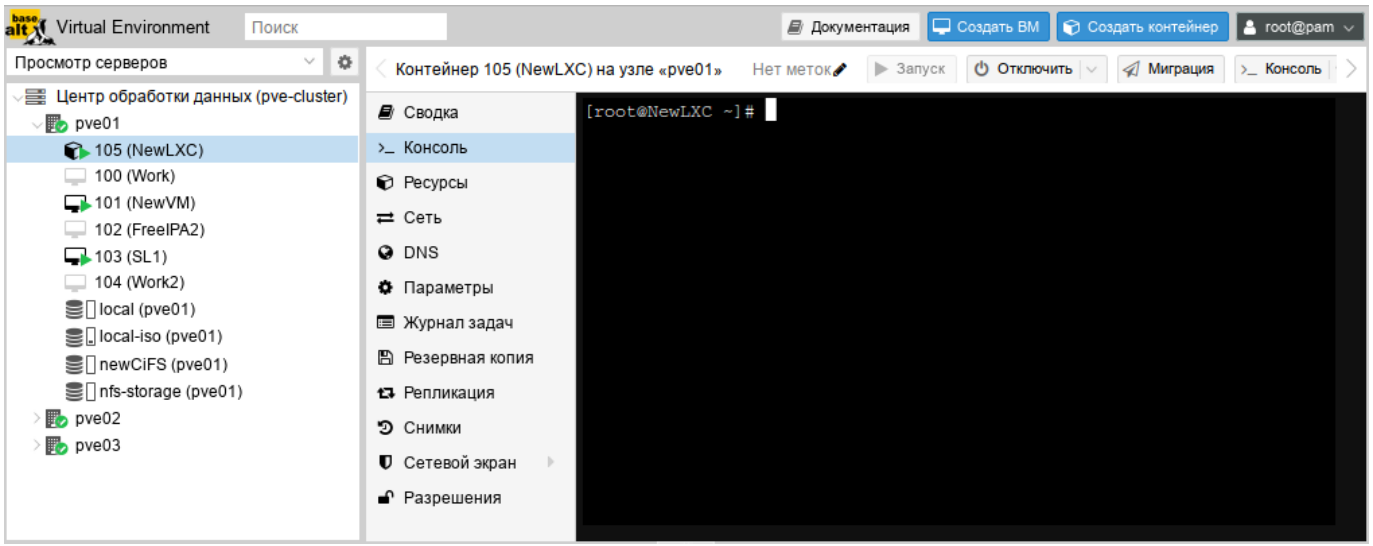

Рис. 310 – Консоль контейнера

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 105
[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 105 mkdir /home/demouser
# pct exec 105 ls /home
demouser
```

Для выполнения внутри контейнера команды с параметрами необходимо изменить команду `pct`, добавив `--` после идентификатора контейнера:

```
# pct exec 105 -- df -H /
Файловая система Размер Использовано  Дост Использовано% Смонтировано в
/dev/loop0         8,4G         516M  7,4G            7%              /
```

8.9. 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 невозможна. 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.

8.9.1. Миграция с применением графического интерфейса

Для миграции ВМ или контейнера необходимо выполнить следующие шаги:

- выбрать ВМ или контейнер для миграции и нажать на кнопку «Миграция» (рис. 311);
- в открывшемся диалоговом окне (рис. 312) выбрать узел назначения, на который будет осуществляться миграция, и нажать на кнопку «Миграция».

Примечание. Режим миграции будет выбран автоматически (рис. 312, рис. 313, рис. 314) в зависимости от состояния ВМ/контейнера (запущен/остановлен).

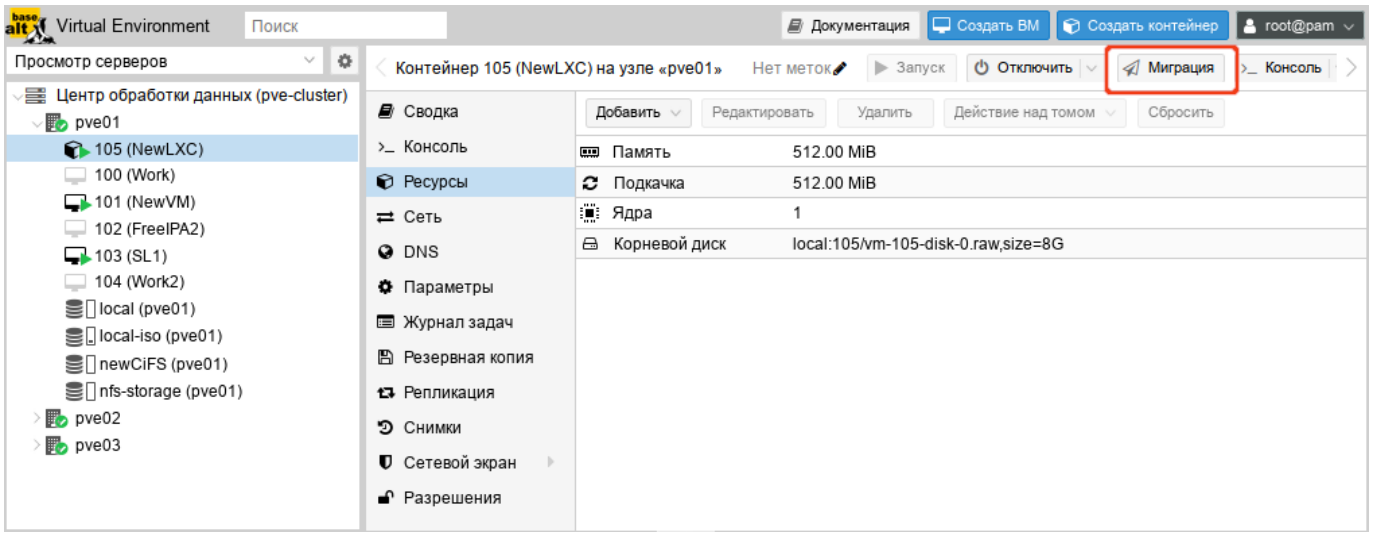

Рис. 311 – Выбор ВМ или контейнера для миграции

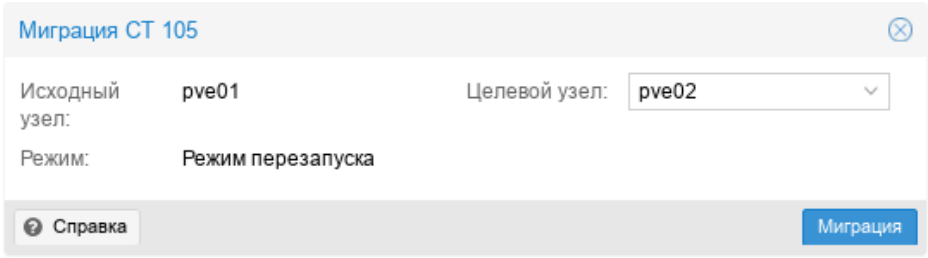

Рис. 312 – Миграция контейнера с перезапуском

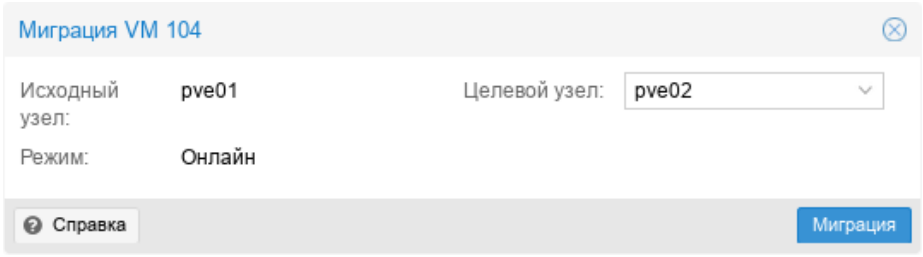

Рис. 313 – Миграция ВМ онлайн

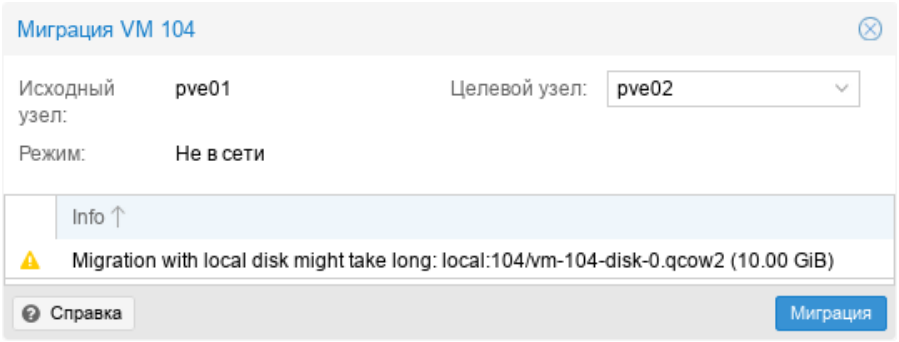

Рис. 314 – Миграция ВМ офлайн

8.9.2. 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 параметр `--online`.

Чтобы осуществить миграцию контейнера необходимо выполнить следующую команду:

```
# pct migrate <ctid> <target> [OPTIONS]
```

Поскольку миграция контейнеров в реальном времени невозможна, миграцию работающего контейнера с перезапуском можно выполнить, добавив параметр `--restart`. Например:

```
# pct migrate 101 pve02 --restart
```

8.9.3. 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.

8.9.3.1. 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), например:

```
$ oneimage show 14
IMAGE 14 INFORMATION
ID             : 14
NAME           : ALT Linux p10
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 жесткого 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:

- создать новую ВМ в веб-интерфейсе PVE или командой:

```
# qm create 120 --bootdisk scsi0 --net0 virtio,bridge=vmbr0 --scsihw virtio-scsi-pci
```

- 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'
```

- привязать диск к ВМ:

а) 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 на кнопку «Добавить» (рис. 315);

б) 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
```

Донастроить параметры процессора, памяти, сетевых интерфейсов, порядок загрузки. Включить ВМ.

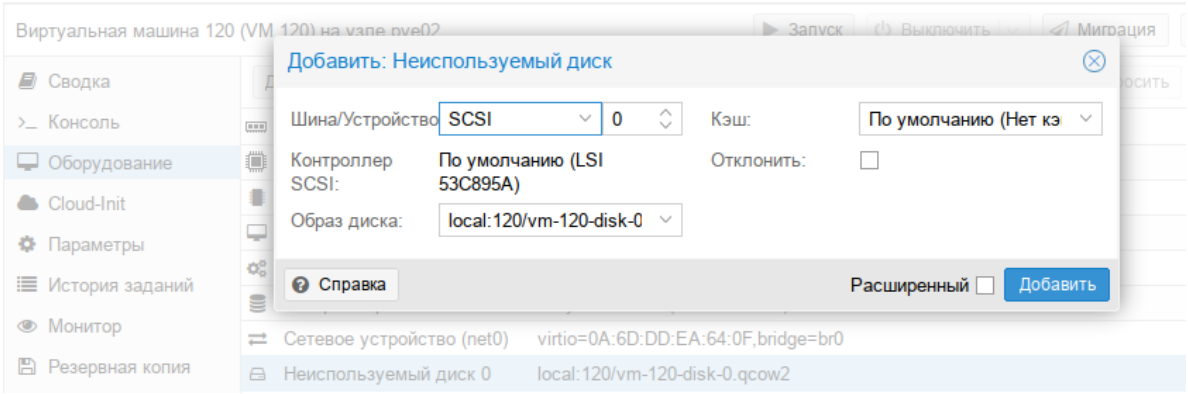

Рис. 315 – Добавление диска к ВМ

8.9.3.2. 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 трех указанных способов:

- подключение образа диска к ВМ на основе Directory Storage:

  а) в веб-интерфейсе PVE создать ВМ KVM;

  б) 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);

  в) преобразовать файл `win7-pve.vmdk` в qemu формат:

  ```
  # qemu-img convert -f vmdk win7-pve.vmdk -O qcow2 win7-pve.qcow2
  ```

  г) 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;

  д) 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 на кнопку «Добавить» (рис. 316);

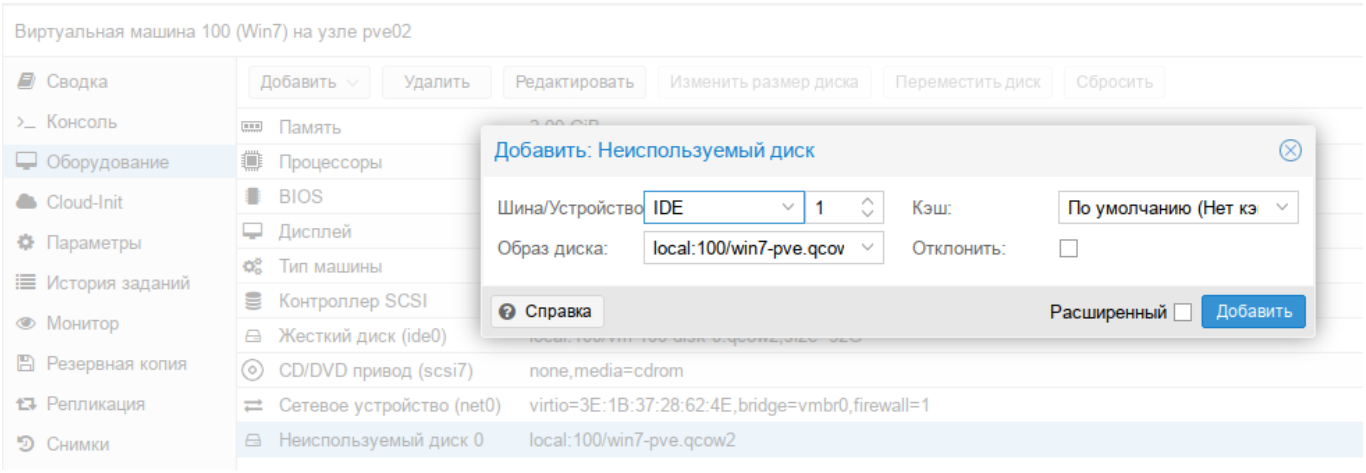

Рис. 316 – Добавление диска к ВМ

- подключение образа диска к ВМ на основе LVM Storage:

а) 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 127 Гбайт;

б) скопировать преобразованный образ win7-pve.vmdk в каталог с образами ВМ `/var/lib/vz/images/VMID`, где VMID – VMID, созданной ВМ (можно воспользоваться WinSCP);

в) 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
```

г) 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
```

- подключение образа диска к ВМ на основе CEPH Storage:

а) 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
```

б) скопировать преобразованный образ win7-pve.vmdk в каталог с образами ВМ `/var/lib/vz/images/VMID`, где VMID – VMID, созданной виртуальной машины;

в) 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 «Центр обработки данных» → «Хранилище» → «rbd-storage».

г) 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).

8.9.3.3. 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
```

П р и м е ч а н и е . Сеть необходимо настроить вручную.

8.10. Клонирование ВМ

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. 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» (рис. 317);
3) откроется диалоговое окно (рис. 318), со следующими полями:
   - «Целевой узел» – 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» – название новой ВМ;
   - «Пул ресурсов» – пул, к которому будет относиться ВМ;
   - «Режим» – метод клонирования (если клонирование происходит из шаблона ВМ). Доступны значения: «Полное клонирование» и «Связанная копия»;
   - «Снимок» – снимок из которого будет создаваться клон (если снимки существуют);
   - «Целевое хранилище» – хранилище для клонируемых виртуальных дисков;
   - «Формат» – формат образа виртуального диска;

4) для запуска процесса клонирования необходимо нажать на кнопку «Клонировать».

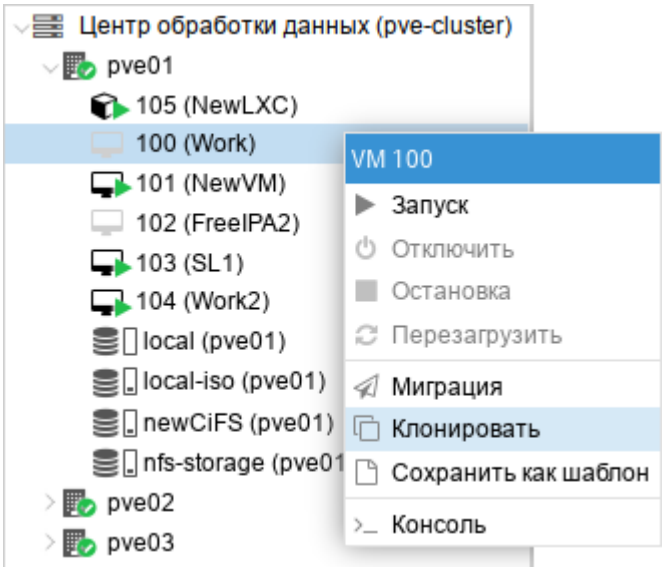

Рис. 317 – Настройки клонирования

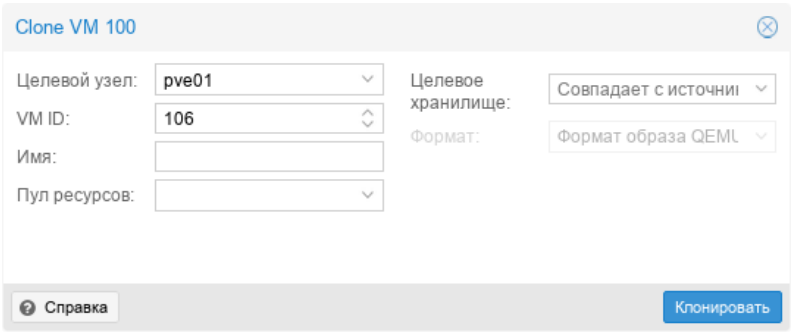

Рис. 318

Некоторые типы хранилищ позволяют копировать определенный снимок ВМ (рис. 319), который по умолчанию соответствует текущим данным ВМ. Клон ВМ никогда не содержит дополнительных снимков оригинальной ВМ.

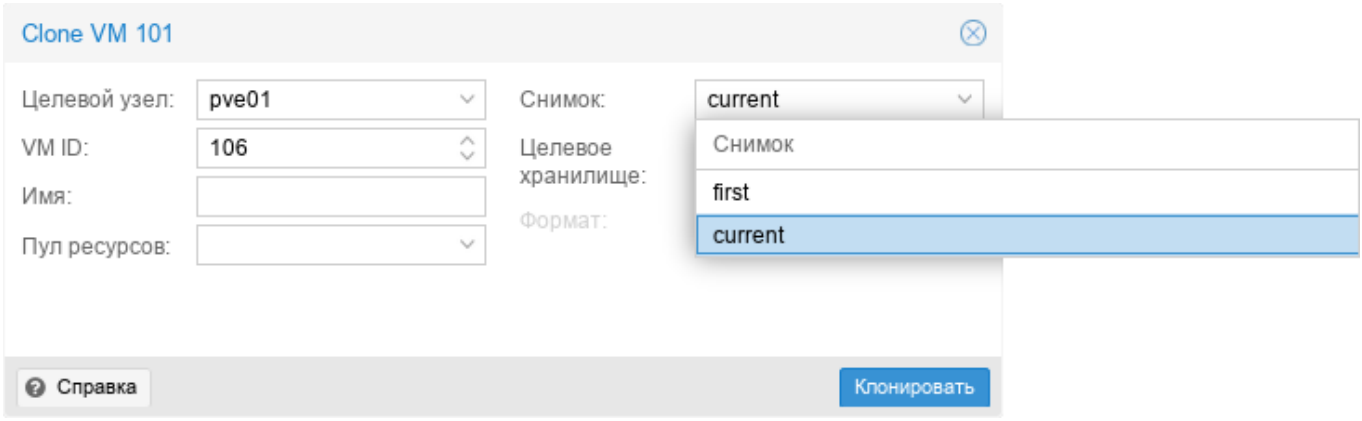

Рис. 319 – Выбор снимка для клонирования

Чтобы избежать конфликтов ресурсов, при клонировании MAC-адреса сетевых интерфейсов рандомизируются, и генерируется новый UUID для настройки BIOS ВМ (smbios1).

8.11. Шаблоны ВМ

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» (рис. 320) и в ответ на запрос на подтверждения, нажать на 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.

Для создания связанного клона необходимо выполнить следующие шаги:

1) в контекстном меню шаблона ВМ выбрать пункт «Клонировать» (рис. 321);
2) в открывшемся диалоговом окне (рис. 322) указать параметры клонирования (в поле «Режим» следует выбрать значение «Связанная копия»);
3) для запуска процесса клонирования нажать на кнопку «Клонировать».

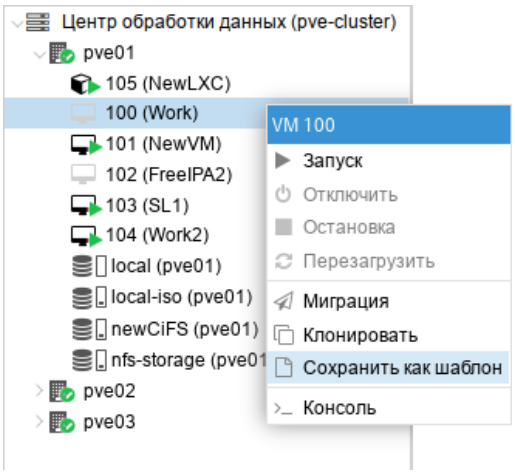

Рис. 320 – Создание шаблона ВМ

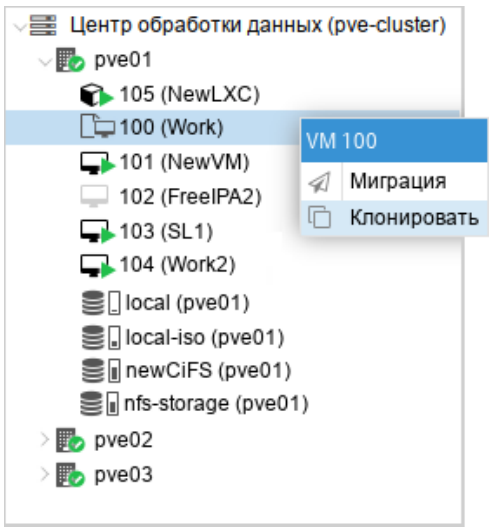

Рис. 321 – Создание связанного клона

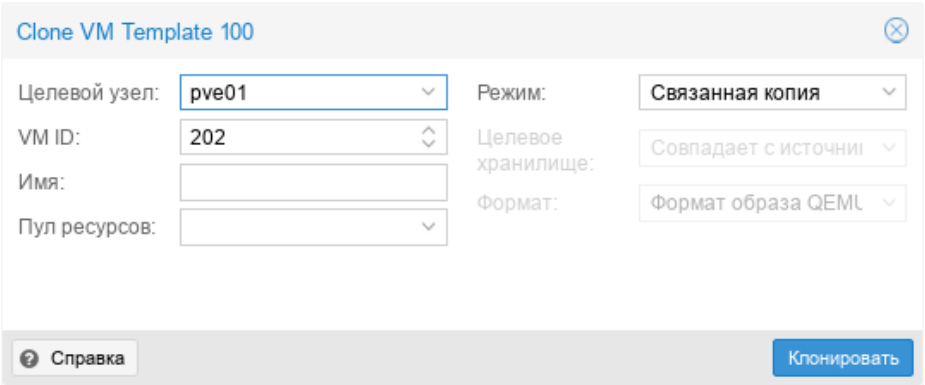

Рис. 322

8.12. Теги (метки) ВМ

В организационных целях для ВМ (KVM и LXC) можно установить теги (метки). Теги отображаются в дереве ресурсов и в строке статуса при выборе ВМ (рис. 323). Теги позволяют фильтровать ВМ (рис. 324).

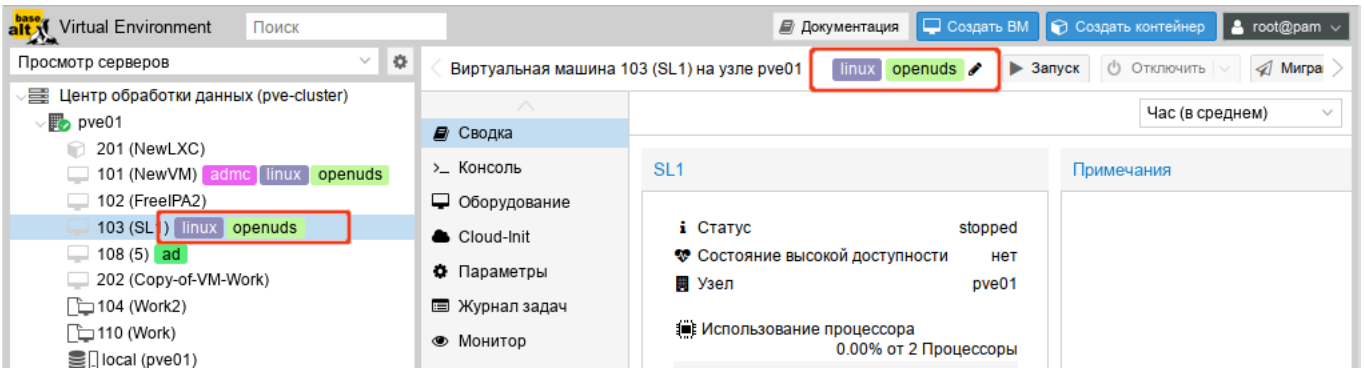

Рис. 323 – Теги ВМ 103

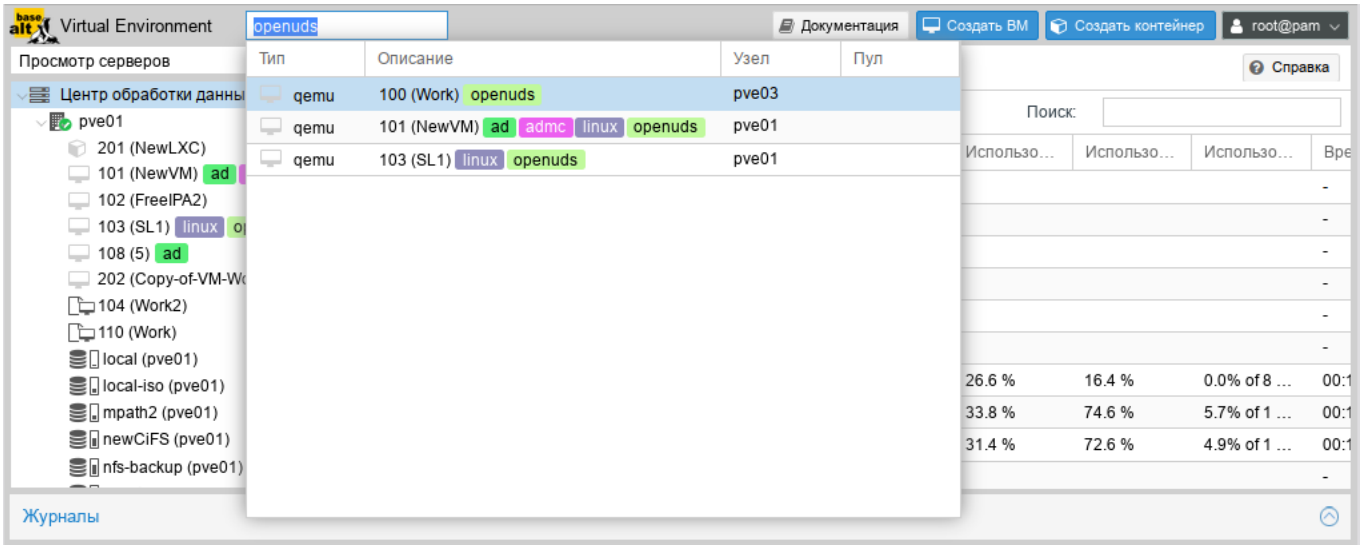

Рис. 324 – Фильтрация ВМ по тегам (меткам)

8.12.1. Работа с тегами

Для добавления, редактирования, удаления тегов необходимо в строке статуса ВМ нажать на значок карандаша (рис. 325). Можно добавить несколько тегов, нажав кнопку «+», и удалить их, нажав кнопку «-». Чтобы сохранить или отменить изменения, используются кнопки «✓» и «x» соответственно (рис. 326).

Рис. 325 – Строка статуса ВМ

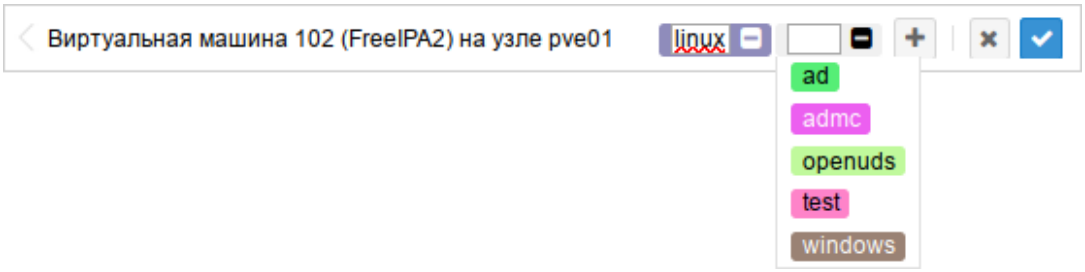

Рис. 326 – Теги ВМ 102

Теги также можно устанавливать в командной строке (несколько тегов разделяются точкой с запятой):

```
# qm set ID --tags myfirsttag;mysecondtag
```

Например:

```
# qm set 103 --tags linux;openuds
```

8.12.2. Настройка тегов

В глобальных параметрах центра обработки данных PVE (раздел «Центр обработки данных» → «Параметры») есть три пункта меню, посвященных тегам (рис. 327). Здесь можно, среди прочего, заранее определить теги и напрямую назначить им цвет.

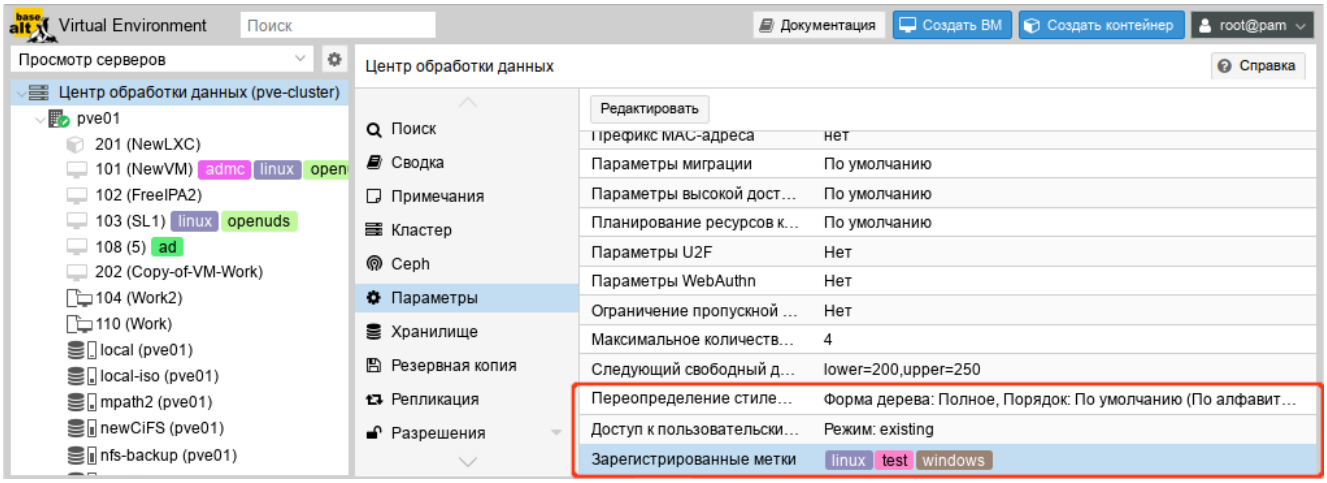

Рис. 327 – Настройки тегов

8.12.2.1. Стиль тегов

Цвет, форму изображения тегов в дереве ресурсов, чувствительность к регистру, а также способ сортировки тегов можно настроить в параметре «Переопределение стилей меток» (рис. 328):

1) «Форма дерева» – позволяет указать форму отображения тегов:
   - «Полное» – отображать полнотекстовую версию;
   - «Круговое» – использовать только цветовой акцент: круг с цветом фона (по умолчанию);
   - «Плотное» – использовать только цветовой акцент: небольшой прямоугольник (полезно, когда каждой ВМ назначено много тегов);
   - «Нет» – отключить отображение тегов;
2) «Порядок» – управляет сортировкой тегов в веб-интерфейсе;
3) «С учетом регистра» – позволяет указать, должна ли фильтрация уникальных тегов учитывать регистр символов;
4) «Переопределение цветов» – позволяет задать цвета для тегов (по умолчанию цвета тегов автоматически выбирает PVE).

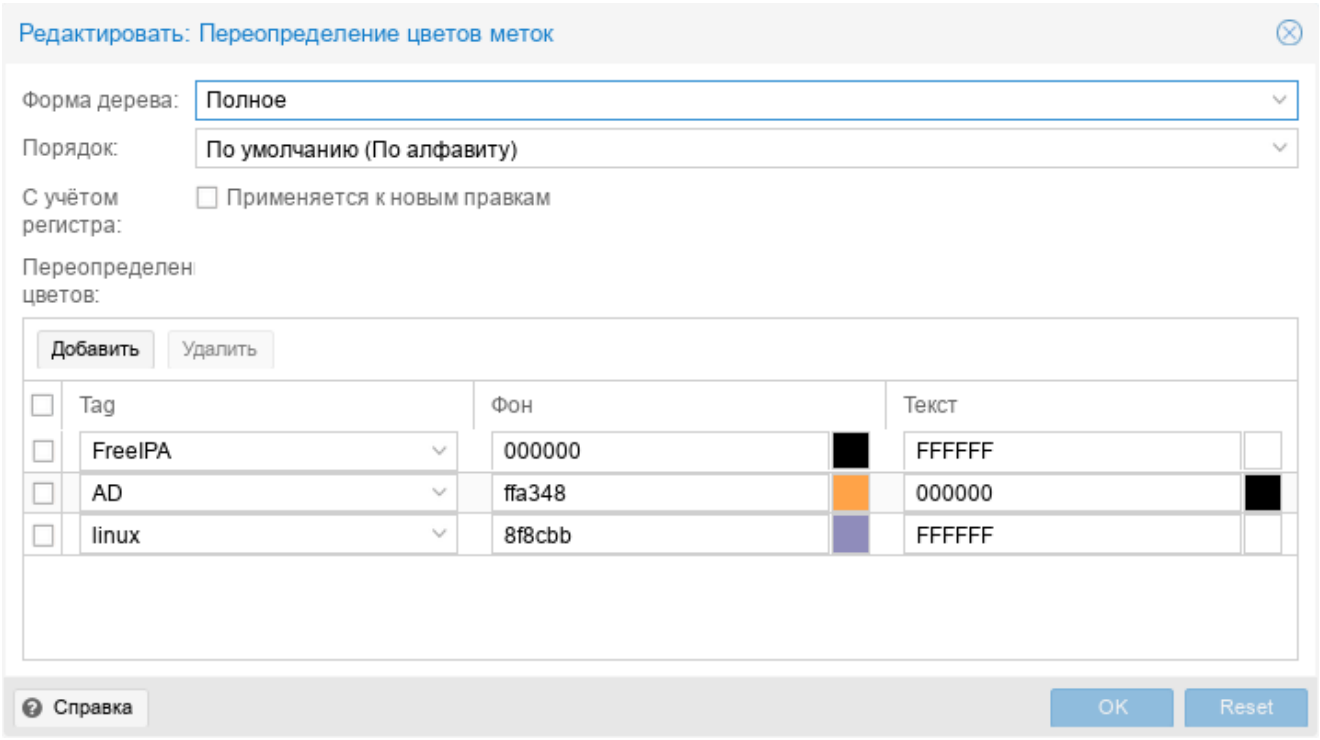

Рис. 328 – Переопределение стилей меток

Настроить стиль тегов можно также в командной строке, используя команду:

```
# pvesh set /cluster/options --tag-style [case-sensitive=<1|0>]\
[,color-map=<tag>:<hex-color> [:<hex-color-for-text>][;<tag>=...]]\
[,ordering=<config|alphabetical>][,shape=<circle|dense|full|none>]
```

Например, следущая команда установит для тега «FreeIPA» цвет фона черный (#000000), а цвет текста – белый (#FFFFFF) и форму тегов «Плотное»:

```
# pvesh set /cluster/options --tag-style color-map=FreeIPA:000000:FFFFFF,shape=dense
```

Примечание. Команда `pvesh set` удалит все ранее переопределенные стили тегов.

8.12.2.2. Права

По умолчанию пользователи с привилегиями VM.Config.Options могут устанавливать любые теги для ВМ (/vms/ID). Если необходимо ограничить такое поведение, соответствующие разрешения можно установить в разделе «Доступ к пользовательским меткам» (рис. 329). Доступны следующие режимы (поле «Режим»):

1) «free» – пользователи не ограничены в установке тегов (по умолчанию);
2) «list» – пользователи могут устанавливать теги на основе заранее определенного списка тегов;
3) «existing» – аналогично режиму «list», но пользователи также могут использовать уже существующие теги;
4) «none» – пользователям запрещено использовать теги.

Здесь же можно определить, какие теги разрешено использовать пользователям (поле «Предустановленные метки») если используются режимы «list» или «existing».

Назначить права можно также и в командной строке, используя команду:

```
# pvesh set /cluster/options --user-tag-access\
[user-allow=<existing|free|list|none>][,user-allow-list=<tag>[;<tag>...]]
```

Например, запретить пользователям использовать теги:

```
# pvesh set /cluster/options --user-tag-access user-allow=none
```

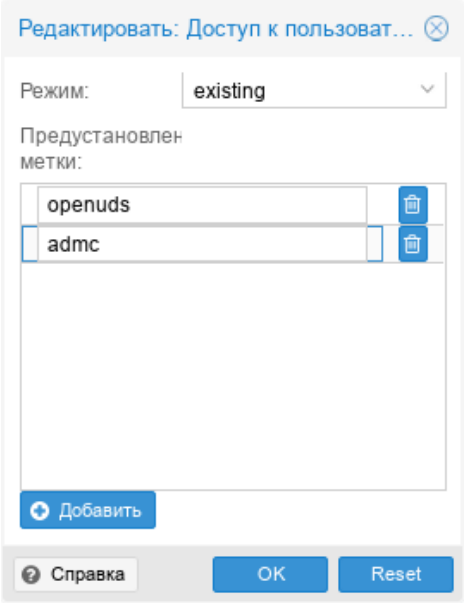

Рис. 329 – Доступ к пользовательским меткам

Следует обратить внимание, что пользователь с привилегиями Sys.Modify на / всегда может устанавливать или удалять любые теги, независимо от настроек в разделе «Доступ к пользовательским меткам». Кроме того, существует настраиваемый список зарегистрированных тегов, которые могут добавлять и удалять только пользователи с привилегией Sys.Modify на /. Список зарегистрированных тегов можно редактировать в разделе «Зарегистрированные метки» (рис. 330) или через интерфейс командной строки:

```
# pvesh set /cluster/options --registered-tags <tag>[;<tag>...]
```

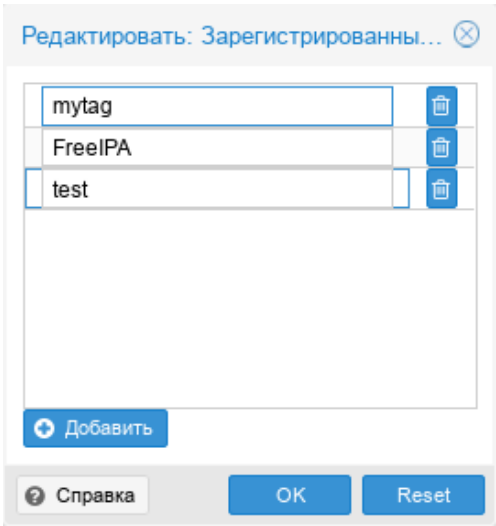

Рис. 330 – Зарегистрированные метки

8.13. 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`.

8.13.1. 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.

8.13.2. 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.

8.13.3. 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. 331).

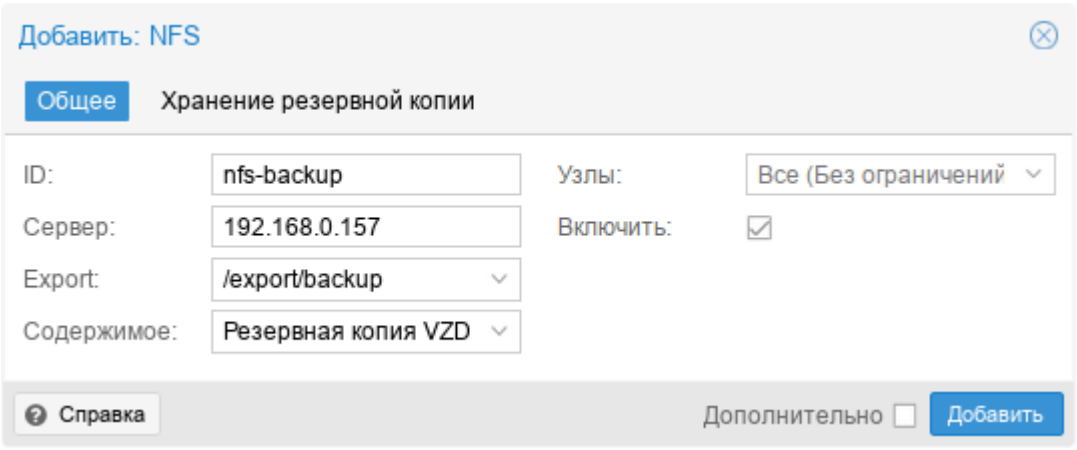

Рис. 331 – Настройка хранилища NFS

На вкладке «Хранение резервной копии» можно указать параметры хранения резервных копий (рис. 332).

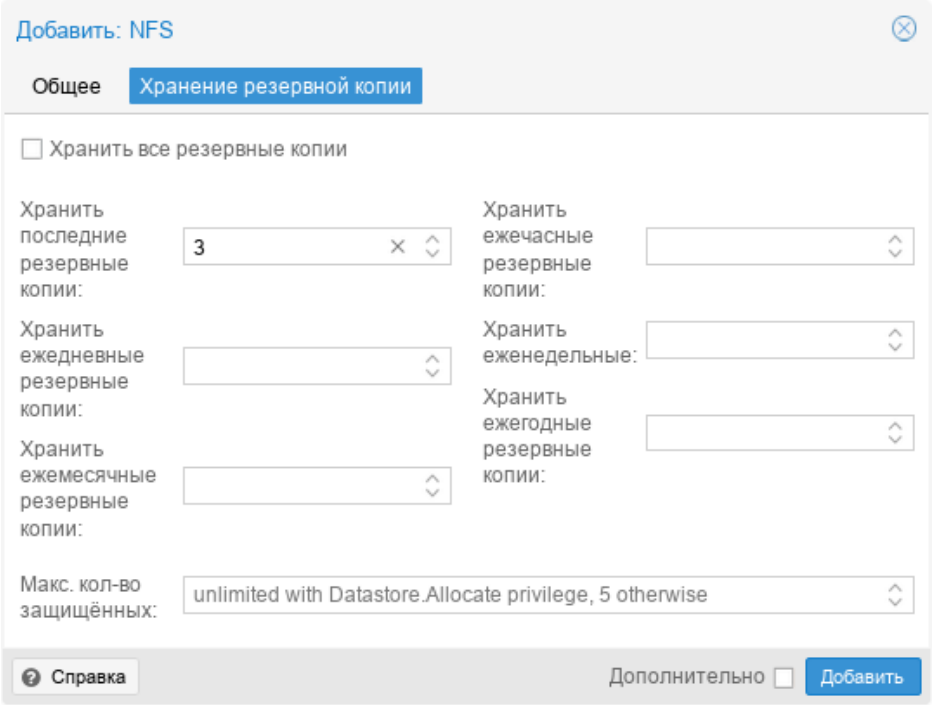

Рис. 332 – Параметры хранения резервных копий в хранилище NFS

Доступны следующие варианты хранения резервных копий (в скобках указаны параметры опции prune-backups команды `vzdump`):

- «Хранить все резервные копии» (`keep-all=<1|0>`) – хранить все резервные копии (если отмечен этот пункт, другие параметры не могут быть установлены);
- «Хранить последние резервные копии» (`keep-last=<N>`) – хранить <N> последних резервных копий;
- «Хранить ежечасные резервные копии» (`keep-hourly=<N>`) – хранить резервные копии за последние <N> часов (если за один час создается более одной резервной копии, сохраняется только последняя);
- «Хранить ежедневные резервные копии» (`keep-daily=<N>`) – хранить резервные копии за последние <N> дней (если за один день создается более одной резервной копии, сохраняется только самая последняя);
- «Хранить еженедельные» (`keep-weekly=<N>`) – хранить резервные копии за последние <N> недель (если за одну неделю создается более одной резервной копии, сохраняется только последняя);

- «Хранить ежемесячные резервные копии» (`keep-monthly=<N>`) – хранить резервные копии за последние <N> месяцев (если за один месяц создается более одной резервной копии, сохраняется только самая последняя);
- «Хранить ежегодные резервные копии» (`keep-yearly <N>`) – хранить резервные копии за последние <N> лет (если за один год создается более одной резервной копии, сохраняется только самая последняя).

«Макс. кол-во защищенных» (параметр хранилища: `max-protected-backups`) – количество защищенных резервных копий на гостевую систему, которое разрешено в хранилище. Для указания неограниченного количества используется значение `-1`. Значение по умолчанию: неограниченно для пользователей с привилегией `Datastore.Allocate` и 5 для других пользователей.

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 при создании задания, рекомендуется настроить эти параметры на уровне хранилища.

8.13.4. 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`.

8.13.5. 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 трехбуквенном варианте на английском: mon, tue, wed, thu, fri, sat и sun. Можно использовать несколько дней в виде списка, разделенного запятыми. Можно задать диапазон дней, указав день начала и окончания, разделенные 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 значения времени приведены в таблице 22. В таблице 23 приведены примеры периодов времени.

Т а б л и ц а 22 – Специальные значения времени

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
semiannually или semi-annually	Раз в полгода	*-01,07-01 00:00:00

Т а б л и ц а 23 – Примеры периодов времени

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

8.13.6. Настройка резервного копирования в графическом интерфейсе

Для того чтобы создать расписание резервного копирования, необходимо перейти во вкладку «Центр обработки данных» → «Резервная копия» (рис. 333) и нажать на кнопку «Добавить».

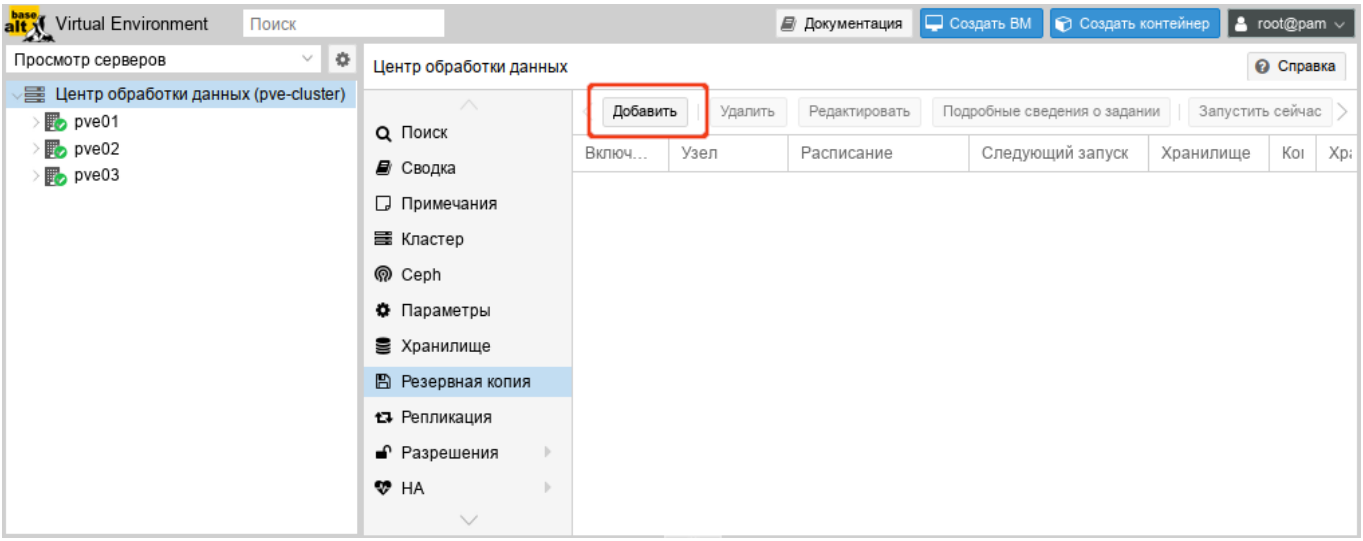

Рис. 333 – Вкладка «Резервная копия»

При создании задания на резервирование, необходимо указать (рис. 334):

- «Узел» – можно создавать график из одного места по разным узлам (серверам);
- «Хранилище» – точка смонтированного накопителя, куда будет проходить копирование;
- «Расписание» – здесь можно указать расписание резервного копирования. Можно выбрать период из списка (рис. 335) или указать вручную;
- «Режим выбора» – возможные значения: «Учитывать выбранные ВМ», «Все», «Исключить выбранные ВМ», «На основе пула»;
- «Отправить письмо» – адрес, на который будут приходить отчеты о выполнении резервного копирования;
- «Адрес эл.почты» – принимает два значения: «Уведомлять всегда» – сообщение будет приходить при любом результате резервного копирования, «Только при ошибках» – сообщение будет приходить только в случае неудачной попытки резервного копирования;
- «Сжатие» – метод сжатия, принимает четыре значения: «ZSTD (быстро и хорошо)» (по умолчанию), «LZO (быстро)», «GZIP (хорошо)» и «нет»;
- «Режим» – режим ВМ, в котором будет выполняться резервное копирование. Может принимать три значения (рис. 336): «Снимок», «Приостановить», «Остановка».

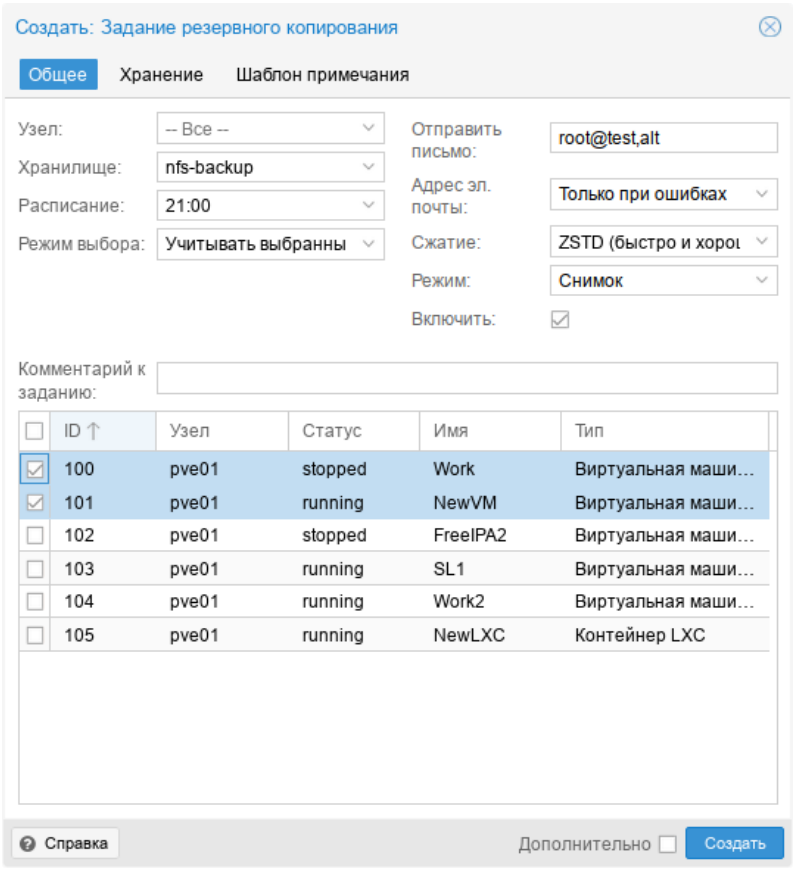

Рис. 334 – Создание задания для резервного копирования. Вкладка «Общее»

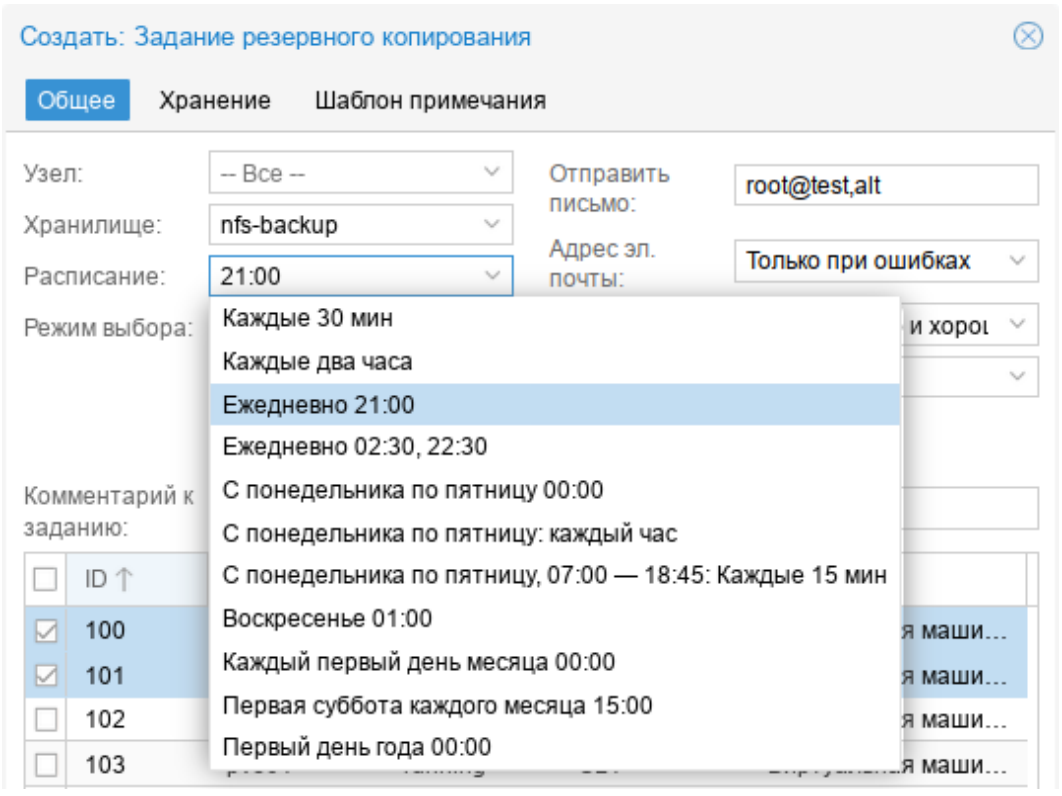

Рис. 335 – Выбор расписания резервного копирования

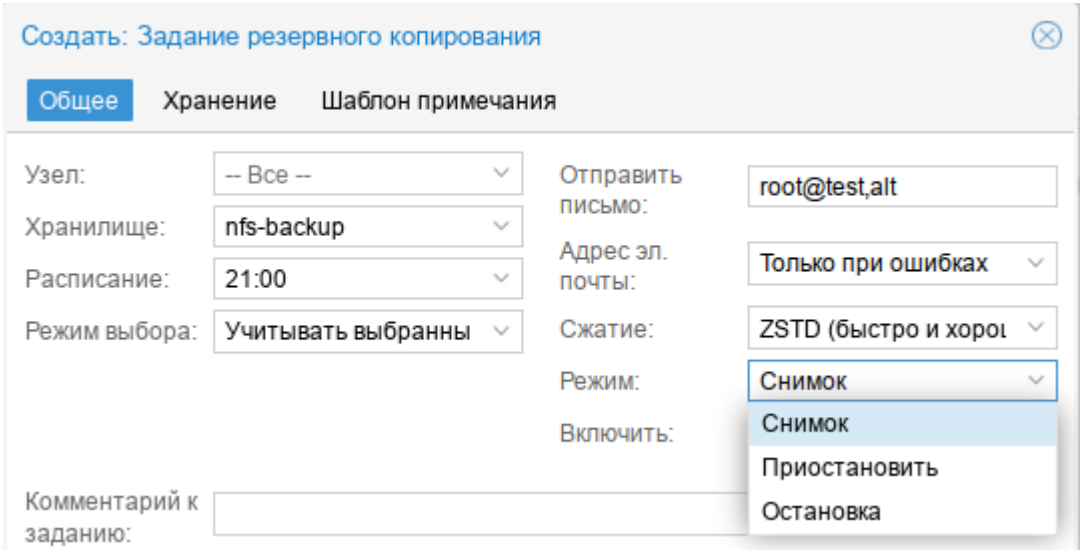

Рис. 336 – Выбор режима создания резервной копии

На вкладке «Хранение» можно настроить параметры хранения резервных копий (рис. 337).

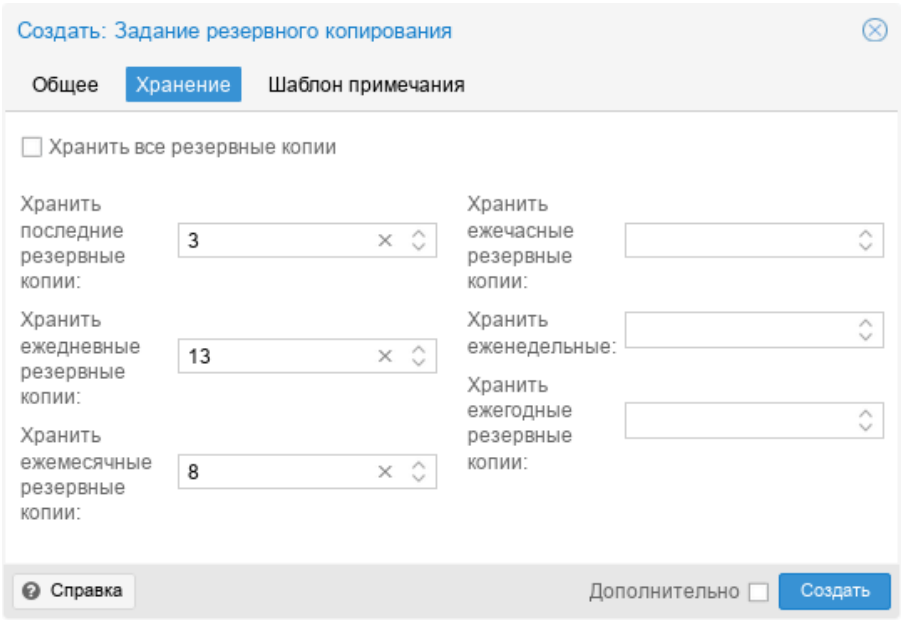

Рис. 337 – Создание задания для резервного копирования. Вкладка «Хранение»

На вкладке «Шаблон примечания» можно настроить примечание, которое будет добавляться к резервным копиям. Строка примечания может содержать переменные, заключенные в две фигурные скобки (рис. 338).

Поддерживаются следующие переменные:

- {{cluster}} – имя кластера;
- {{guestname}} – имя ВМ/контейнера;
- {{node}} – имя узла, для которого создается резервная копия;
- {{vmid}} – VMID ВМ/контейнера.

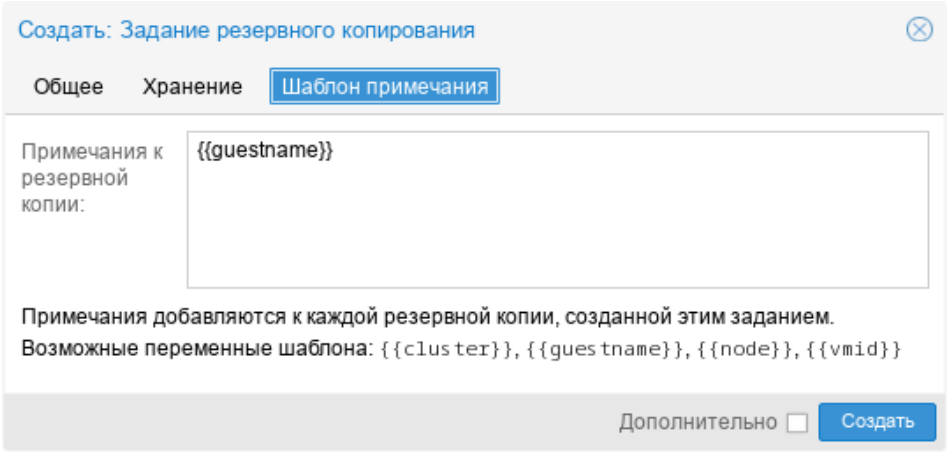

Рис. 338 – Создание задания для резервного копирования. Вкладка «Шаблон примечания»

После указания необходимых параметров и нажатия кнопки «Создать», задание для резервного копирования появляется в списке (рис. 339). Запись о задании создается в файле `/etc/pve/jobs.cfg`.

Данное задание будет запускаться в назначенное время. Время следующего запуска задания отображается в столбце «Следующий запуск». Также существует возможность запустить задание по требованию – кнопка «Запустить сейчас».

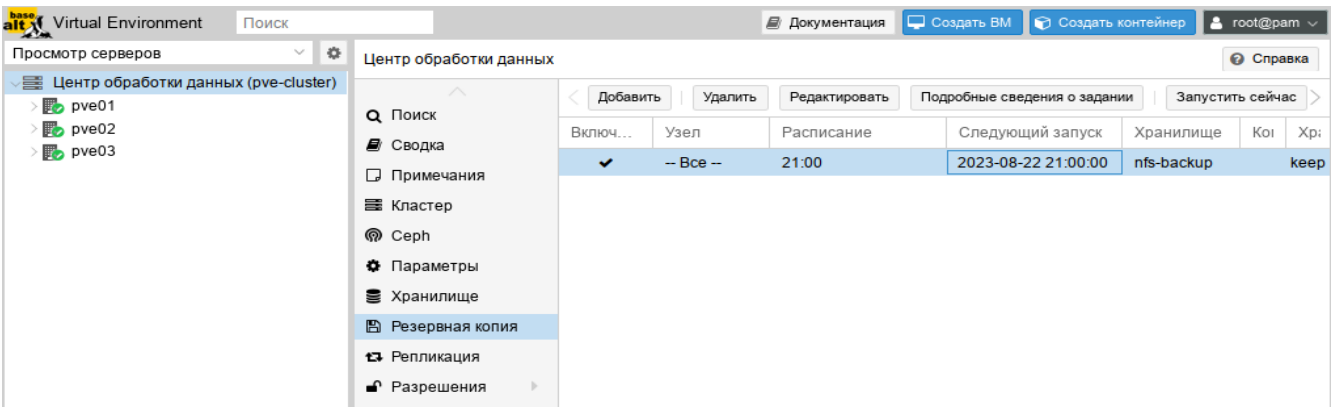

Рис. 339 – Задание резервного копирования

Для того чтобы разово создать резервную копию конкретной ВМ, достаточно выбрать ВМ, перейти в раздел «Резервная копия» и нажать на кнопку «Создать резервную копию сейчас» (рис. 340).

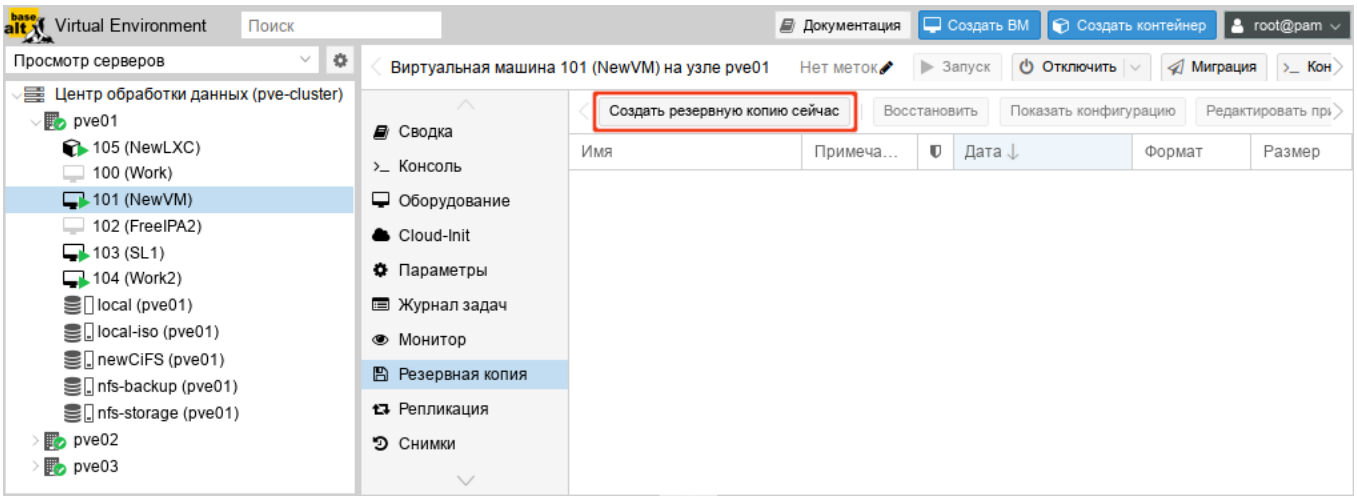

Рис. 340 – Вкладка «Резервная копия» ВМ

Далее, в открывшемся окне (рис. 341), следует указать параметры резервного копирования.

После создания резервной копии рекомендуется сразу убедиться, что из нее можно восстановить ВМ. Для этого необходимо открыть хранилище с резервной копией, выбрать резервную копию (рис. 342) и начать процесс восстановления (рис. 343). При восстановлении можно указать новое имя и переопределить некоторые параметры ВМ.

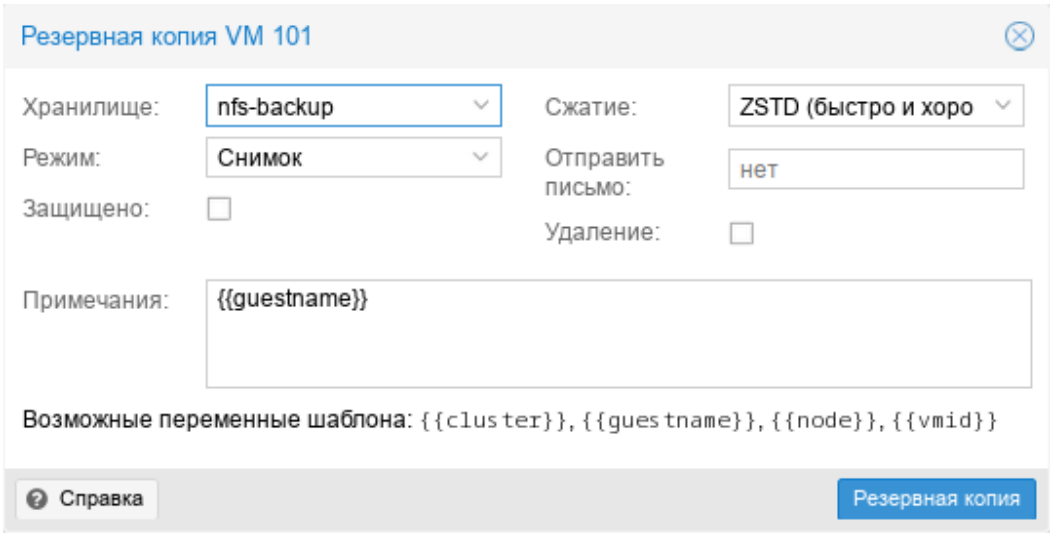

Рис. 341 – Выбор режима создания резервной копии

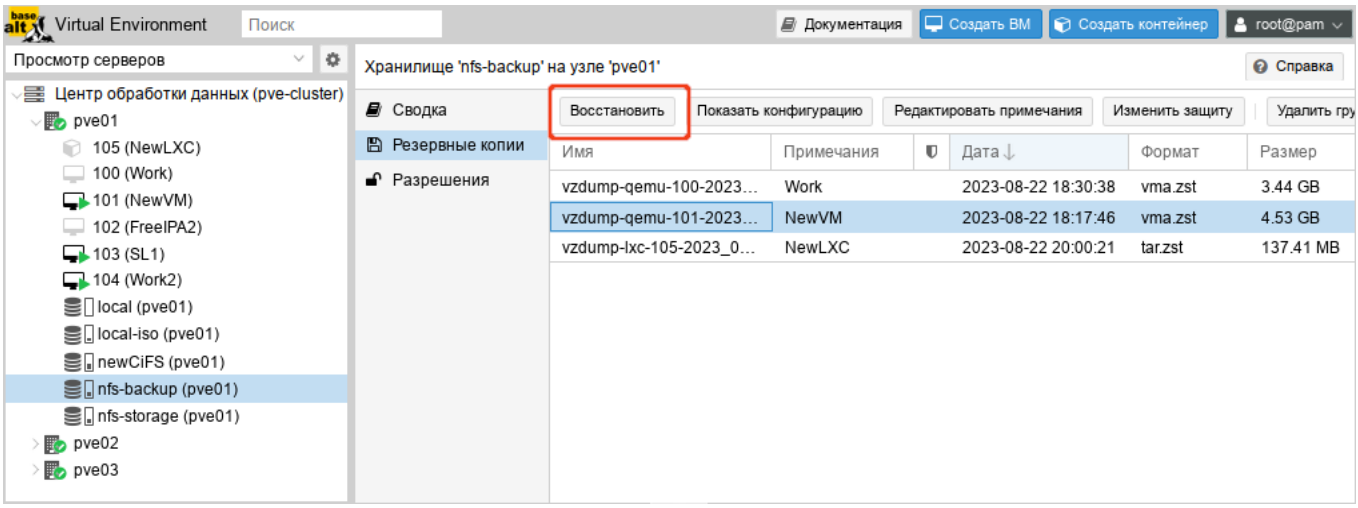

Рис. 342 – Резервная копия в хранилище nfs-backup

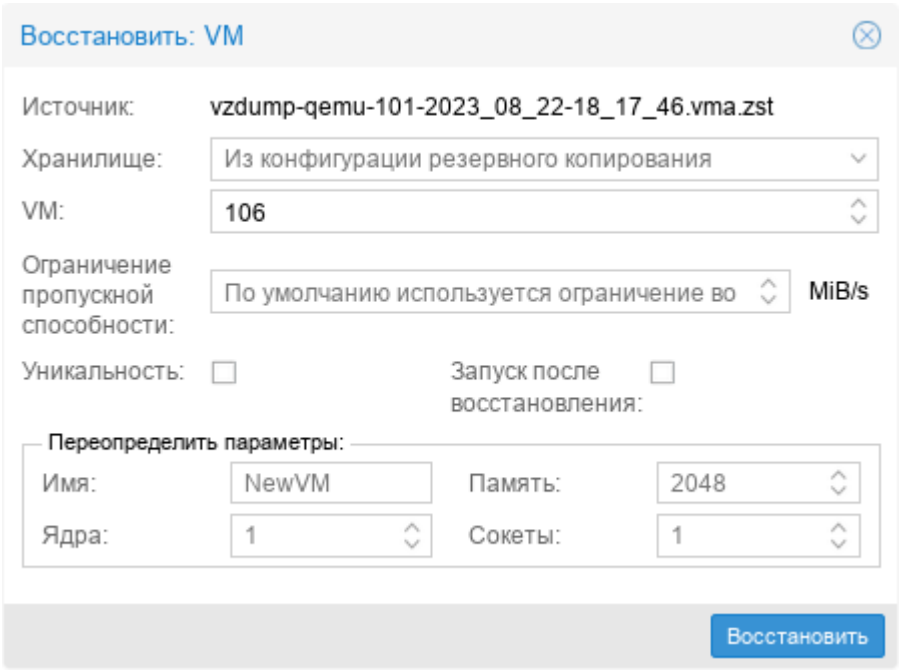

Рис. 343 – Восстановить ВМ из резервной копии

Если восстанавливать из резервной копии в интерфейсе ВМ (рис. 344), то будет предложена только замена существующей ВМ.

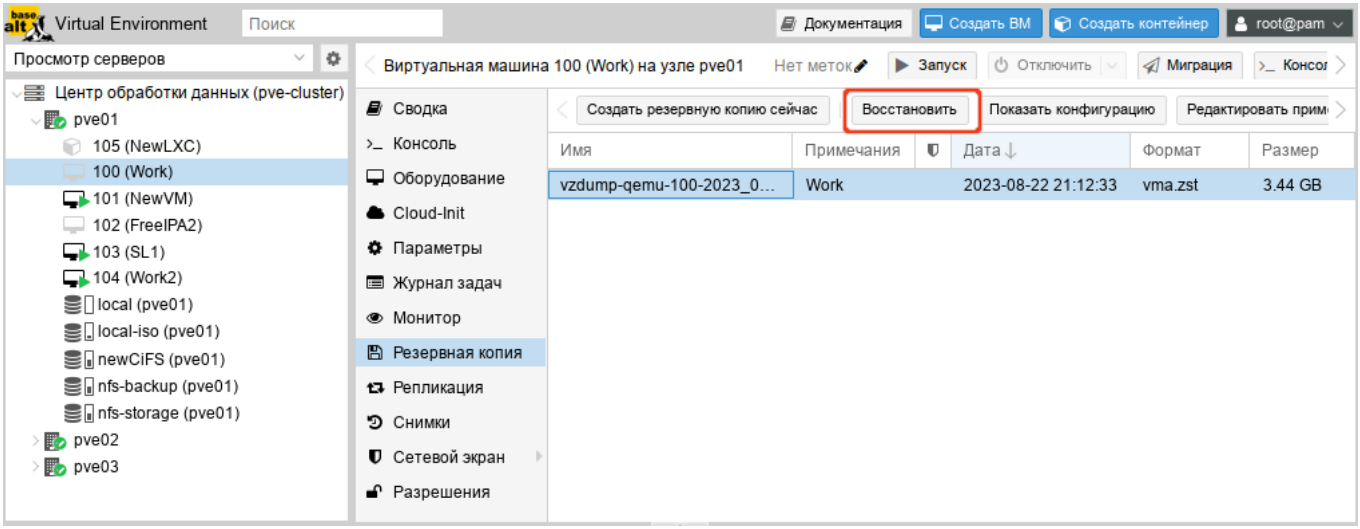

Рис. 344 – Восстановление из резервной копии в интерфейсе ВМ

Резервную копию можно пометить как защищенную (кнопка «Изменить защиту»), чтобы предотвратить ее удаление (рис. 345).

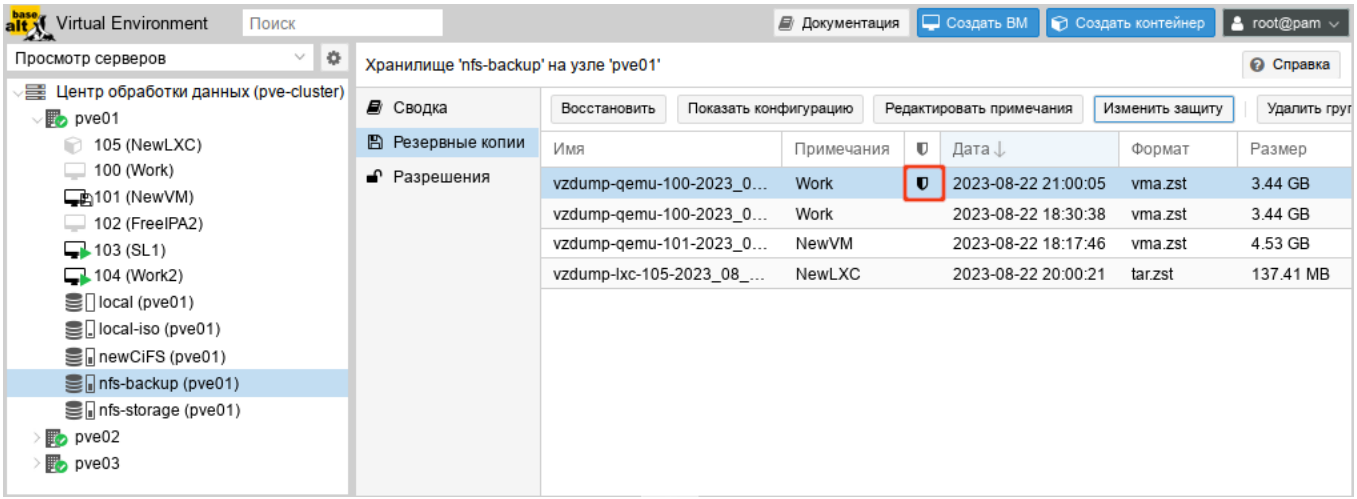

Рис. 345 – Защищенная резервная копия

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.

8.13.7. Резервное копирование из командной строки

8.13.7.1. 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.

8.13.7.2. 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.

8.13.7.3. 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 приведет 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
```

8.13.7.4. 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 таблице 24.

Пример `vzdump.conf`:

```
tmpdir: /mnt/fast_local_disk
storage: my_backup_storage
mode: snapshot
bwlimit: 10000
```

Таблица 24 – 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

Окончание таблицы 24

Опция	Описание
`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

8.13.7.5. 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.

8.13.7.6. 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
```

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 4 Гбайт корневой файловой системы:

```
# vzdump 101 --stdout | pct restore --rootfs 4 300 -
```

8.14. 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» и нажать на кнопку «Сделать снимок» (рис. 346). В открывшемся окне (рис. 347) следует ввести название снимка и нажать на кнопку «Сделать снимок».

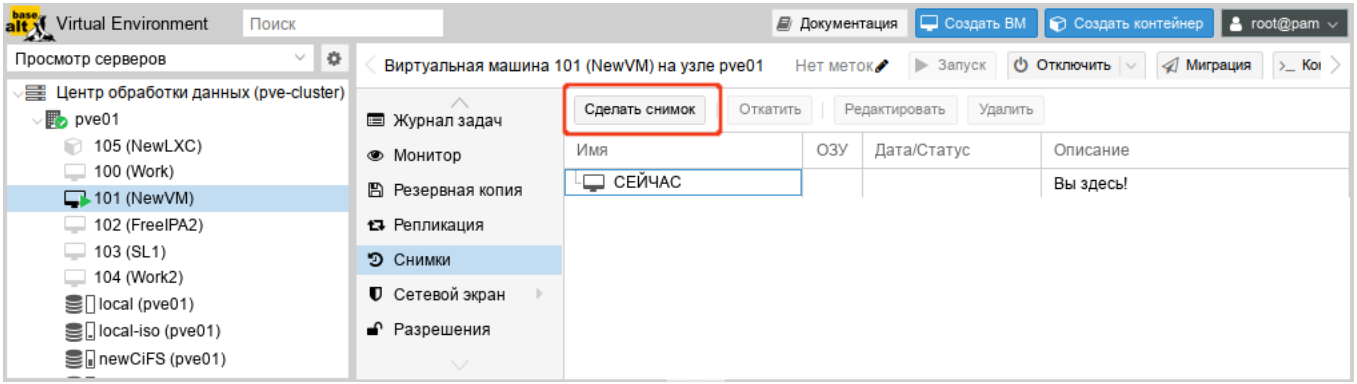

Рис. 346 – Окно управления снимками ВМ

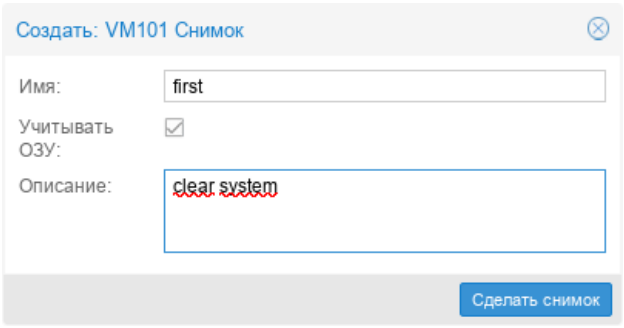

Рис. 347 – Создание снимка ВМ

Для того чтобы восстановить ВМ из снимка, необходимо в меню ВМ выбрать пункт «Снимки», выбрать снимок (рис. 348) и нажать на кнопку «Откатить».

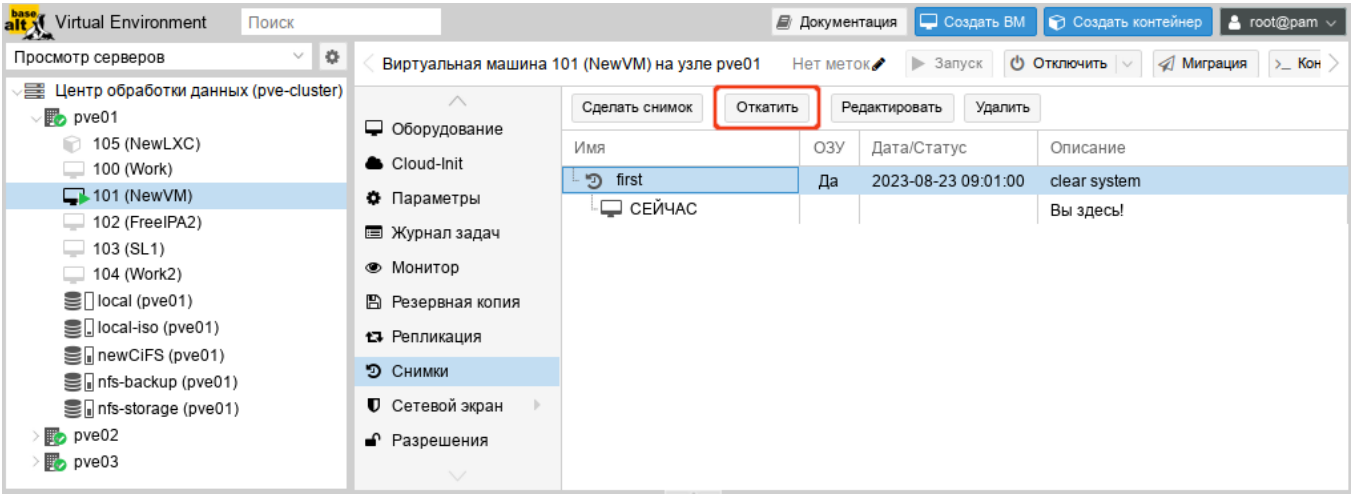

Рис. 348 – Восстановление ОС из снимка

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=7.2.10,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=7.2.10,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).

8.15. Встроенный мониторинг PVE

Все данные о потреблении ресурсов и производительности можно найти на вкладках «Сводка» узлов PVE и ВМ. Можно просматривать данные на основе почасового, ежедневного, еженедельного или за год периодов.

На рис. 349 показана «Сводка» узла pve01 со списком для выбора периода данных.

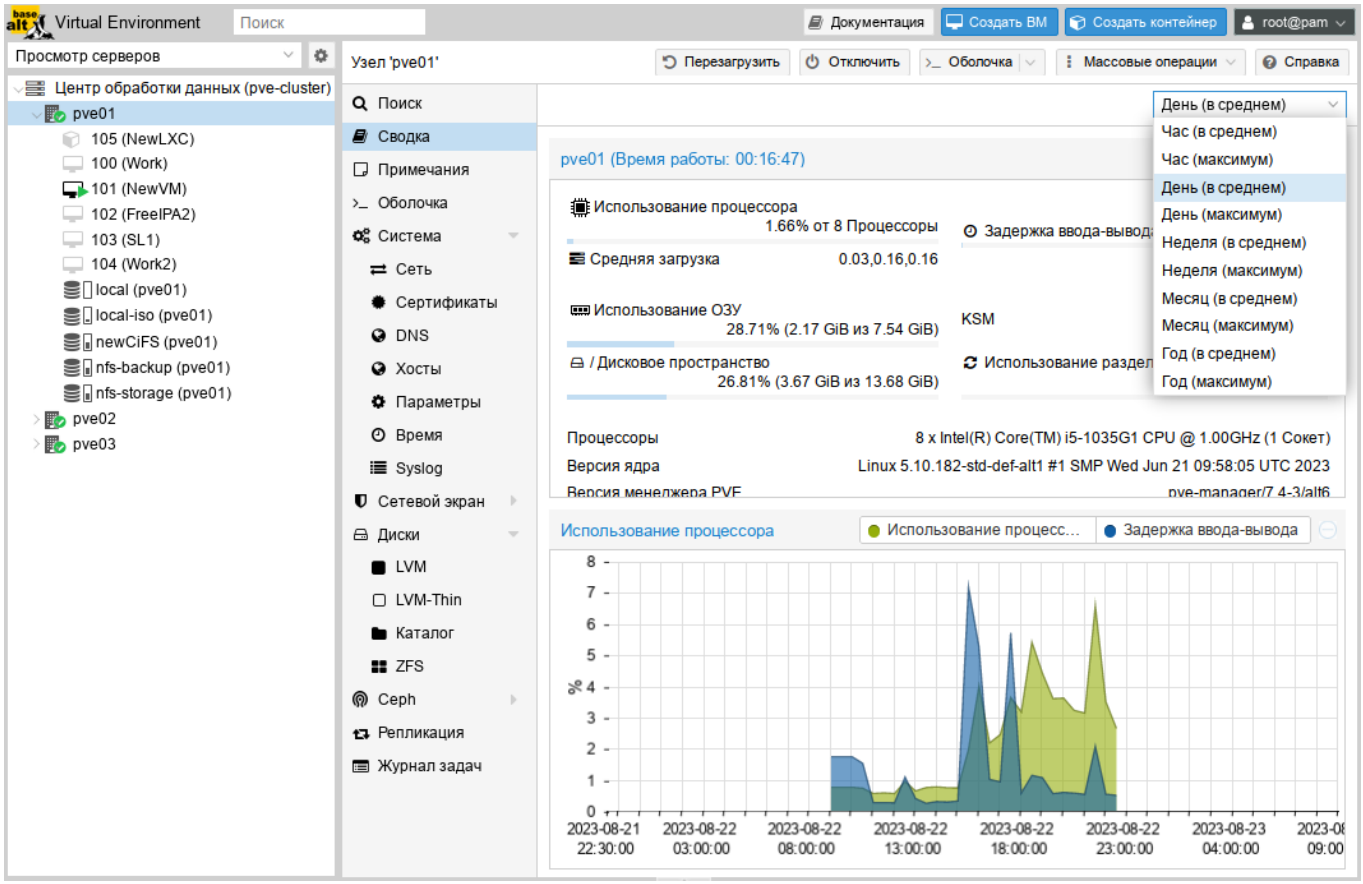

Рис. 349 – Выбор периода данных, для отображения отчета

Просмотреть список всех узлов, ВМ и контейнеров в кластере можно, выбрав «Центр обработки данных» → «Поиск» (рис. 350). Список может быть отсортирован по полям: «Тип», «Описание», «Использование диска %», «Использование

памяти %», «Использование процессора» и «Время работы». В этом списке отображается потребление ресурсов только в реальном масштабе времени.

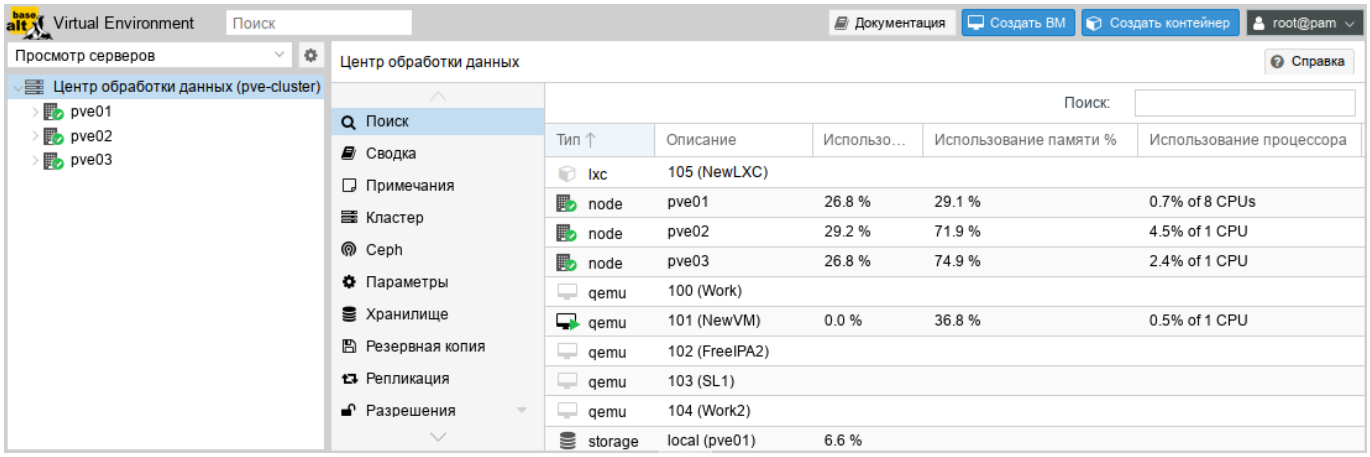

Рис. 350 – Потребление ресурсов

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. 351).

По умолчанию, `smartmontools daemon smartd` активен и включен, и сканирует диски в `/dev` 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).

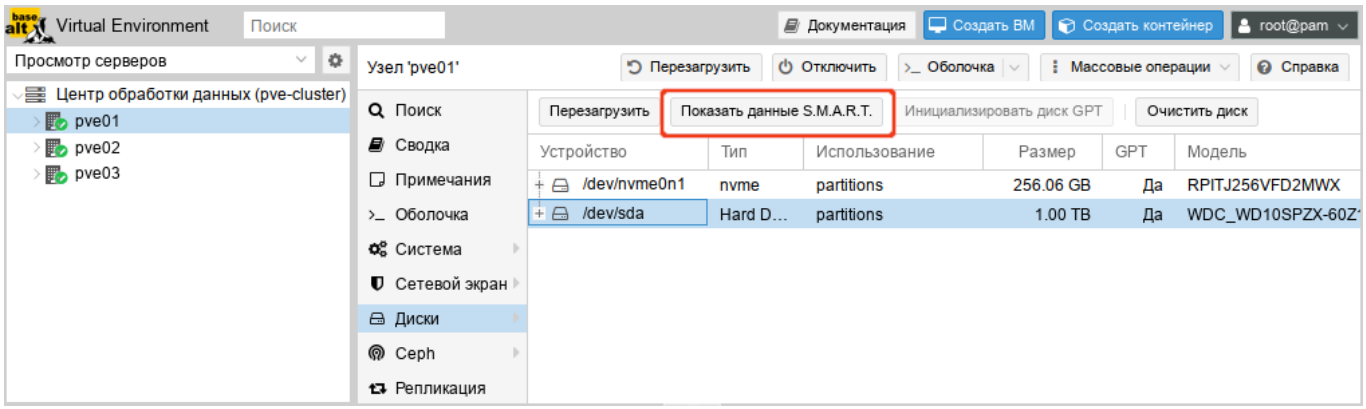

Рис. 351 – Кнопка «Показать данные S.M.A.R.T.»

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.

8.16. 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.

8.16.1. 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 выполнить ограждение (fencing) отказавшего узла.

Ограждение предотвращает службы кластера от возврата в рабочее состояние в этом месте. Затем HA перемещает данные 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.

8.16.2. 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
```

8.16.3. Настройка высокой доступности PVE

Все настройки HA PVE могут быть выполнены в веб-интерфейсе в разделе «Центр обработки данных» → «HA» (рис. 352).

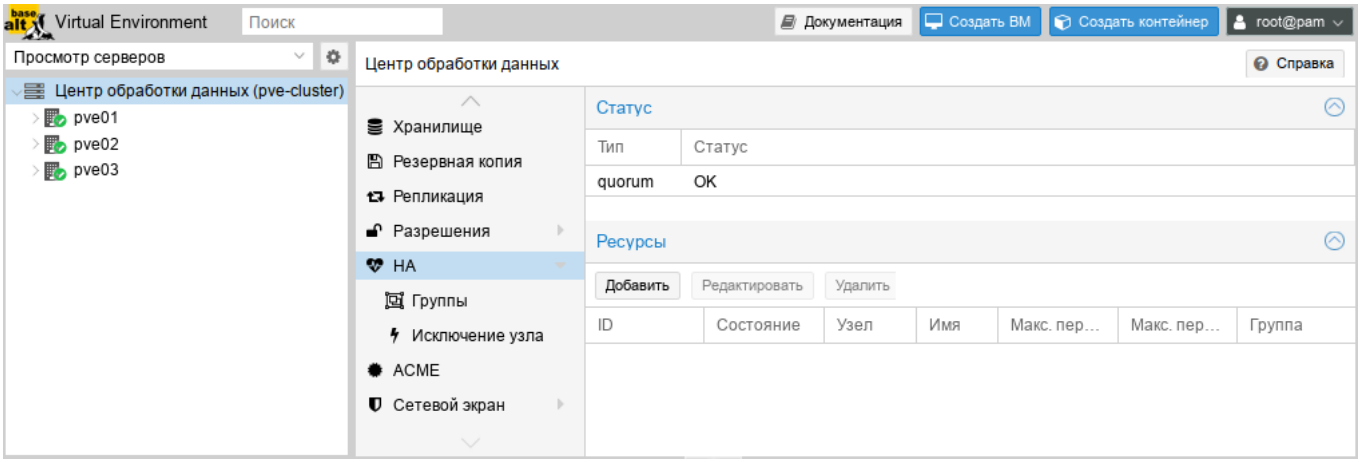

Рис. 352 – Меню HA. Статус настройки HA

8.16.3.1. 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 на кнопку «Создать» в подменю «Группы».

Элементы, доступные в блоке диалога «Группа высокой доступности» (рис. 353):

- «ID» – название HA группы;
- «Узел» – 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).

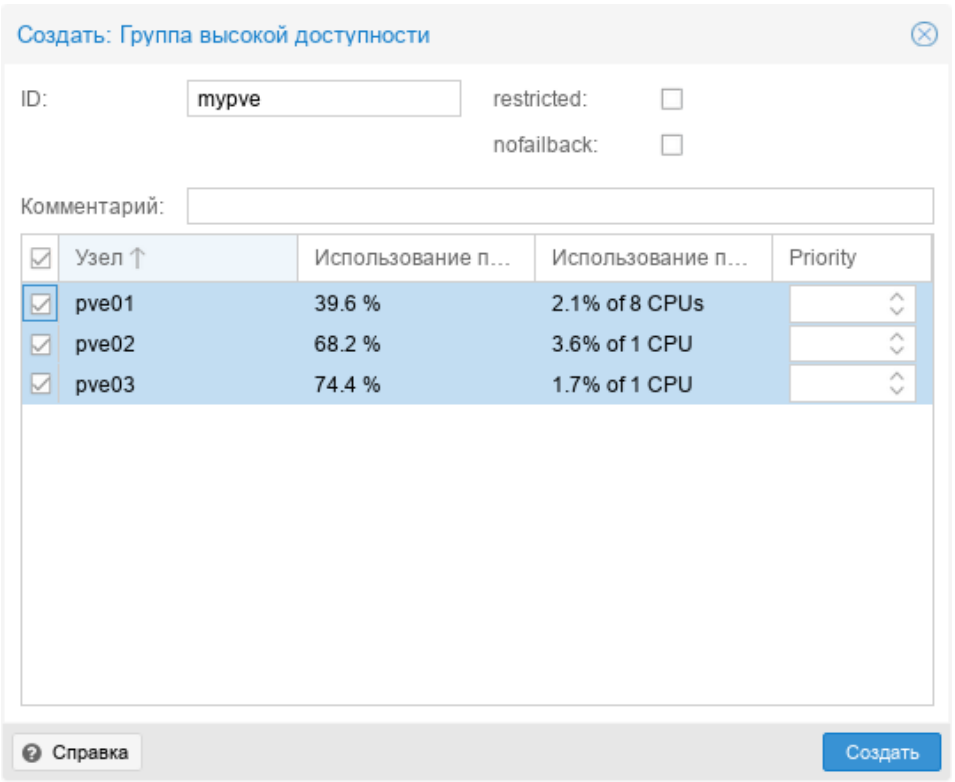

Рис. 353 – Диалог создания группы высокой доступности

На рис. 354 представлено подменю «Группы» с созданной группой.

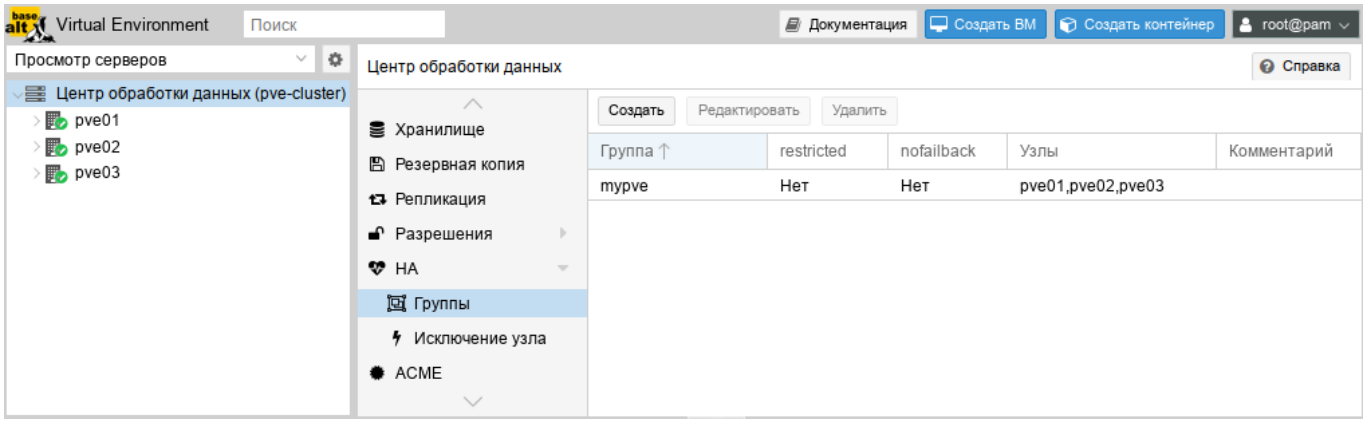

Рис. 354 – Подменю «Группы» с созданной группой

8.16.3.2. 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. 355).

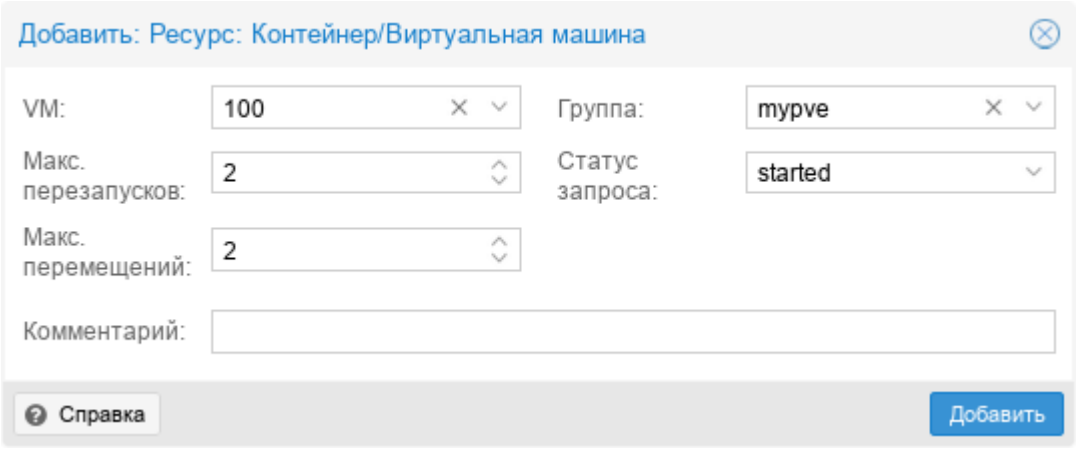

Рис. 355 – Добавление ресурса в группу

В окне можно настроить следующие параметры:

- «Макс. перезапусков» – количество попыток запуска ВМ/контейнера на новом узле после перемещения;
- «Макс. перемещений» – количество попыток перемещения ВМ/контейнера на новый узел;

- «Статус запроса» – доступны варианты:

  а) «started» – кластер менеджер будет пытаться поддерживать состояние машины в запущенном состоянии;

  б) «stopped» – при отказе узла перемещать ресурс, но не пытаться запустить; «ignored» – ресурс, который не надо перемещать при отказе узла;

  в) «disabled» – в этот статус переходят ВМ, которые находятся в состоянии «error».

На рис. 356 показана группа HA PVE и добавленные в нее ВМ и контейнеры, которыми будет управлять HA.

Раздел «Статус» отображает текущее состояние функциональности HA:

- кворум кластера установлен;
- главный узел pve01 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.

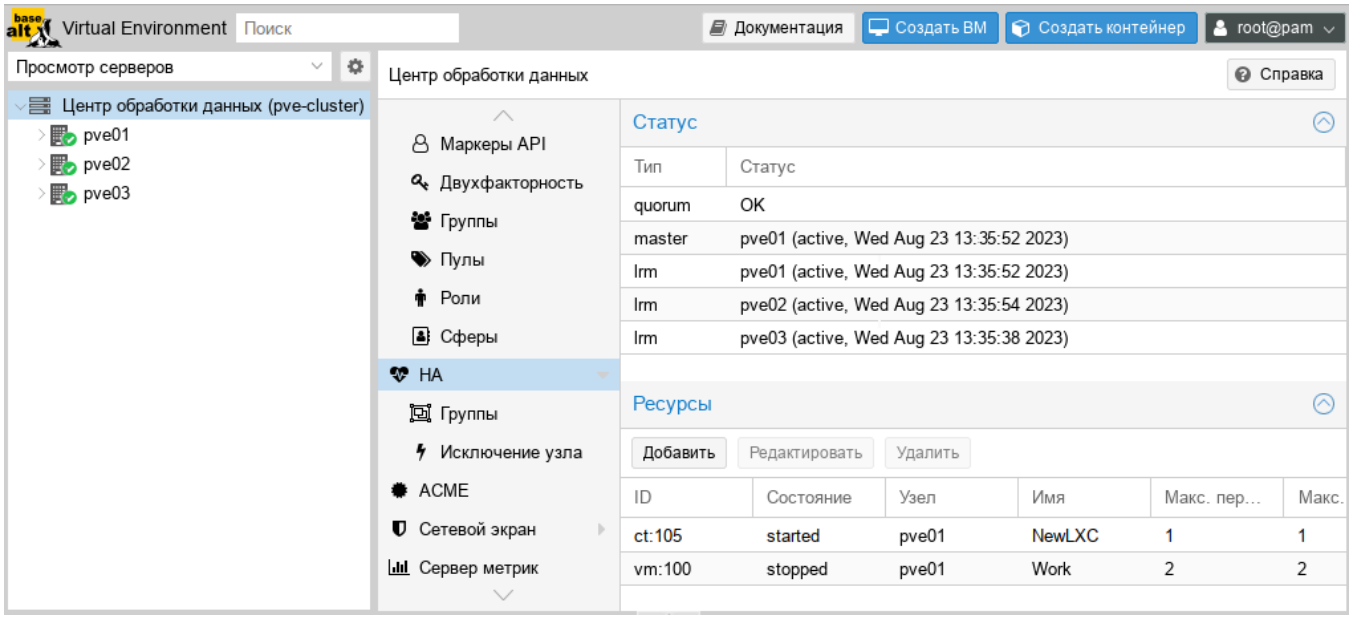

Рис. 356 – Список ресурсов

Просмотреть состояние функциональности HA можно и в консоли:

```
# ha-manager status
```

```
quorum OK
master pve01 (active, Wed Aug 23 13:26:31 2023)
lrm pve01 (active, Wed Aug 23 13:26:31 2023)
lrm pve02 (active, Wed Aug 23 13:26:33 2023)
lrm pve03 (active, Wed Aug 23 13:26:26 2023)
service ct:105 (pve01, started)
service vm:100 (pve01, stopped)
```

8.16.4. Тестирование настройки высокой доступности PVE

Для того чтобы убедиться, что HA действительно работает, можно отключить сетевое соединение для pve01 и понаблюдать за окном «Статус» (рис. 357) на предмет изменений HA.

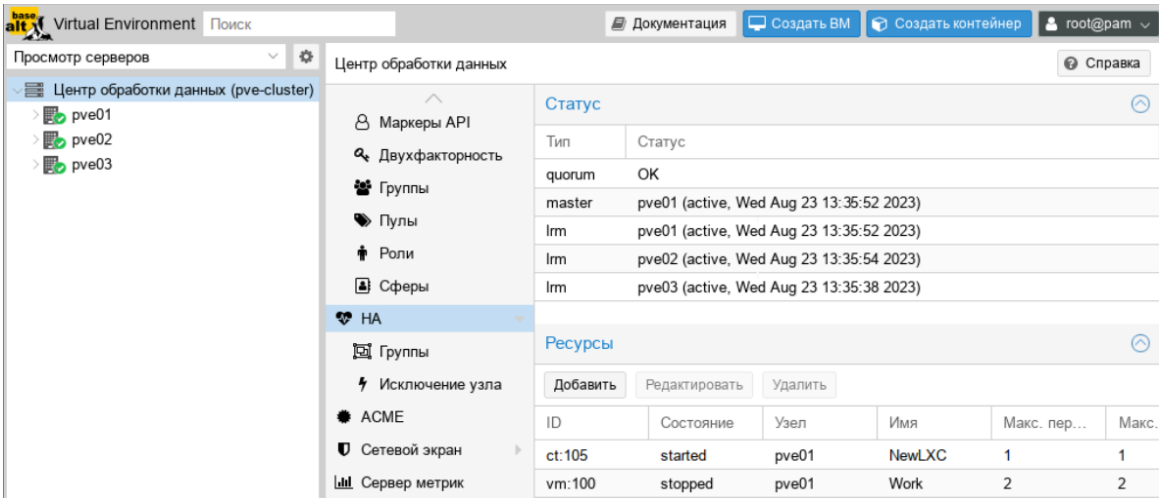

Рис. 357

После того как соединение с узлом pve01 будет потеряно, он будет помечен как недоступный. По истечению 60 секунд, HA PVE предоставит следующий доступный в группе HA узел в качестве главного (рис. 358).

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, ВМ не смогут 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.

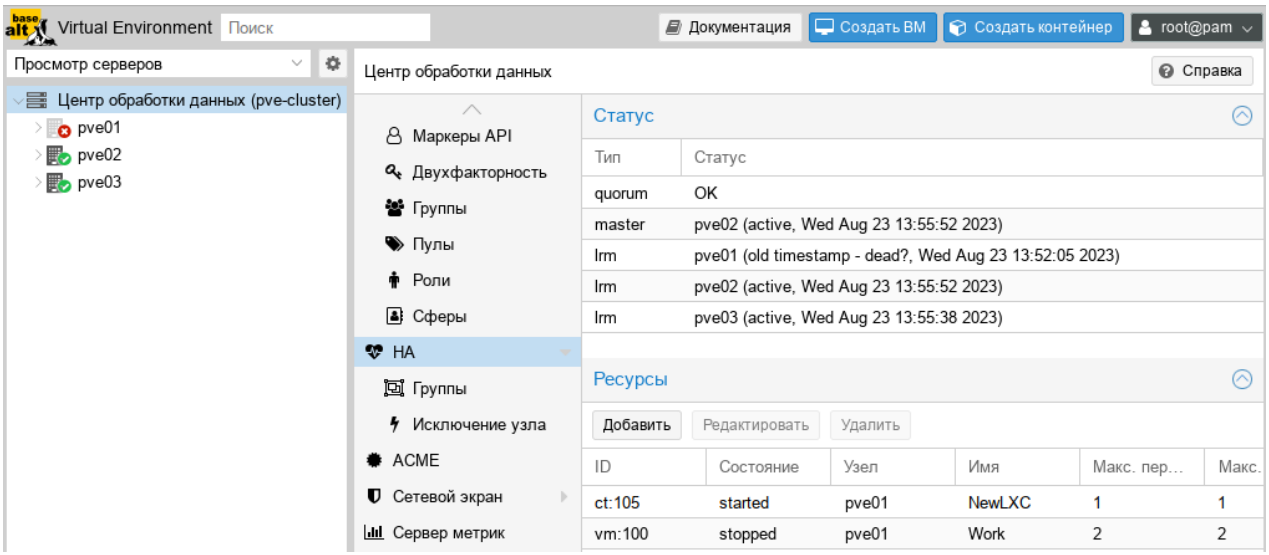

Рис. 358 – Изменение главного узла на pve02

После остановки, ВМ/контейнер перемещается на следующий свободный узел в группе HA и автоматически запускается. В данном примере контейнер 105 перемещен на узел pve02 и запущен (рис. 359).

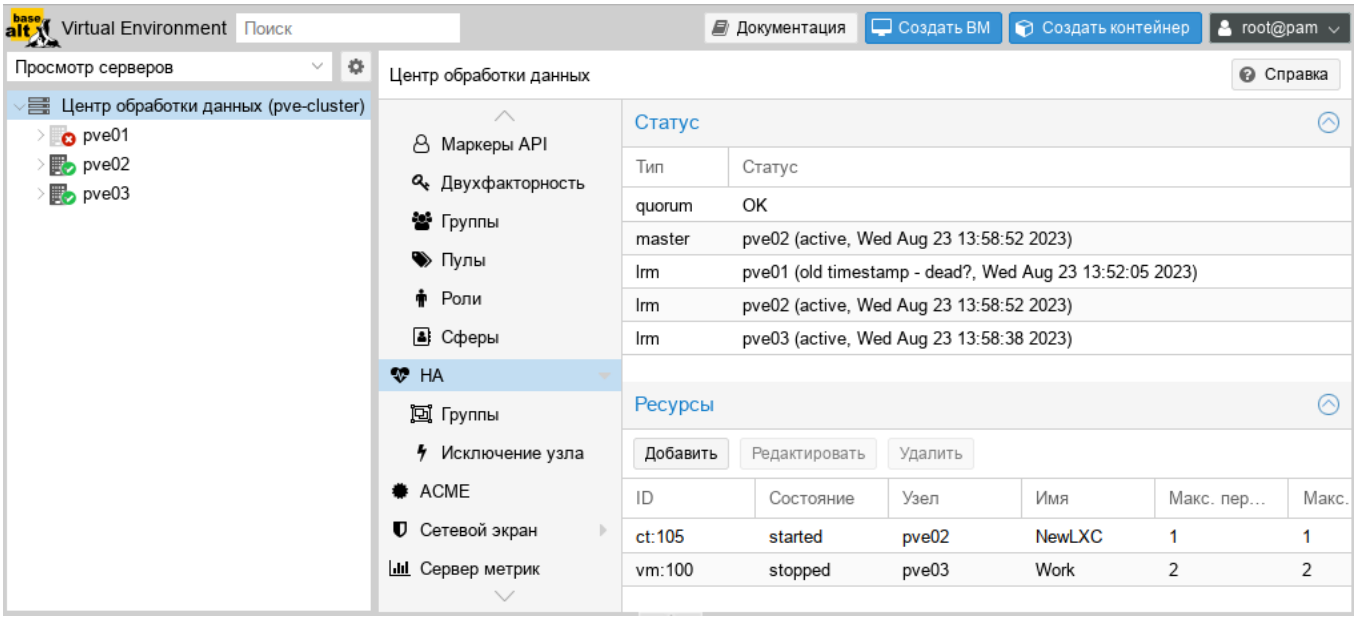

Рис. 359 – Контейнер 105 запущен на узле pve02

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.

8.17. Пользователи и их права

PVE поддерживает несколько источников аутентификации, например, Linux PAM, интегрированный сервер аутентификации PVE (рис. 360), LDAP, Active Directory и OpenID Connect.

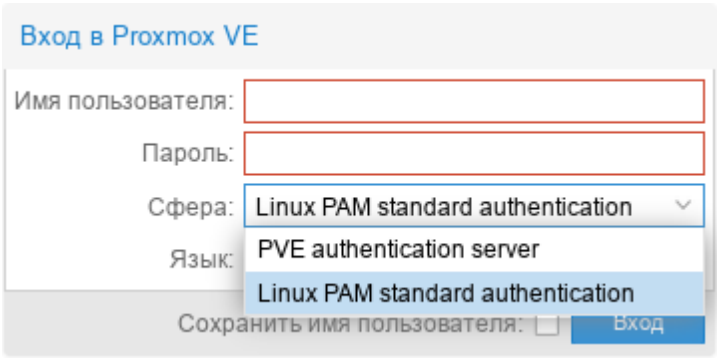

Рис. 360 – Выбор типа аутентификации в веб-интерфейсе

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::
```

Примечание. Файл `/etc/pve/storage.cfg` по умолчанию генерируется при создании пользователя.

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. Группы являются предпочтительным способом организации прав доступа. Всегда следует предоставлять права доступа группам, а не отдельным пользователям.

8.17.1. API-токены

API-токены позволяют получить доступ без сохранения состояния к REST API из другой системы. Токены могут быть сгенерированы для отдельных пользователей. Для токенов могут быть установлены отдельные разрешения и сроки действия, чтобы ограничить объем и продолжительность доступа. Если API-токен скомпрометирован, его можно отозвать, не отключая самого пользователя.

API-токены бывают двух основных типов:

- токен с раздельными привилегиями – токену необходимо предоставить явный доступ с помощью ACL. Эффективные разрешения токена вычисляются путем пересечения разрешений пользователя и токена;
- токен с полными привилегиями – разрешения токена идентичны разрешениям связанного с ним пользователя.

API-токен состоит из двух частей:

- идентификатор (Token ID), который состоит из имени пользователя, области и имени токена (`user@realm!имя токена`);
- секретное значение.

Для генерации API-токена в веб-интерфейсе необходимо в окне «Центр обработки данных» → «Разрешения» → «Маркеры API» нажать на кнопку «Добавить». В открывшемся окне следует выбрать пользователя и указать ID-токена (рис. 368).

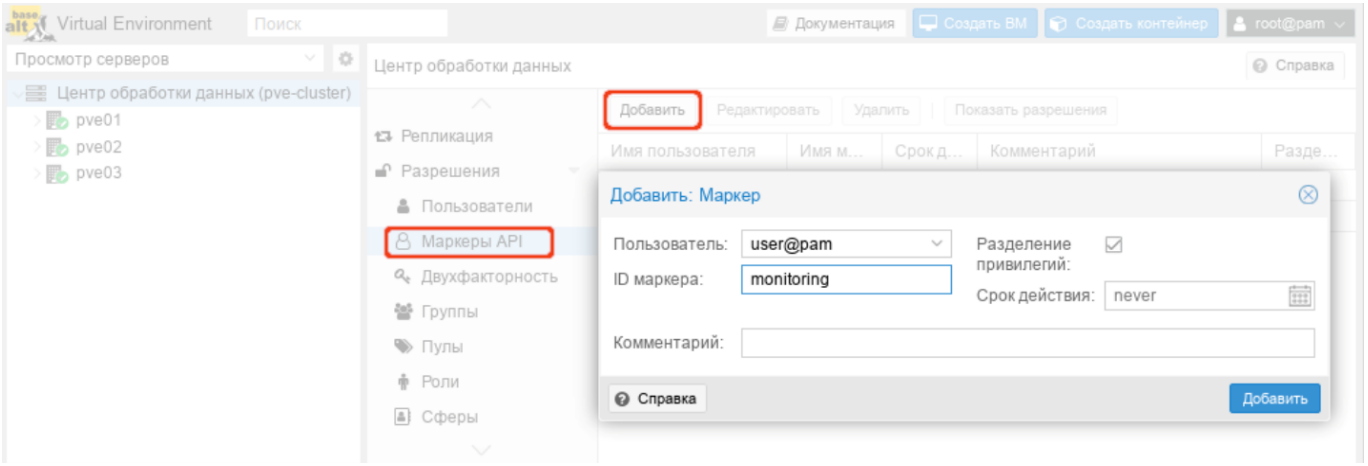

Рис. 361

Примечание. Опция «Разделение привилегий» должна быть отключена, в противном случае токену необходимо назначить явные права. Подробнее см. п. 8.17.5.

После нажатия кнопки «Добавить» будет сгенерирован API-токен (рис. 362).

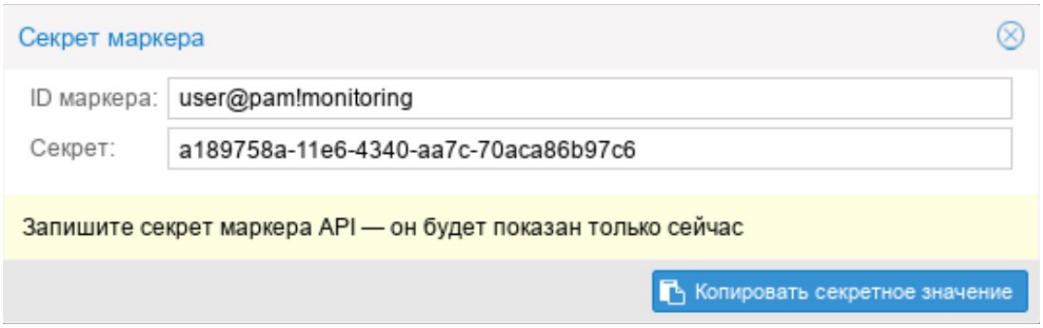

Рис. 362

Отображаемое секретное значение необходимо сохранить.

ВНИМАНИЕ!

Значение токена отображается/возвращается только один раз при создании токена. Его нельзя будет снова получить через API позже!

Если был создан токен с раздельными привилегиями, токену необходимо предоставить разрешения:

1) в окне «Центр обработки данных» → «Разрешения» нажать на кнопку «Добавить» → «Разрешения маркера API» (рис. 363);

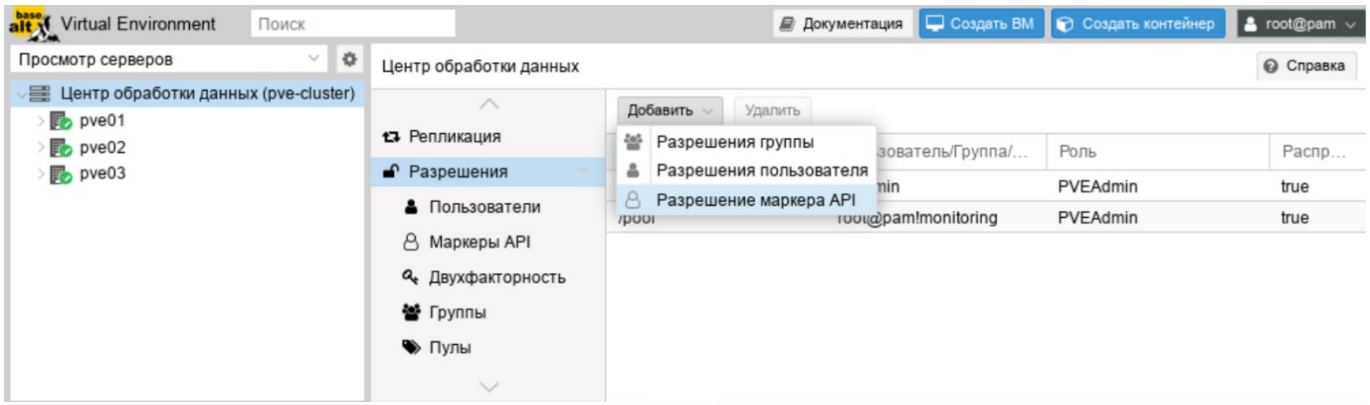

Рис. 363

2) в открывшемся окне выбрать путь, токен и роль и нажать на кнопку «Добавить» (рис. 364).

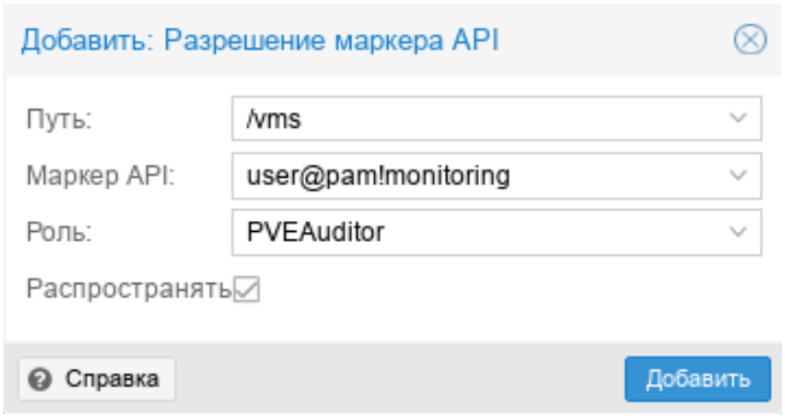

Рис. 364

Для создания API-токена в консоли используется команда:

```
# pveum user token add <userid> <tokenid> [ОПЦИИ]
```

Возможные опции:

- `--comment <строка>` – комментарий к токену;
- `--expire <целое число>` – дата истечения срока действия API-токена в секундах с начала эпохи (по умолчанию срок действия API-токена совпадает со сроком действия пользователя). Значение 0 указывает, что срок действия токена не ограничен;
- `--privsep <логическое значение>` – ограничить привилегии API-токена с помощью отдельных списков контроля доступа (по умолчанию) или

предоставить полные привилегии соответствующего пользователя (значение 0).

Примеры команд для работы с токенами:

- создать токен t2 для пользователя user@pam с полными привилегиями (рис. 365):

```
# pveum user token add user@pam t2 --privsep 0
```

key	value
full-tokenid	user@pam!t2
info	{"privsep":"0"}
value	3c749375-e189-493d-8037-a1179317c406

Рис. 365

- вывести список токенов пользователя (рис. 366):

```
# pveum user token list user@pam
```

tokenid	comment	expire	privsep
monitoring		0	1
t2		0	0

Рис. 366

- вывести эффективные разрешения для токена:

```
# pveum user token permissions user@pam t2
```

Можно использовать опцию `--path`, чтобы вывести разрешения для этого пути, а не все дерево:

```
# pveum user token permissions user@pam t2 --path /storage
```

- добавить разрешения для токена с раздельными привилегиями:

```
# pveum acl modify /vms --tokens 'user@pam!monitoring' --roles
PVEAdmin,PVEAuditor
```

- удалить токен пользователя:

```
# pveum user token remove user@pam t2
```

П р и м е ч а н и е . Разрешения на API-токены всегда являются подмножеством разрешений соответствующего пользователя. То есть API-токен не может использоваться для выполнения задачи, на которую у пользователя владельца токена нет разрешения.

Пример:

- предоставить пользователю `test@pve` роль `PVEVMAdmin` на всех ВМ:

```
# pveum acl modify /vms --users test@pve --roles PVEVMAdmin
```

- создать API-токен с раздельными привилегиями с правами только на просмотр информации о ВМ:

```
# pveum user token add test@pve monitoring --privsep 1
# pveum acl modify /vms --tokens 'test@pve!monitoring' --roles PVEAuditor
```

- проверить разрешения пользователя и токена:

```
# pveum user permissions test@pve
# pveum user token permissions test@pve monitoring
```

Чтобы использовать API-токен при выполнении API-запросов, следует установить заголовок HTTP Authorization в значение `PVEAPIToken=USER@REALM!TOKENID=UUID`.

8.17.2. Пулы ресурсов

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 вместе с группами для предоставления доступа их членам к определенным машинам и хранилищам.

Пример создания пула ресурсов в веб-интерфейсе:

1) в окне «Центр обработки данных» → «Разрешения» → «Пулы» нажать на кнопку «Создать». В открывшемся окне указать название пула и нажать на кнопку «ОК» (рис. 367);

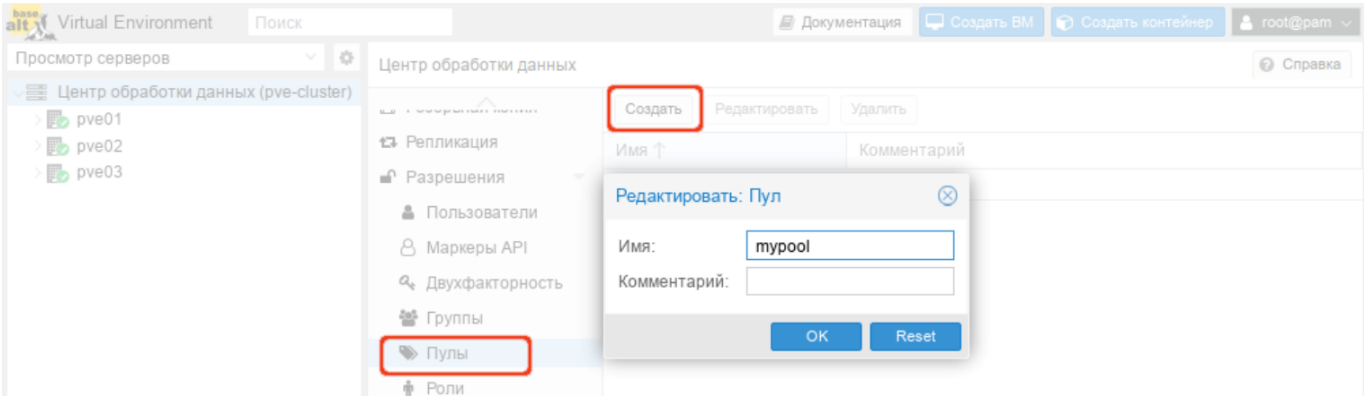

Рис. 367

2) добавить в пул ВМ. Для этого выбрать пул («Пул» → «Члены»), нажать на кнопку «Добавить» → «Виртуальная машина», выбрать ВМ и нажать на кнопку «Добавить» (рис. 368);

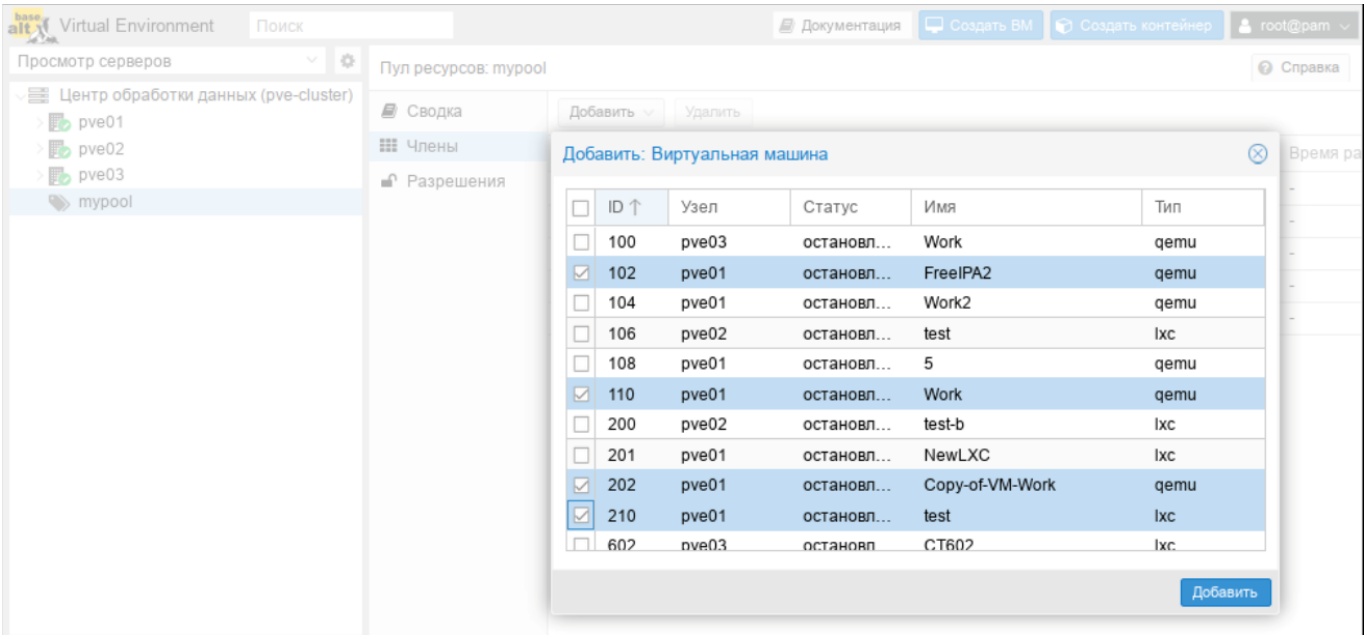

Рис. 368

3) добавить в пул хранилища. Для этого выбрать пул («Пул» → «Члены»), нажать на кнопку «Добавить» → «Хранилище», выбрать хранилище и нажать на кнопку «Добавить» (рис. 369).

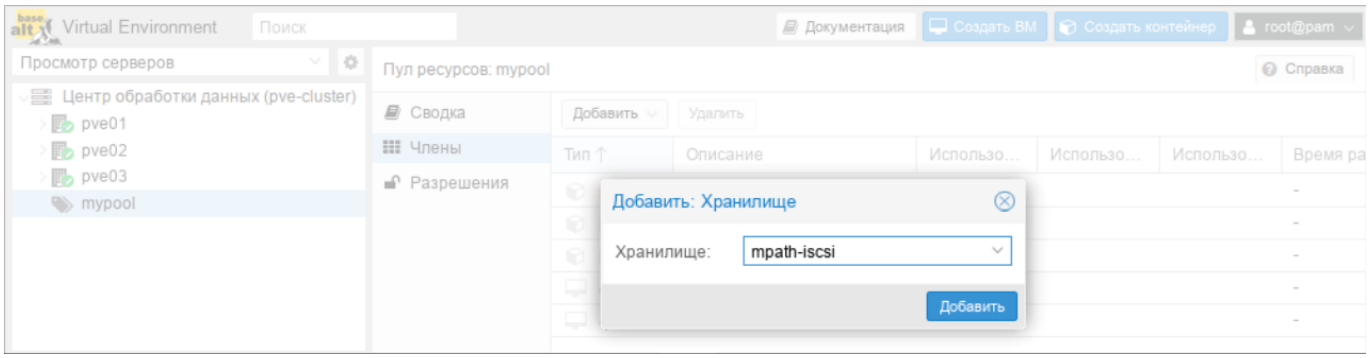

Рис. 369

Работа с пулами ресурсов в командной строке:

- создать пул:

```
# pveum pool add IT --comment 'IT development pool'
```

- вывести список пулов:

```
# pveum pool list
```

poolid
IT
mypool

- добавить ВМ и хранилища в пул:

```
# pveum pool modify IT --vms 201,108,202,104,208 --storage mpath2,nfs-storage
```

- удалить ВМ из пула:

```
# pveum pool modify IT --delete 1 --vms 108,104
```

- удалить пул:

```
# pveum pool delete IT
```

П р и м е ч а н и е . Можно удалить только пустой пул.

8.17.3. Области аутентификации

Доступны следующие сферы (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 средних установок PVE, где пользователям не требуется доступ к другим ресурсам;

- «Сервер LDAP» – позволяет использовать внешний LDAP-сервер для аутентификации пользователей (например, OpenLDAP);
- «Сервер Active Directory» – позволяет аутентифицировать пользователей через AD. Поддерживает LDAP в качестве протокола аутентификации;
- «Сервер OpenID Connect» – уровень идентификации поверх протокола OATH 2.0. Позволяет аутентифицировать пользователей на основе аутентификации, выполняемой внешним сервером авторизации.

Настройки сферы аутентификации хранятся в файле `/etc/pve/domains.cfg`.

8.17.3.1. Стандартная аутентификация Linux PAM

При использовании «Стандартная аутентификация 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 («Требовать двухфакторную проверку подлинности») и установить ее в качестве области по умолчанию для входа в систему («По умолчанию») (рис. 370).

Для добавления нового пользователя, необходимо в окне «Центр обработки данных» → «Разрешения» → «Пользователи» нажать на кнопку «Добавить». На рис. 371 показано создание нового пользователя с использованием PAM аутентификации (системный пользователь user должен существовать, в качестве пароля будет использоваться пароль для входа в систему).

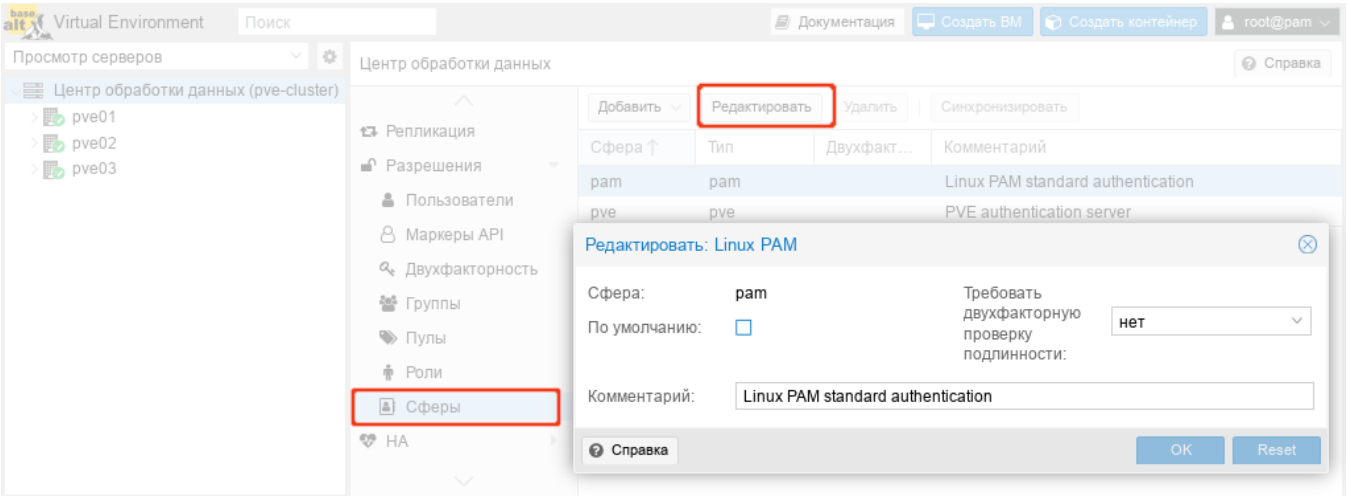

Рис. 370 – Конфигурация PAM аутентификации

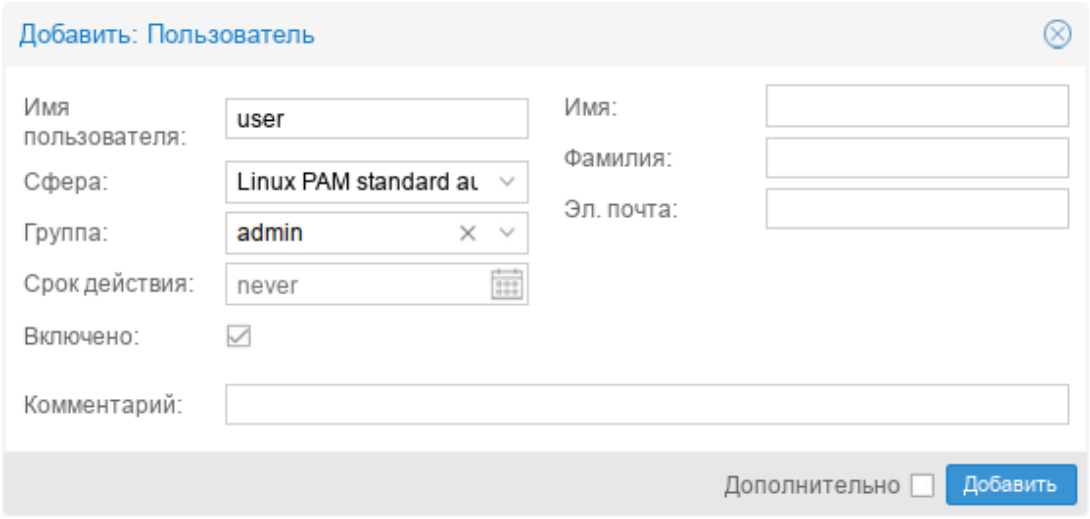

Рис. 371 – Создание нового пользователя с использованием PAM аутентификации

8.17.3.2. Сервер аутентификации PVE

Область «Сервер 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. Администратор может включить двухфакторную аутентификацию для пользователей данной области («Требовать двухфакторную проверку подлинности») и установить ее в качестве области по умолчанию для входа в систему («По умолчанию») (рис. 372).

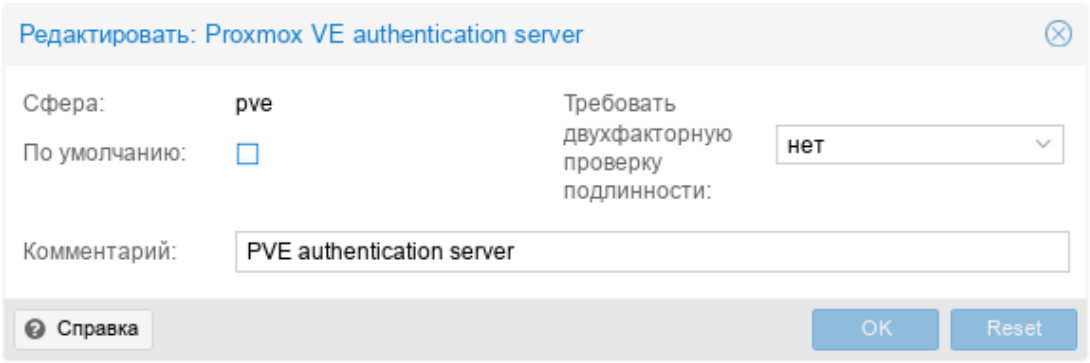

Рис. 372 – Конфигурация PVE аутентификации

Для добавления нового пользователя, необходимо в окне «Центр обработки данных» → «Разрешения» → «Пользователи» нажать на кнопку «Добавить». На рис. 373 показано создание нового пользователя с использованием PVE аутентификации.

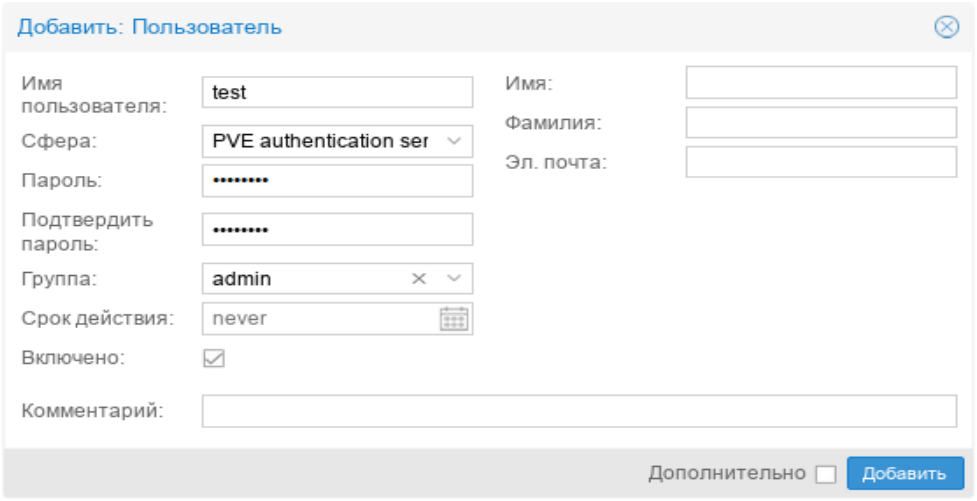

Рис. 373 – Создание нового пользователя с использованием PVE аутентификации

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"
```

8.17.3.3. LDAP аутентификация

В данном разделе приведен 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 – учетная запись с правами чтения LDAP;
- pve – группа, пользователи которой имеют право аутентифицироваться в PVE.

Для настройки аутентификации LDAP необходимо выполнить следующие шаги:

1) создать область аутентификации LDAP. Для этого в разделе «Центр обработки данных» → «Разрешения» → «Сферы» нажать на кнопку «Добавить» → «Сервер LDAP» (рис. 374);
2) на вкладке «Общее» (рис. 375) указать следующие данные:
   - «Сфера» – 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;
- «Требовать двухфакторную проверку подлинности» – включить двухфакторную аутентификацию;

3) на вкладке «Параметры синхронизации» (рис. 376) заполнить параметры синхронизации (в скобках указаны значения, используемые в данном примере):
   - «Пользователь (bind)» – имя пользователя (`uid=admin,cn=users,cn=accounts,dc=example,dc=test`);
   - «Пароль (bind)» – пароль пользователя;
   - «Атрибут электронной почты» (опционально);
   - «Аттр. имени группы» – атрибут имени группы (`cn`);
   - «Классы пользователей» – класс пользователей LDAP (`person`);
   - «Классы групп» – класс групп LDAP (`posixGroup`);
   - «Фильтр пользователей» – фильтр пользователей (`memberOf=cn=pve,cn=groups,cn=accounts,dc=example,dc=test`);
   - «Фильтр групп» – фильтр групп (`(|(cn=*pve*)(dc=ipa)(dc=example)(dc=test))`);
4) нажать на кнопку «Добавить»;
5) выбрать добавленную область и нажать на кнопку «Синхронизировать» (рис. 377);
6) указать, если необходимо, параметры синхронизации и нажать на кнопку «Синхронизировать» (рис. 378).

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».

Настроить разрешения для группы/пользователя на вкладке можно на вкладке «Разрешения».

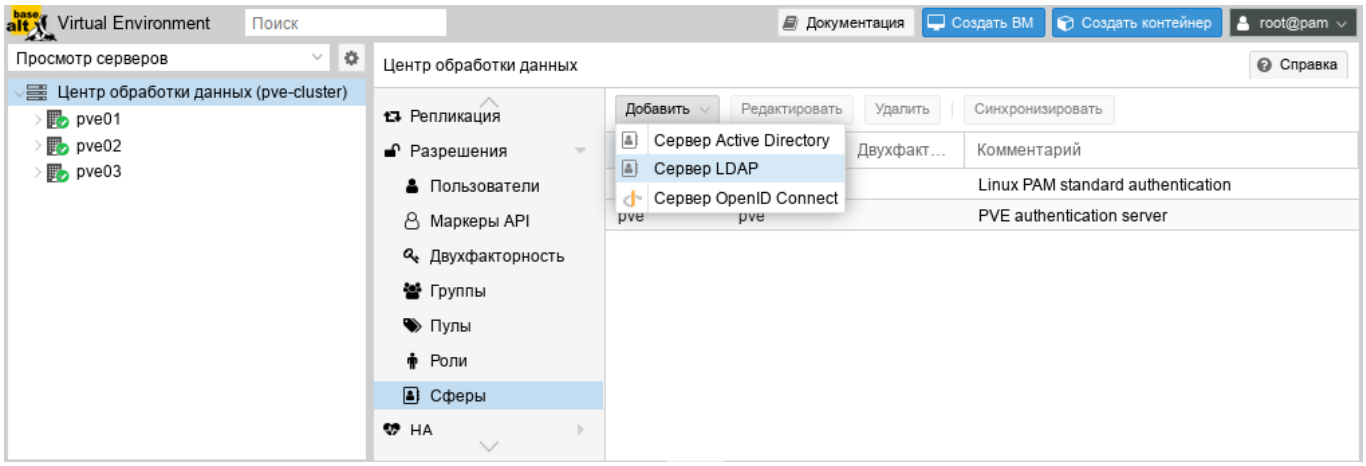

Рис. 374 – Создать область аутентификации LDAP

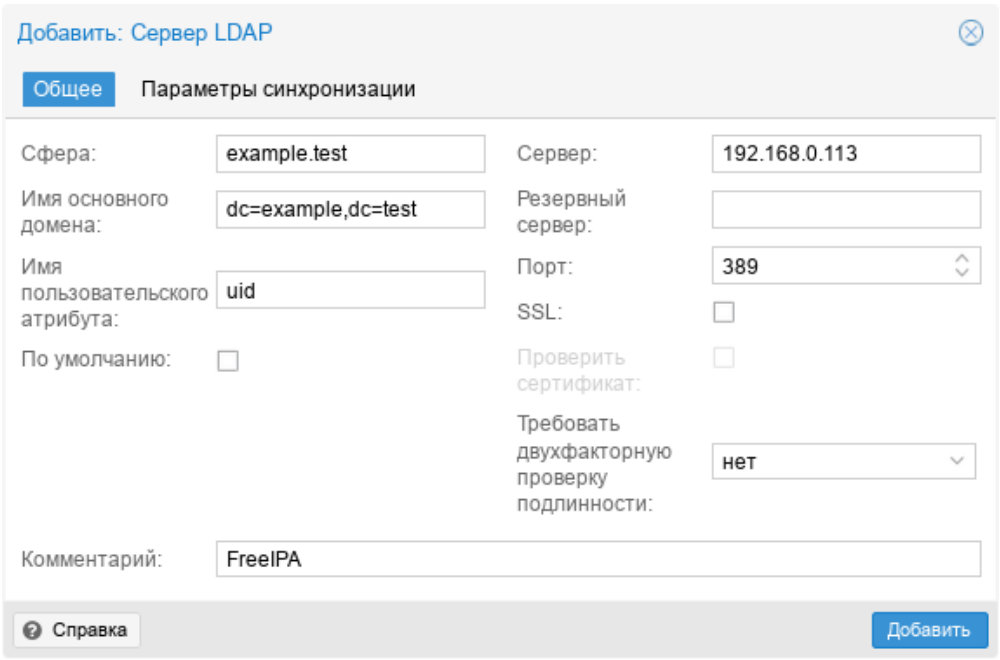

Рис. 375 – Настройка LDAP аутентификации (вкладка «Общее»)

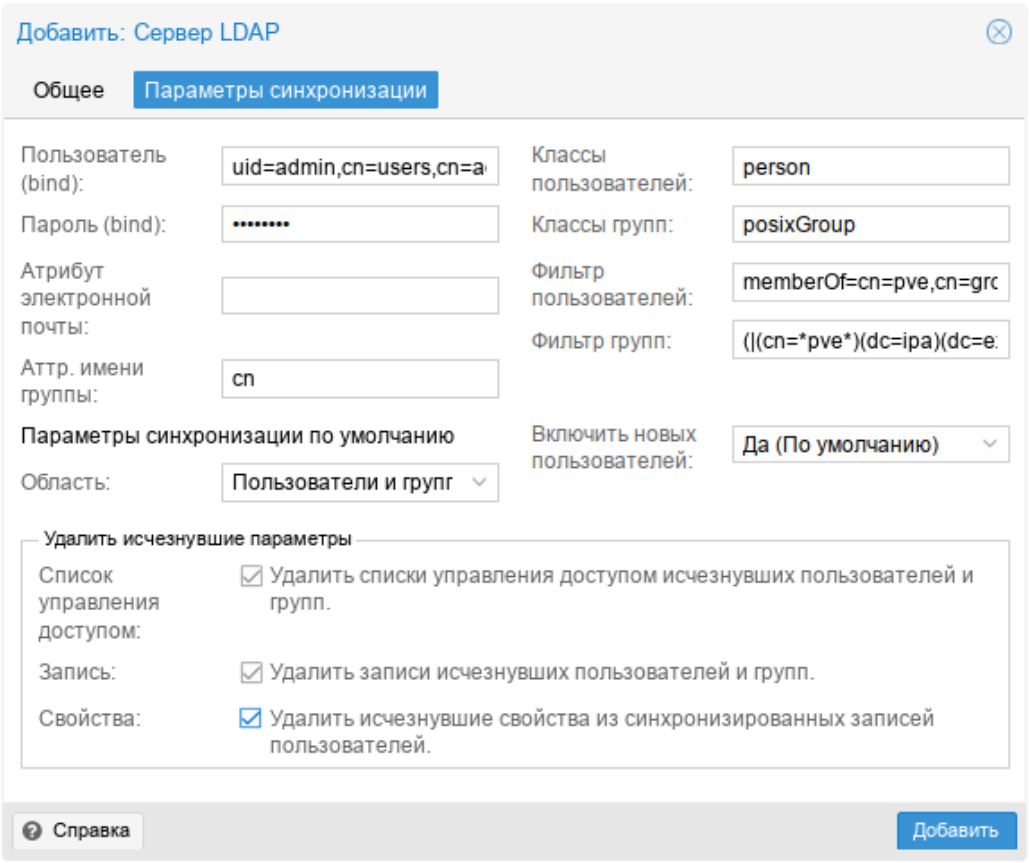

Рис. 376 – Настройка LDAP аутентификации (вкладка «Параметры синхронизации»)

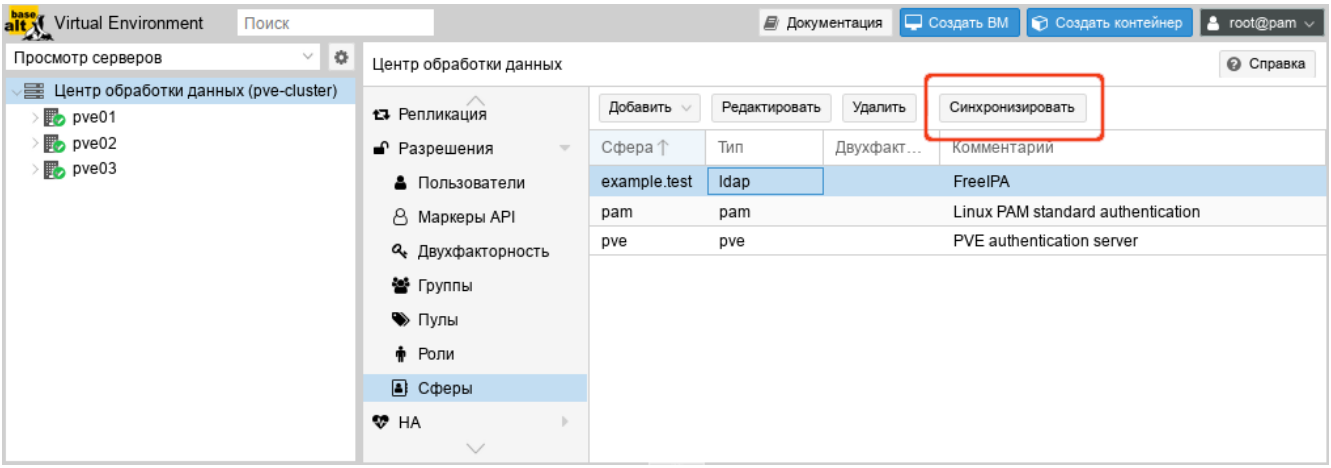

Рис. 377 – Кнопка «Синхронизировать»

Примечание. Команда синхронизации пользователей и групп:

```
# pveum realm sync example.test
```

Для автоматической синхронизации пользователей и групп можно добавить команду синхронизации в планировщик задач.

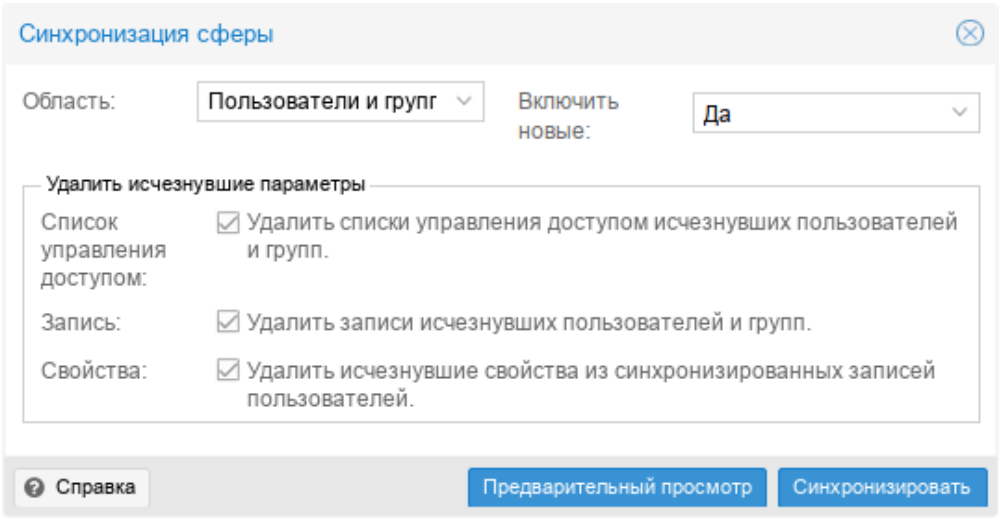

Рис. 378 – Параметры синхронизации области аутентификации

8.17.3.4. AD аутентификация

В данном разделе приведен пример настройки аутентификации на сервере AD. В примере используются следующие исходные данные:

- dc.test.alt, 192.168.0.122 – сервер AD;
- administrator@test.alt – учетная запись администратора (для большей безопасности рекомендуется создать отдельную учетную запись с доступом только для чтения к объектам домена и не использовать учетную 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 «Центр обработки данных» → «Разрешения» → «Сферы» нажать на кнопку «Добавить» → «Сервер Active Directory» (см. рис. 374);
2) на вкладке «Общее» (рис. 379) указать следующие данные:
   - «Сфера» – 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;
- «Требовать двухфакторную проверку подлинности» – включить двухфакторную аутентификацию;

3) на вкладке «Параметры синхронизации» (рис. 380) заполнить параметры синхронизации (в скобках указаны значения, используемые в данном примере):
   - «Пользователь (bind)» – имя пользователя (`cn=Administrator,cn=Users,dc=test,dc=alt`);
   - «Пароль (bind)» – пароль пользователя;
   - «Атрибут электронной почты» (опционально);
   - «Аттр. имени группы» – атрибут имени группы (`cn`);
   - «Классы пользователей» – класс пользователей LDAP;
   - «Классы групп» – класс групп LDAP;
   - «Фильтр пользователей » – фильтр пользователей (`((&(objectclass=user)(samaccountname=*)(MemberOf=CN=office,ou=OU,dc=TEST,dc=ALT))`);
   - «Фильтр групп» – фильтр групп (`((|(cn=*office*)(dc=dc)(dc=test)(dc=alt))`);
4) нажать на кнопку «Добавить»;
5) выбрать добавленную область и нажать на кнопку «Синхронизировать»;
6) указать, если необходимо, параметры синхронизации и нажать на кнопку «Синхронизировать» («Sync») (см. рис. 378). 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».

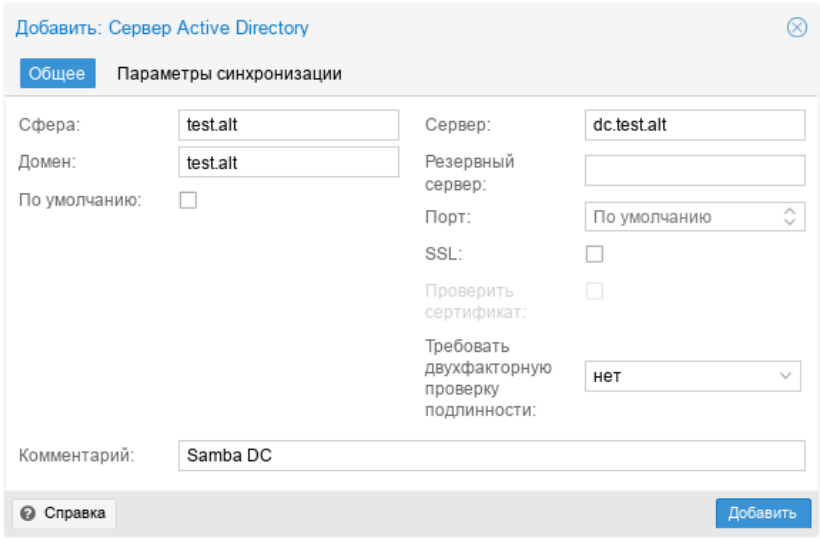

Рис. 379 – Настройка AD аутентификации (вкладка «Общее»)

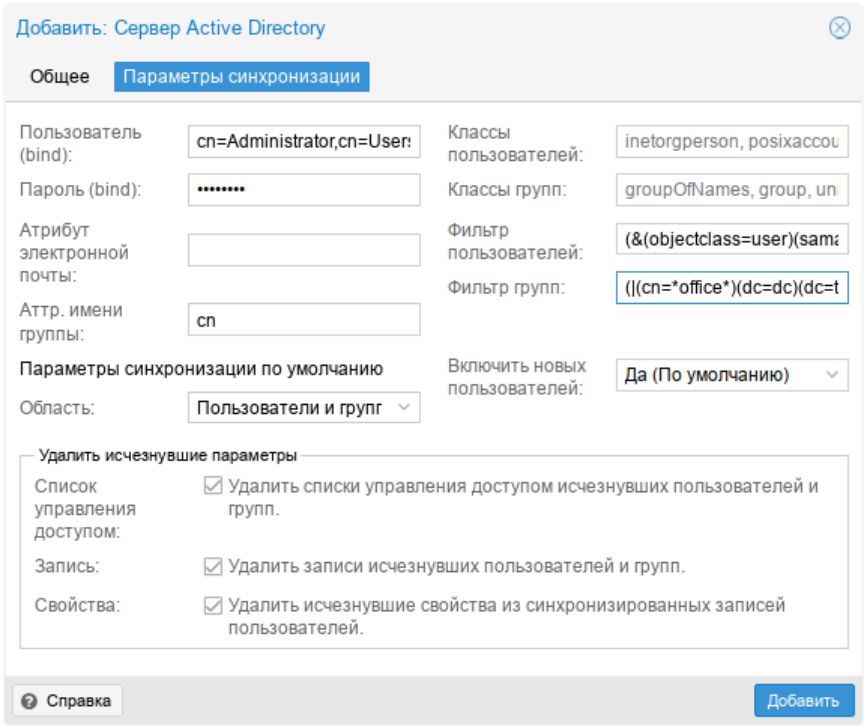

Рис. 380 – Настройка AD аутентификации (вкладка «Параметры синхронизации»)

П р и м е ч а н и е . Команда синхронизации пользователей и групп:

```
# pveum realm sync test.alt
```

Для автоматической синхронизации пользователей и групп можно добавить команду синхронизации в планировщик задач.

8.17.4. Двухфакторная аутентификация

В PVE можно настроить двухфакторную аутентификацию двумя способами:

- требование двухфакторной аутентификации (ДФА) можно включить при настройке области аутентификации. Если в области аутентификации включена ДФА, это становится требованием, и только пользователи с настроенным ДФА смогут войти в систему. Новому пользователю необходимо сразу добавить ключи, так как возможности войти в систему, без предъявления второго фактора, нет. Настроить принудительную двухфакторную аутентификацию можно при добавлении или редактировании области аутентификации (рис. 381);
- пользователи могут сами настроить двухфакторную аутентификацию (рис. 382), даже если она не требуется в области аутентификации (пункт TFA в выпадающем списке пользователя (рис. 383).

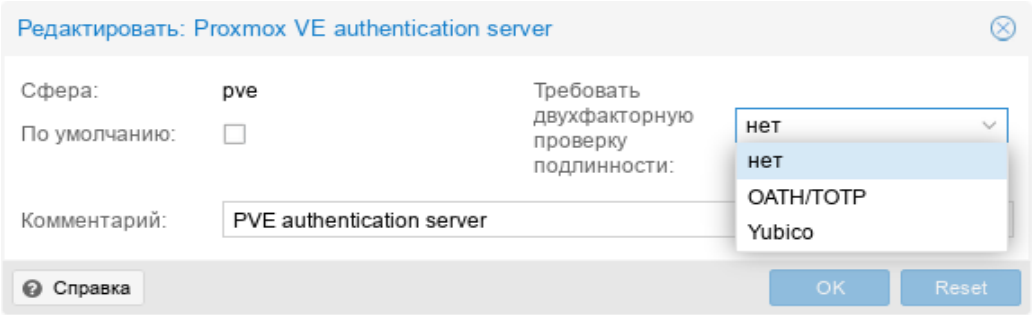

Рис. 381 – Настройка двухфакторной аутентификации при редактировании области

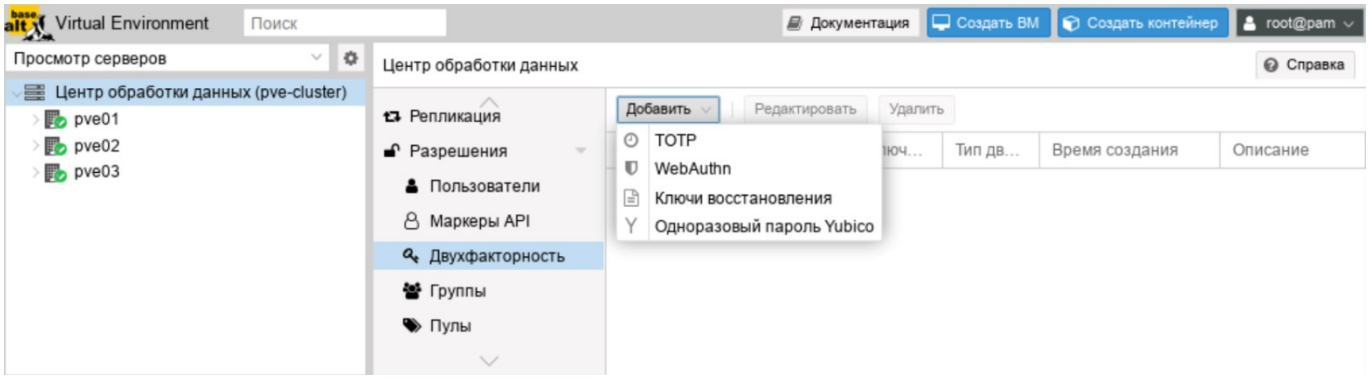

Рис. 382 – Настройка двухфакторной аутентификации пользователем

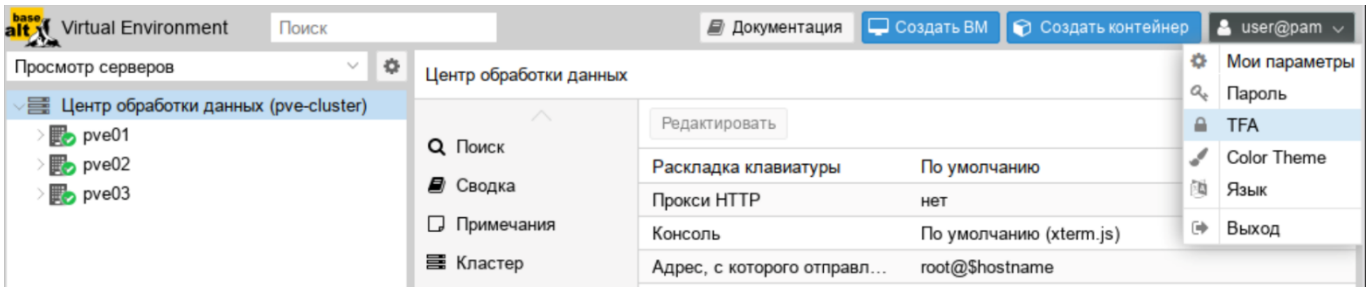

Рис. 383 – Меню пользователя

При добавлении в области аутентификации доступны следующие методы двухфакторной аутентификации (рис. 381):

- «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. 384). 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.

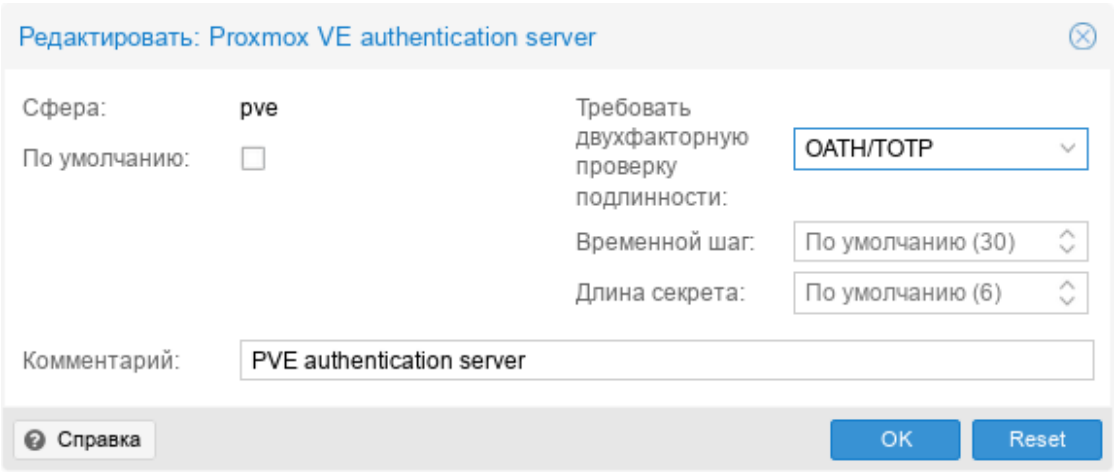

Рис. 384 – Основанная на времени OATH (TOTP)

В дополнение к TOTP и Yubikey OTP пользователям доступны следующие методы двухфакторной аутентификации (рис. 382):

- «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;
- «Ключи восстановления» (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.

Примечания:

1. Пользователи могут использовать TOTP или WebAuthn в качестве второго фактора при входе в систему, только если область аутентификации не применяет YubiKey OTP.

2. Чтобы избежать ситуации, когда потеря электронного ключа навсегда блокирует доступ можно настроить несколько вторых факторов для одной учетной записи (рис. 385).

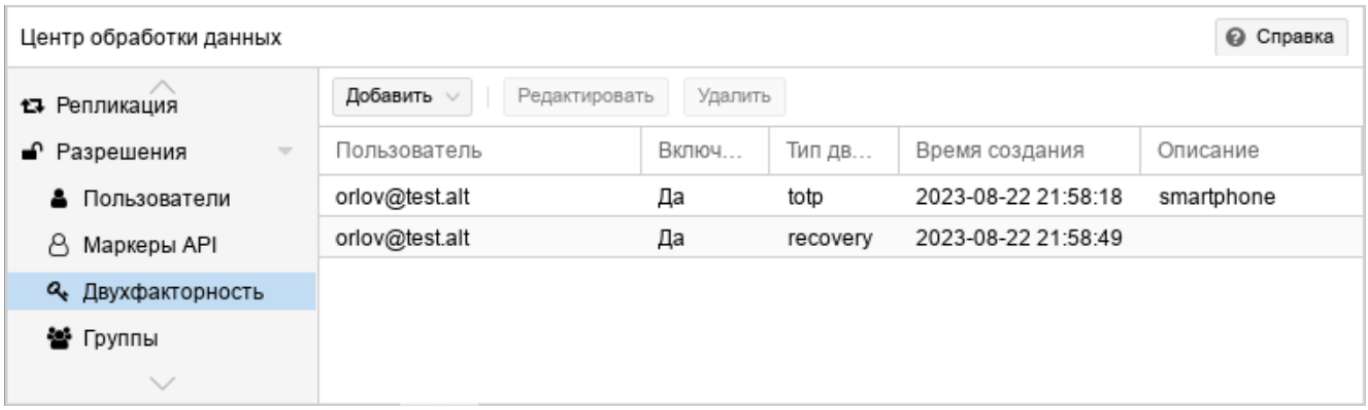

Рис. 385 – Несколько настроенных вторых факторов для учетной записи

Настройка аутентификации TOTP:

- добавление аутентификации TOTP на сервере (рис. 386);
- использование TOTP при аутентификации пользователя (рис. 387).

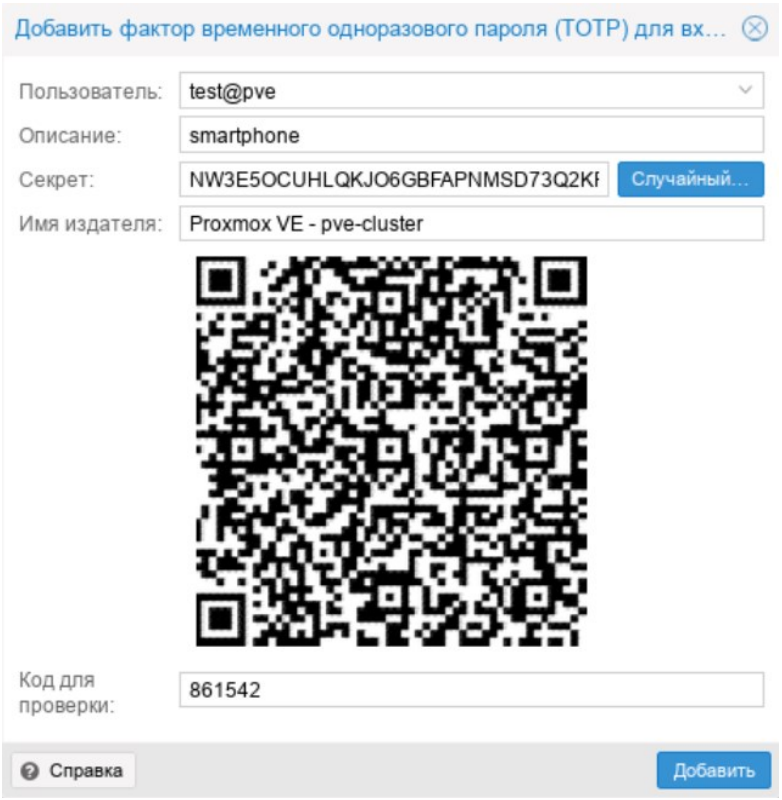

Рис. 386 – PVE. Настройка аутентификации TOTP

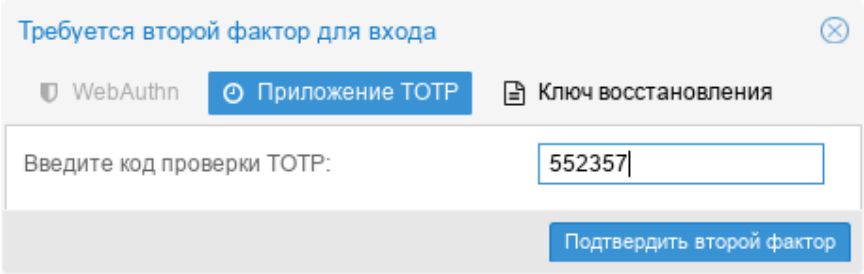

Рис. 387 – PVE. Запрос второго фактора (TOTP) при аутентификации пользователя в веб-интерфейсе

Настройка аутентификации Recovery Key:

- создание набора ключей: (рис. 388);
- использование Recovery Key при аутентификации пользователя (рис. 389).

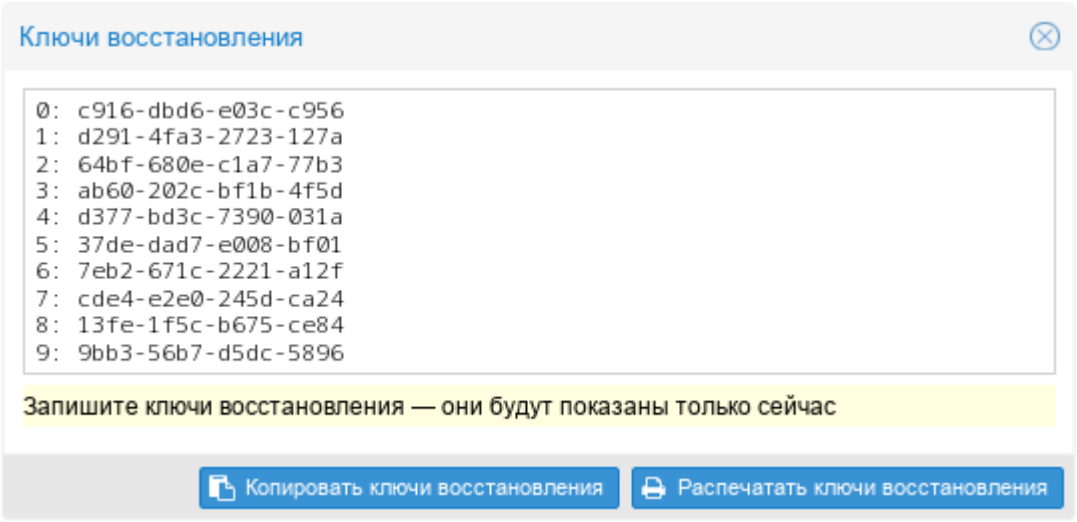

Рис. 388 – PVE. Настройка аутентификации «Ключи восстановления»

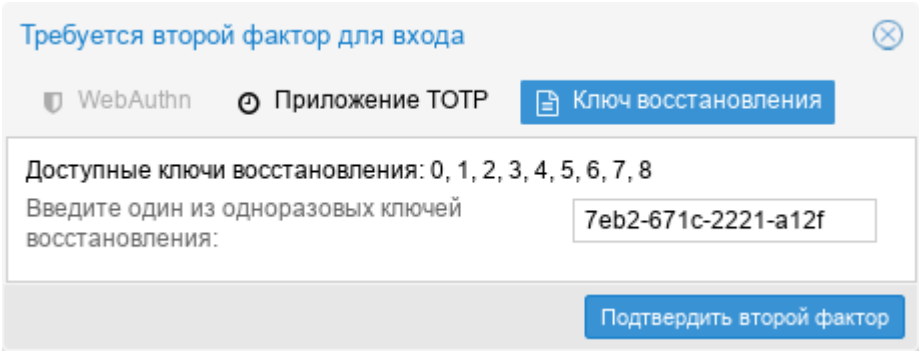

Рис. 389 – PVE. Запрос второго фактора (Recovery Key) при аутентификации пользователя в веб-интерфейсе

8.17.5. 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. 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 предопределен 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 «Центр обработки данных» → «Разрешения» → «Роли» (рис. 390).

Добавить новую роль можно как в веб-интерфейсе, так и в командной строке.

Пример добавления роли в командной строке:

```
# pveum role add VM_Power-only --privs "VM.PowerMgmt VM.Console"
```

Привилегия – это право на выполнение определенного действия. Для упрощения управления списки привилегий сгруппированы в роли, которые затем можно использовать в таблице разрешений. Привилегии не могут быть напрямую назначены пользователям, не будучи частью роли.

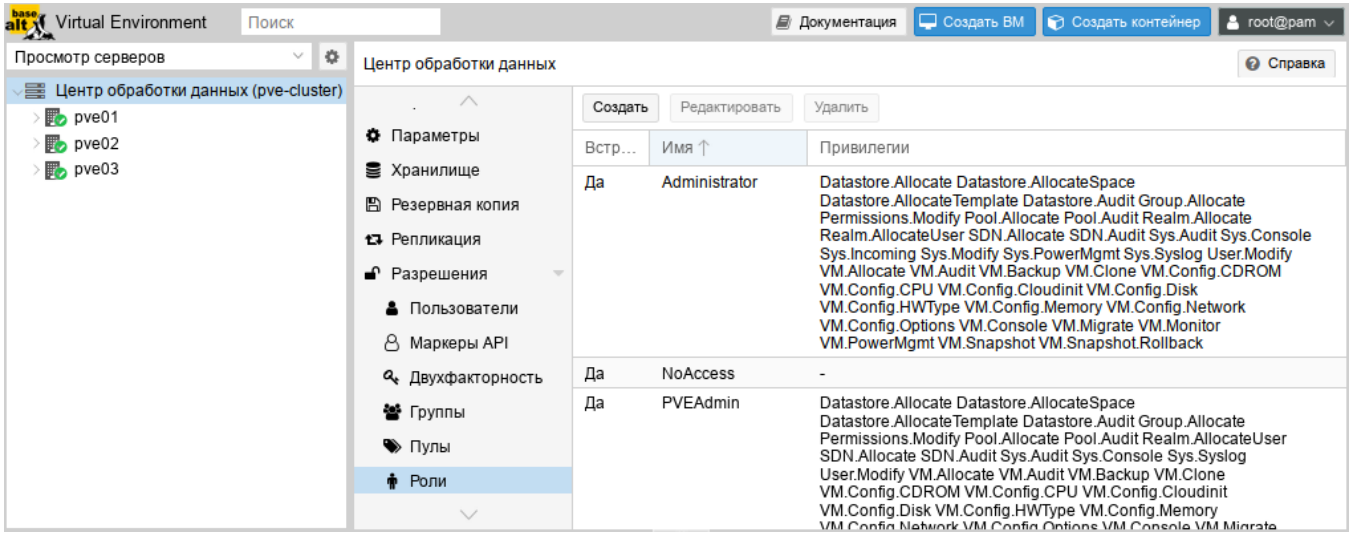

Рис. 390 – Список предопределенных ролей

Список используемых привилегий приведен в таблице 25.

Т а б л и ц а 25 – 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.

Путь может представлять шаблон. Когда API-вызов требует разрешений на шаблонный путь, путь может содержать ссылки на параметры вызова API. Эти ссылки указываются в фигурных скобках. Некоторые параметры неявно берутся из URI вызова API. Например, путь `/nodes/{node}` при вызове `/nodes/pve01/status` требует разрешений на `/nodes/pve01`, в то время как путь `{path}` в запросе PUT к `/access/acl` ссылается на параметр метода `path`.

Примеры:

- `/nodes/{node}` – доступ к серверам PVE;
- `/vms` – распространяется на все ВМ;
- `/vms/{vmid}` – доступ к определенным ВМ;
- `/storage/{storeid}` – доступ к определенным хранилищам;
- `/access/groups` – администрирование групп;
- `/access/realms/{realmid}` – административный доступ к области аутентификации.

Используются следующие правила наследования:

- разрешения для отдельных пользователей всегда заменяют разрешения для групп;
- разрешения для групп применяются, если пользователь является членом этой группы;
- разрешения на более глубоких уровнях перекрывают разрешения, унаследованные от верхнего уровня.

Кроме того, токены с разделением привилегий (см. п. 8.17.1) не могут обладать разрешениями на пути, которых нет у связанного с ними пользователя.

Для назначения разрешений необходимо в окне «Центр обработки данных» → «Разрешения» нажать на кнопку «Добавить», в выпадающем меню выбрать «Разрешения группы», если разрешения назначаются группе пользователей, или

выбрать «Разрешения пользователя», если разрешения назначаются пользователю (рис. 391).

Далее в открывшемся окне выбрать путь, группу и роль и нажать на кнопку «Добавить» (рис. 392).

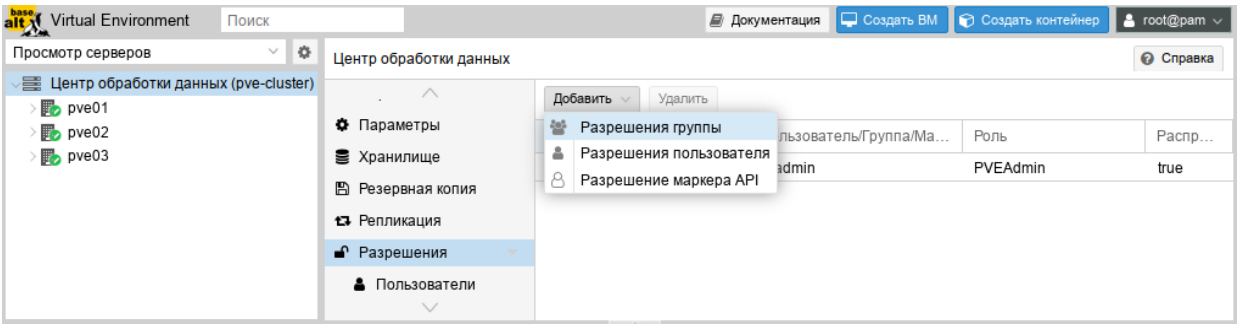

Рис. 391 – Добавление разрешений

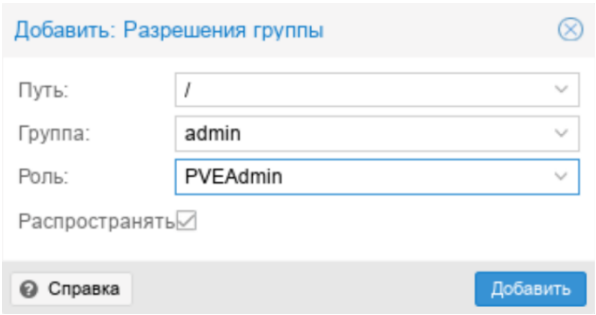

Рис. 392 – Добавление разрешений группе

Примеры работы с разрешениями в командной строке:

- предоставить группе admin полные права администратора:

```
# pveum acl modify / --groups admin --roles Administrator
```

- предоставить пользователю test@pve доступ к ВМ только для чтения:

```
# pveum acl modify /vms --users test@pve --roles PVEAuditor
```

- делегировать управление пользователями пользователю test@pve:

```
# pveum acl modify /access --users test@pve --roles PVEUserAdmin
```

- разрешить пользователю orlov@test.alt изменять пользователей в области test.alt, если они являются членами группы office-test.alt:

```
# pveum acl modify /access/realm/test.alt --users orlov@test.alt \
--roles PVEUserAdmin
# pveum acl modify /access/groups/office-test.alt --users \
orlov@test.alt --roles PVEUserAdmin
```

- разрешить пользователям группы developers администрировать ресурсы, назначенные пулу IT:

```
# pveum acl modify /pool/IT/ --groups developers --roles PVEAdmin
```

- удалить у пользователя test@pve право на просмотр ВМ:

```
 # pveum acl delete /vms --users test@pve --roles PVEAuditor
```

Примечание. Назначение привилегий на токены см. п. 8.17.1.

8.18. Просмотр событий PVE

При устранении неполадок сервера, например, неудачных заданий резервного копирования, полезно иметь журнал ранее выполненных задач.

Действия, такие как, создание ВМ, выполняются в фоновом режиме. Такое фоновое задание называется задачей. Вывод каждой задачи сохраняется в отдельный файл журнала. Получить доступ к истории задач узлов можно с помощью команды `pvenode task`, а также в веб-интерфейсе PVE.

8.18.1. Просмотр событий с помощью `pvenode task`

Команды `pvenode task` приведены в таблице 26.

Примечание. Формат идентификатора задачи (UPID):

```
UPID:$node:$pid:$pstart:$starttime:$dtype:$id:$user
```

pid, pstart и starttime имеют шестнадцатеричную кодировку.

Примеры использования команды `pvenode task`:

1) получить список завершенных задач, связанных с ВМ 105, которые завершились с ошибкой:

```
# pvenode task list --errors --vmid 105
```

Список задач будет представлен в виде таблицы (рис. 393).

2) получить список задач пользователя user:

```
# pvenode task list --userfilter user
```

Таблица 26 – Команды `pvenode task`

Команда	Описание
`pvenode task list [Параметры]`	Вывести список выполненных задач для данного узла. - `--errors <логическое значение>` – вывести только те задачи, которые завершились ошибкой (по умолчанию 0); - `--limit <целое число>` – количество задач, которые должны быть выведены (по умолчанию 50); - `--since <целое число>` – отметка времени (эпоха Unix), начиная с которой будут показаны задачи; - `--source <active | all | archive>` – вывести список активных, всех или завершенных (по умолчанию) задач; - `--start <целое число>` – смещение, начиная с которого будут выведены задачи (по умолчанию 0); - `--statusfilter <строка>` – статус задач, которые должны быть показаны; - `--typefilter <строка>`– вывести задачи указанного типа (например, vzstart, vzdump); - `--until <целое число>` – отметка времени (эпоха Unix), до которой будут показаны задачи; - `--userfilter <строка>` – пользователь, чьи задачи будут показаны; - `--vmid <целое число>` – идентификатор ВМ, задачи которой будут показаны.
`pvenode task log <upid> [Параметры]`	Вывести журнал задачи. - `<upid>: <строка>` – идентификатор задачи; - `--start <целое число>` – при чтении журнала задачи начать с этой строки (по умолчанию 0).
`pvenode task status <upid>`	Вывести статус задачи. vmid – идентификатор задачи.

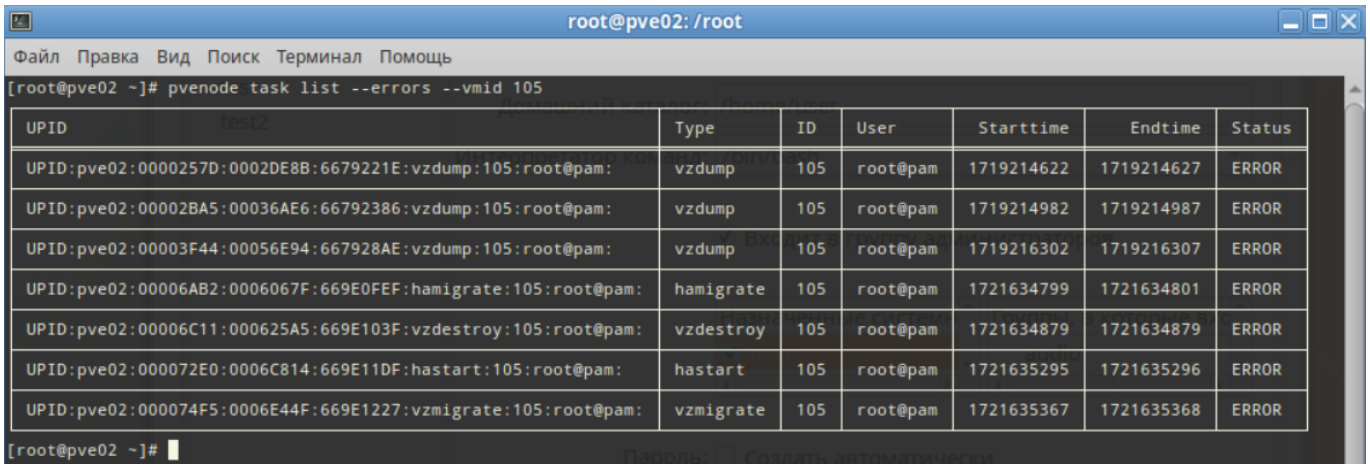

Рис. 393 – Список задач, связанных с ВМ 105

3) вывести журнал задачи, используя ее UPID:

```
# pvenode task log UPID:pve02:0000257D:0002DE8B:6679221E:vzdump:105:root@pam:
INFO: starting new backup job: vzdump 105 --node pve02 --compress zstd --mailnotification always --notes-template '{{guestname}}' --storage nfs-backup --quiet 1 --mailto test@basealt.ru --mode snapshot
INFO: Starting Backup of VM 105 (lxc)
INFO: Backup started at 2024-06-24 09:37:03
INFO: status = stopped
INFO: backup mode: stop
INFO: ionice priority: 7
INFO: CT Name: NewLXC
INFO: including mount point rootfs ('/') in backup
INFO: creating vzdump archive '/mnt/pve/nfs-backup/dump/vzdump-lxc-105-2024_06_24-09_37_03.tar.zst'
ERROR: Backup of VM 105 failed - volume 'local:105/vm-105-disk-0.raw' does not exist
INFO: Failed at 2024-06-24 09:37:04
INFO: Backup job finished with errors
postdrop: warning: unable to look up public/pickup: No such file or directory
TASK ERROR: job errors
```

4) вывести статус задачи, используя ее UPID (рис. 394):

```
# pvenode task status
UPID:pve02:0000257D:0002DE8B:6679221E:vzdump:105:root@pam:
```

key	value
exitstatus	job errors
id	105
node	pve02
pid	9597
starttime	1719214622
status	stopped
type	vzdump
upid	UPID:pve02:0000257D:0002DE8B:6679221E:vzdump:105:root@pam:
user	root@pam

Рис. 394 – Пример вывода

8.18.2. Просмотр событий в веб-интерфейсе PVE

8.18.2.1. Панель журнала

Основная цель панели журнала – показать, что в данный момент происходит в кластере. Панель журнала расположена в нижней части интерфейса PVE (рис. 395).

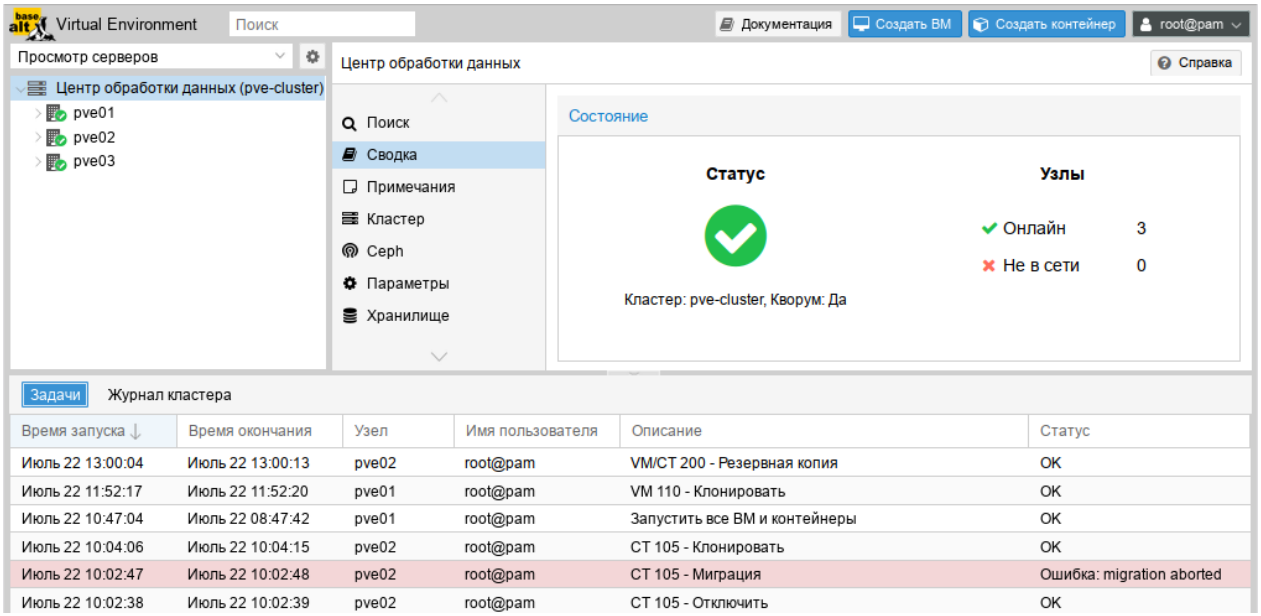

Рис. 395 – Панель журнала

На панели журнала (вкладка «Задачи») отображаются последние задачи со всех узлов кластера. Таким образом, здесь можно в режиме реального времени видеть, что кто-то еще работает на другом узле кластера.

Для того чтобы получить подробную информацию о задаче или прервать выполнение выполняемой задачи, следует дважды щелкнуть мышью по записи журнала. Откроется окно (рис. 396) с журналом задачи (вкладка «Выход») и ее статусом (вкладка «Статус»). Нажав кнопку «Остановить» можно остановить выполняемую задачу. Кнопка «Загрузка» позволяет сохранить журнал задачи в файл.

Примечание. Кнопка «Остановить» доступна, только если задача еще выполняется.

Некоторые кратковременные действия просто отправляют логи всем членам кластера. Эти сообщения можно увидеть на панели журнала на вкладке «Журнал кластера».

Примечание. Панель журнала можно полностью скрыть, если нужно больше места для отображения другого контента

На вкладке «Задачи» панели журнала отображаются записи журнала только для недавних задач. Найти все задачи можно в журнале задач узла PVE.

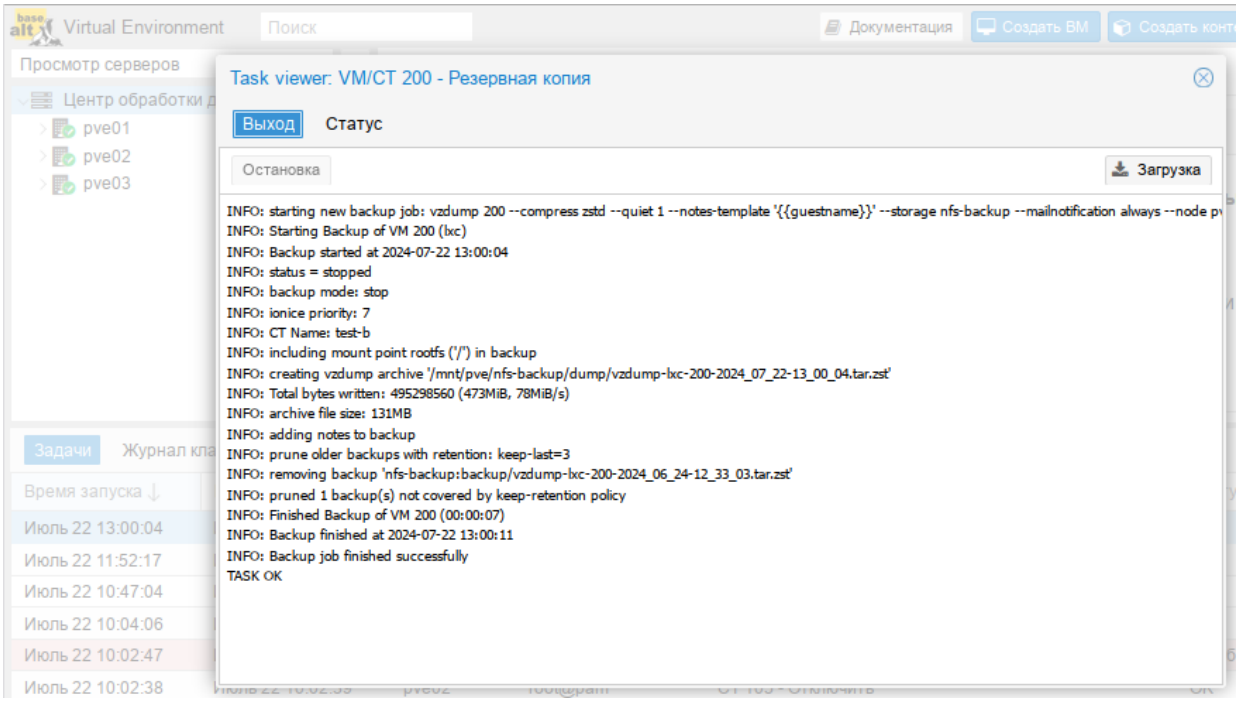

Рис. 396 – Информация о задаче

8.18.2.2. Журнал задач узла PVE

Просмотреть список всех задач узла PVE можно, выбрав «Узел» → «Журнал задач» (рис. 397). Записи журнала можно отфильтровать. Для этого следует нажать на кнопку «Фильтр» и задать нужные значения фильтра (рис. 398). Просмотреть журнал задачи можно, дважды щелкнув запись журнала задач или нажав кнопку «Просмотр».

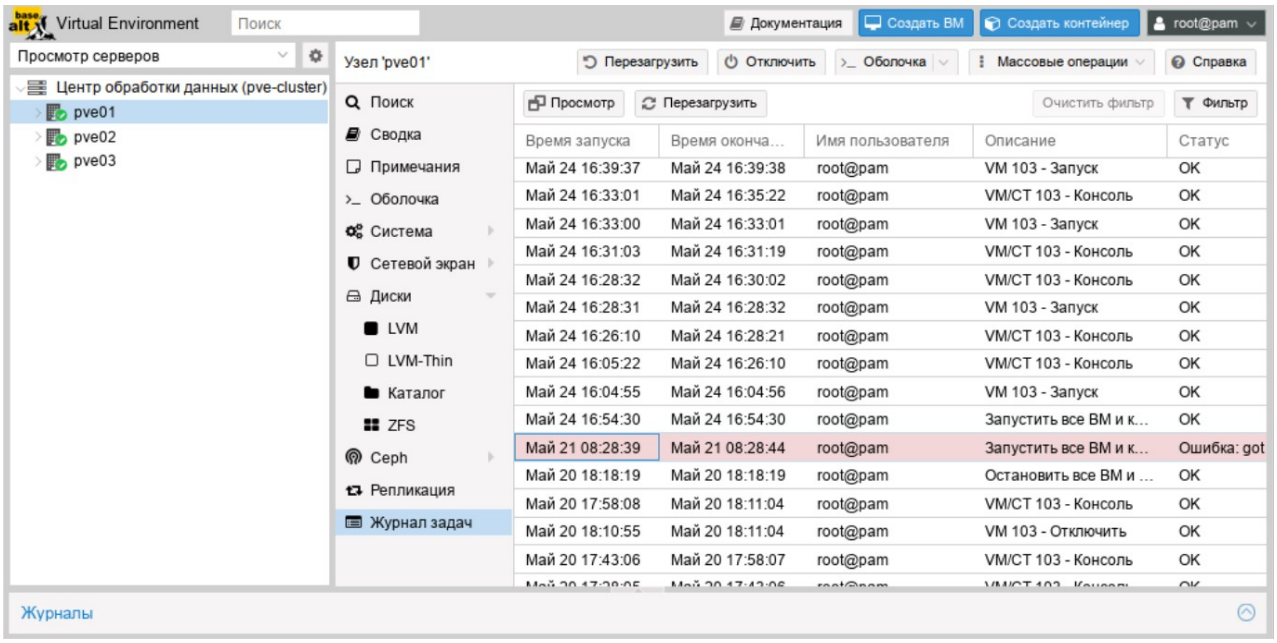

Рис. 397 – Журнал задач узла pve01

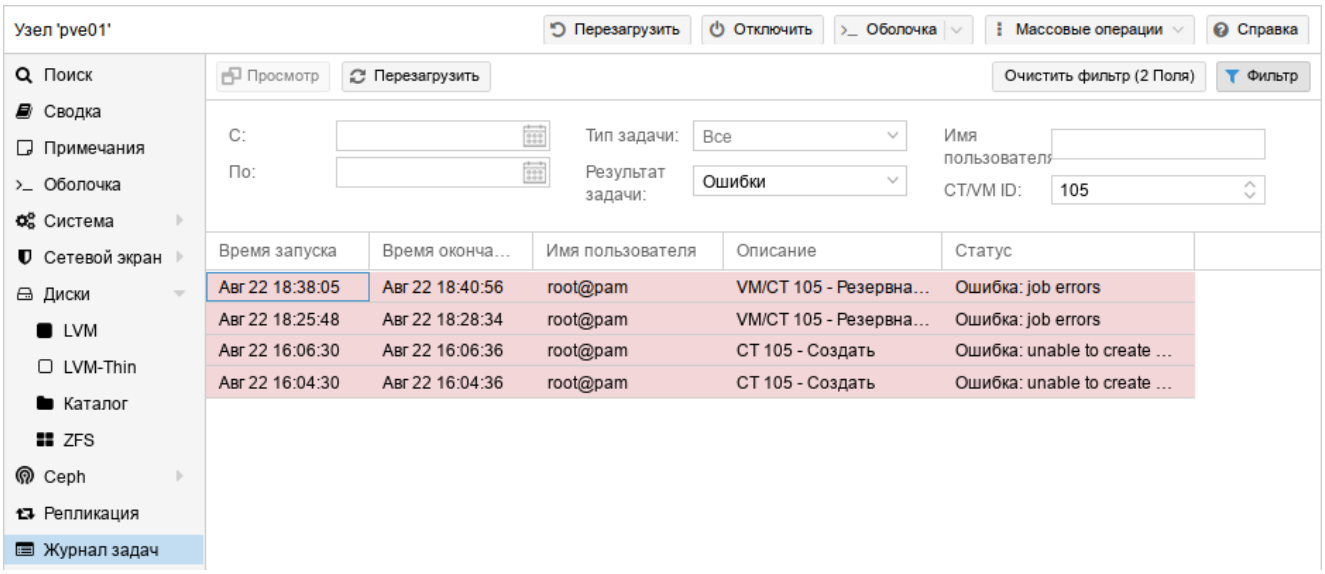

Рис. 398 – Отфильтрованные задачи узла pve01

8.18.2.3. Журнал задач ВМ

Для просмотра задач ВМ необходимо выбрать «Узел» → «ВМ» → «Журнал задач» (рис. 399). Записи журнала можно отфильтровать. Для этого следует нажать на кнопку «Фильтр» и задать нужные значения фильтра (рис. 400). Просмотреть журнал задачи можно, дважды щелкнув запись журнала задач или нажав кнопку «Просмотр».

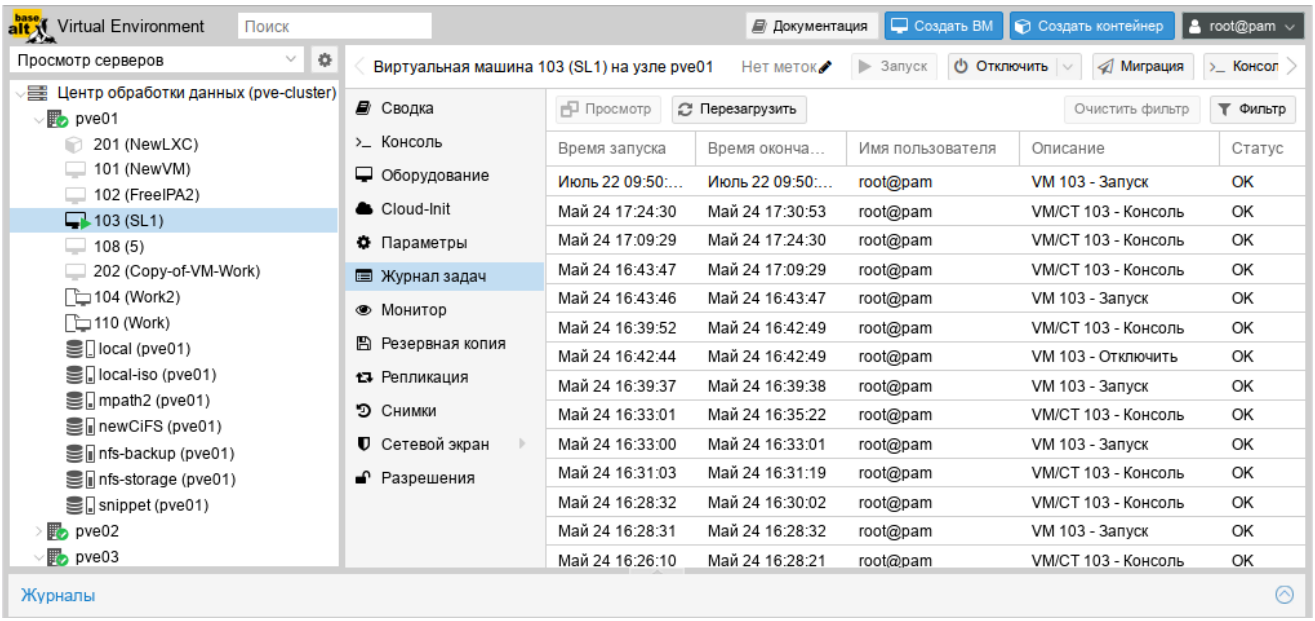

Рис. 399 – Журнал задач ВМ 103

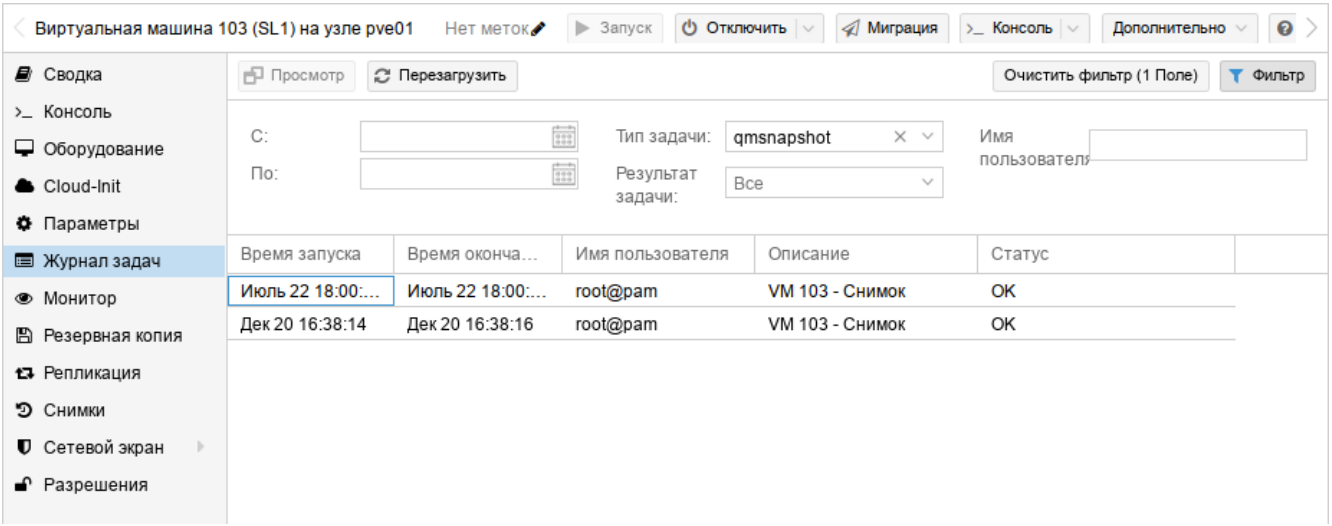

Рис. 400 – Задачи ВМ 103 типа qmsnapshot

8.19. PVE API

PVE использует RESTful API. В качестве основного формата данных используется JSON, и весь API формально определен с использованием JSON Schema.

Документация API доступна по адресу:

https://docs.altlinux.org/pve-api/v7/index.html

Каждая команда, доступная команде `pvesh`, доступна в веб-API, поскольку они используют одну и ту же конечную точку.

Запрос (URL, к которому происходит обращение) содержит четыре компонента:

- конечная точка, являющаяся URL-адресом, по которому отправляется запрос;
- метод с типом (GET, POST, PUT, PATCH, DELETE);
- заголовки, выполняющие функции аутентификации, предоставление информации о содержимом тела (допустимо использовать параметр `-H` или `--header` для отправки заголовков HTTP) и т. д.;
- данные (или тело) – то, что отправляется на сервер с помощью опции `-d` или `--data` при запросах POST, PUT, PATCH или DELETE.

Примечание. При передаче не буквенно-цифровых параметров нужно кодировать тело HTTP-запроса. Для этого можно использовать опцию `--data-urlencode`.

HTTP-запросы разрешают работать с базой данных, например:

- GET-запрос на чтение или получение ресурса с сервера;
- POST-запрос для создания записей;
- PUT-запрос для изменения записей;
- DELETE-запрос для удаления записей;
- PATCH-запрос для обновления записей.

Для передачи команд через REST API можно использовать утилиту `curl`.

Примечание. По мере роста числа пользователей и ВМ, API PVE может начать реагировать на изменения с задержкой. Для решения этой проблемы нужно очистить `/var/lib/rrdcached/`, например, выполнив команду:

```
# find /var/lib/rrdcached -type f -mtime +5 -delete
```

Или, добавив соответствующее задание в crontab.

8.19.1. URL API

API PVE использует протокол HTTPS, а сервер прослушивает порт 8006. Таким образом, базовый URL для API – `https://server:8006/api2/json/`.

Параметры можно передавать с помощью стандартных методов HTTP:

- через URL;

- используя `x-www-form-urlencoded content-type` для запросов PUT и POST.

В URL можно указать формат возвращаемых данных:

- json – формат JSON;
- extjs – формат JSON, но результат вложен в объект, с объектом данных, вариант, совместимый с формами ExtJS;
- html – данные в формате HTML (иногда полезно для отладки);
- text – формат простой текст (иногда полезно для отладки);

В приведенном выше примере используется JSON.

8.19.2. Аутентификация

Есть два способа доступа к API PVE:

- использование временно сгенерированного токена (билета);
- использование API-токена.

Все API-запросы должны включать в себя билет в заголовке Cookie или отправлять API-токен через заголовок Authorization.

8.19.2.1. Билет Cookie

Билет – это подписанное случайное текстовое значение с указанием пользователя и времени создания. Билеты подписываются общекластерным ключом аутентификации, который обновляется один раз в день.

Кроме того, любой запрос на запись (POST/PUT/DELETE) должен содержать CSRF-токен для предотвращения CSRF-атак (cross-site request forgery).

Пример получения нового билета и CSRF-токена:

```
$ curl -k -d 'username=root@pam' --data-urlencode 'password=xxxxxxxxx' \
https://192.168.0.186:8006/api2/json/access/ticket
```

П р и м е ч а н и е . Параметры командной строки видны всей системе, поэтому следует избегать запуска команды с указанием пароля на ненадежных узлах.

Пример получения нового билета и CSRF-токена с паролем, записанным в файл, доступный для чтения только пользователю:

```
$ curl -k -d 'username=root@pam' \
--data-urlencode "password@$HOME/.pve-pass-file" \
https://192.168.0.186:8006/api2/json/access/ticket
```

Примечание. Для форматированного вывода можно использовать команду `jq` (должен быть установлен пакет jq):

```
$ curl -k -d 'username=root@pam' \
--data-urlencode "password@$HOME/.pve-pass-file" \
https://192.168.0.186:8006/api2/json/access/ticket | jq
```

Пример ответа:

```
{
  "data": {
    "ticket":"PVE:root@pam:66AA52D6::d85E+IIFAuG731…",
    "CSRFPreventionToken":"66AA52D6:Y2zvIXjRVpxx4ZG74F14Ab0EHn8NRoso/WmVqZEnAuM",
    "username":"root@pam"
  }
}
```

Примечание. Билет действителен в течение двух часов и должен быть повторно запрошен по истечении срока его действия. Но можно получить новый билет, передав старый билет в качестве пароля методу `/access/ticket` до истечения срока его действия.

Полученный билет необходимо передавать с Cookie при любом запросе, например:

```
$ curl -k -b "PVEAuthCookie=PVE:root@pam:66AA52D6::d85E+IIFAuG731…" \
https://192.168.0.186:8006/api2/json/
```

Ответ:

```
{
  "data": [
    { "subdir": "version" },
    { "subdir": "cluster" },
    { "subdir": "nodes" },
    { "subdir": "storage" },
    { "subdir": "access" },
    { "subdir": "pools" }
  ]
}
```

Примечание. Для передачи данных в заголовке Cookie используется параметр `--cookie` (`-b`).

Любой запрос на запись (POST, PUT, DELETE) кроме билета должен включать заголовок `CSRFPreventionToken`, например:

```
$ curl -k -XDELETE \
'https://pve01:8006/api2/json/access/users/testuser@pve' \
-b "PVEAuthCookie=PVE:root@pam:66AA52D6::d85E+IIFAuG731…" \
-H "CSRFPreventionToken:
66AA52D6:Y2zvIXjRVpxx4ZG74F14Ab0EHn8NRoso/WmVqZEnAuM"
```

8.19.2.2. API-токены

API-токены позволяют другой системе, программному обеспечению или API-клиенту получать доступ без сохранения состояния к большинству частей REST API. Токены могут быть сгенерированы для отдельных пользователей и им могут быть предоставлены отдельные разрешения и даты истечения срока действия для ограничения объема и продолжительности доступа (подробнее см. п. 8.17.1). Если API-токен будет скомпрометирован, его можно отозвать, не отключая самого пользователя.

Примеры запросов с использованием API-токена:

- получить список пользователей:

```
$ curl -H 'Authorization: PVEAPIToken=root@pam!test=373007e1-
4ecb-4e56-b843-d0fbed543375' \
https://192.168.0.186:8006/api2/json/access/users
```

- добавить пользователя `testuser@pve`:

```
$ curl -k -X 'POST' \
'https://pve01:8006/api2/json/access/users' \
--data-urlencode 'userid=testuser@pve' \
-H 'Authorization: PVEAPIToken=root@pam!test=373007e1-4ecb-4e56-
b843-d0fbed543375'
```

- удалить пользователя `testuser@pve`:

```
$ curl -k -X 'DELETE' \
'https://pve01:8006/api2/json/access/users/testuser@pve' \
-H 'Authorization: PVEAPIToken=root@pam!test=373007e1-4ecb-4e56-
b843-d0fbed543375'
```

Примечание. Если запрос завершается ошибкой вида:

```
curl: (60) SSL certificate problem: unable to get local issuer certificate
```

можно дополнить запрос опцией `--insecure` (`-k`), для отключения проверки валидности сертификатов:

```
$ curl -k -H 'Authorization: PVEAPIToken=root@pam!test=373007e1-
4ecb-4e56-b843-d0fbed543375' https://192.168.0.186:8006/api2/json/
```

Примечание. API-токены не нуждаются в значениях CSRF для POST, PUT или DELETE запросов. Обычно токены не используются в контексте браузера, поэтому основной вектор атаки CSRF изначально неприменим.

8.19.3. Пример создания контейнера с использованием API

Исходные данные:

- `APINODE` – узел, на котором производится аутентификация;
- `TARGETNODE` – узел, на котором будет создан контейнер;
- `cookie` – файл, в который будет помещен cookie;
- `csrftoken` – файл, в который будет помещен CSRF-токен.

Пример создания контейнера с использованием API:

- для удобства установить переменные окружения:

```
$ export APINODE=pve01
$ export TARGETNODE=pve03
```

- сохранить авторизационный cookie в файл `cookie`:

```
$ curl --silent --insecure --data \
"username=root@pam&password=yourpassword" \
 https://$APINODE:8006/api2/json/access/ticket \
| jq --raw-output '.data.ticket' | sed 's/^/PVEAuthCookie=/' > cookie
```

- сохранить CSRF-токен в файл `csrftoken`:

```
$ curl --silent --insecure --data \
"username=root@pam&password=yourpassword" \
 https://$APINODE:8006/api2/json/access/ticket \
| jq --raw-output '.data.CSRFPreventionToken' \
| sed 's/^/CSRFPreventionToken:/' > csrftoken
```

- отобразить статус целевого узла, чтобы проверить, что создание куки-файла сработало:

```
$ curl --insecure --cookie \
"$(<cookie)" https://$APINODE:8006/api2/json/nodes/$TARGETNODE/status \
| jq '.'
```

- создать LXC-контейнер:

```
$ curl --silent --insecure --cookie "$(<cookie)" \
 --header "$(<csrftoken)" -X POST \
 --data-urlencode net0="name=myct0,bridge=vmbr0" \
 --data-urlencode ostemplate="local:vztmpl/alt-p10-rootfs-systemd-x86_64.tar.xz" \
 --data vmid=601 \
 https://$APINODE:8006/api2/json/nodes/$TARGETNODE/lxc

{"data":"UPID:pve03:00005470:00083F6D:66A76C80:vzcreate:601:root@pam:"}
```

Команда должна вернуть структуру JSON, содержащую идентификатор задачи (UPID).

Примечание. При создании контейнера должен использоваться доступный vmid.

8.19.4. Утилита pvesh

Инструмент управления PVE (pvesh) позволяет напрямую вызывать функции API, без использования сервера REST/HTTPS.

```
# pvesh ls /
Dr---       access
Dr---       cluster
Dr---       nodes
Dr-c-       pools
Dr-c-       storage
-r---       version
```

Примечание. pvesh может использовать только пользователь root.

Инструмент автоматически проксирует вызовы другим членам кластера с помощью ssh.

Примеры:

- вывести текущую версию:

```
# pvesh get /version
```

- получить список узлов в кластере:

```
#pvesh get /nodes
```

- получить список доступных опций для центра обработки данных:

```
# pvesh usage cluster/options -v
```

- создать нового пользователя:

```
# pvesh create /access/users --userid testuser@pve
```

- удалить пользователя:

```
# pvesh delete /access/users/testuser@pve
```

- установить консоль HTML5 NoVNC в качестве консоли по умолчанию:

```
# pvesh set cluster/options -console html5
```

- создать и запустить новый контейнер на узле pve03:

```
# pvesh create nodes/pve03/lxc -vmid 210 -hostname test \
--storage local \
--password "supersecret" \
```

```
--ostemplate \
nfs-storage:vztmpl/alt-p10-rootfs-systemd-x86_64.tar.xz \
--memory 512 --swap 512


UPID:pve03:0000286E:0003553C:66A75FE7:vzcreate:210:root@pam:

# pvesh create /nodes/pve03/lxc/210/status/start
UPID:pve03:0000294B:00036B33:66A7601F:vzstart:210:root@pam
```

8.20. Основные службы PVE

Команды служб PVE на примере pvedaemon:

- вывести справку:

```
# pvedaemon help
```

- перезапустить службу (или запустить, если она не запущена):

```
# pvedaemon restart
```

- запустить службу:

```
# pvedaemon start
```

- запустить службу в режиме отладки:

```
# pvedaemon start --debug 1
```

- вывести статус службы:

```
# pvedaemon status
```

- остановить службу:

```
# pvedaemon stop
```

8.20.1. pvedaemon – служба PVE API

Служба pvedaemon предоставляет весь API PVE на 127.0.0.1:85. Она работает от имени пользователя root и имеет разрешение на выполнение всех привилегированных операций.

П р и м е ч а н и е . Служба слушает только локальный адрес, поэтому к ней нельзя получить доступ извне. Доступ к API извне предоставляет служба pveproxy.

8.20.2. pveproxy – служба PVE API Proxy

Служба pveproxy предоставляет весь PVE API на TCP-порту 8006 с использованием HTTPS. Она работает от имени пользователя www-data и имеет

очень ограниченные разрешения. Операции, требующие дополнительных разрешений, перенаправляются локальному pvedaemon.

Запросы, предназначенные для других узлов, автоматически перенаправляются на них. Поэтому можно управлять всем кластером, подключившись к одному узлу PVE.

8.20.2.1. Управление доступом на основе хоста

Можно настраивать apache2-подобные списки контроля доступа. Значения считываются из файла `/etc/default/pveproxy`. Например:

```
ALLOW_FROM="10.0.0.1-10.0.0.5,192.168.0.0/22"
DENY_FROM="all"
POLICY="allow"
```

IP-адреса можно указывать с использованием любого синтаксиса, понятного `Net::IP`. Ключевое слово `all` является псевдонимом для 0/0 и ::/0 (все адреса IPv4 и IPv6).

Политика по умолчанию – `allow`.

Правила обработки запросов приведены в таблице 27.

Т а б л и ц а 27 – Правила обработки запросов

Соответствие	`POLICY=deny`	`POLICY=allow`
Соответствует только `Allow`	Запрос разрешен	Запрос разрешен
Соответствует только `Deny`	Запрос отклонен	Запрос отклонен
Нет соответствий	Запрос отклонен	Запрос разрешен
Соответствует и `Allow` и `Deny`	Запрос отклонен	Запрос разрешен

8.20.2.2. Прослушиваемый IP-адрес

По умолчанию службы pveproxy и spiceproxy прослушивают подстановочный адрес и принимают соединения от клиентов как IPv4, так и IPv6.

Установив опцию `LISTEN_IP` в `/etc/default/pveproxy`, можно контролировать, к какому IP-адресу будут привязываться службы `pveproxy` и `spiceproxy`. IP-адрес должен быть настроен в системе.

Установка `sysctl net.ipv6.bindv6only` в значение 1 приведет к тому, что службы будут принимать соединения только от клиентов IPv6, что может вызвать множество проблем. Если устанавливается эта конфигурация, рекомендуется либо

удалить настройку `sysctl`, либо установить `LISTEN_IP` в значение 0.0.0.0 (что позволит использовать только клиентов IPv4).

`LISTEN_IP` можно использовать для привязки сокета к внутреннему интерфейсу, например:

```
LISTEN_IP="192.168.0.186"
```

Аналогично можно задать IPv6-адрес:

```
LISTEN_IP="2001:db8:85a3::1"
```

Если указывается локальный IPv6-адрес, необходимо указать имя интерфейса, например:

```
LISTEN_IP="fe80::c463:8cff:feb9:6a4e%vmbr0"
```

Примечание. Не рекомендуется устанавливать `LISTEN_IP` в кластерных системах.

Для применения изменений нужно перезагрузить узел или полностью перезапустить pveproxy и spiceproxy:

```
# systemctl restart pveproxy.service spiceproxy.service
```

Примечание. Перезапуск службы pveproxy, в отличие от перезагрузки конфигурации (reload), может прервать некоторые рабочие процессы, например, запущенную консоль или оболочку ВМ. Поэтому, следует дождаться остановки системы на обслуживания, чтобы это изменение вступило в силу.

8.20.2.3. Набор SSL-шифров

Список шифров можно определить в `/etc/default/pveproxy` с помощью ключей `CIPHERS` (TLS = 1.2) и `CIPHERSUITES` (TLS >= 1.3), например:

```
CIPHERS="ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-
AES128-SHA256:ECDHE-RSA-AES128-SHA256"
CIPHERSUITES="TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
:TLS_AES_128_GCM_SHA256"
```

Кроме того, можно настроить клиент на выбор шифра, используемого в `/etc/default/pveproxy` (по умолчанию используется первый шифр в списке, доступном как клиенту, так и pveproxy):

```
HONOR_CIPHER_ORDER=0
```

8.20.2.4. Поддерживаемые версии TLS

Для отключения TLS версий 1.2 или 1.3, необходимо установить следующий параметр в `/etc/default/pveproxy`:

```
DISABLE_TLS_1_2=1
```

или, соответственно:

```
DISABLE_TLS_1_3=1
```

Примечание. Если нет особой причины, не рекомендуется вручную настраивать поддерживаемые версии TLS.

8.20.2.5. Параметры Диффи-Хеллмана

Определить используемые параметры Диффи-Хеллмана можно в `/etc/default/pveproxy`, указав в параметре `DHPARAMS` путь к файлу, содержащему параметры DH в формате PEM, например:

```
DHPARAMS="/path/to/dhparams.pem"
```

Примечание. Параметры DH используются только в том случае, если согласован набор шифров, использующий алгоритм обмена ключами DH.

8.20.2.6. Альтернативный сертификат HTTPS

pveproxy использует `/etc/pve/local/pveproxy-ssl.pem` и `/etc/pve/local/pveproxy-ssl.key`, если они есть, или `/etc/pve/local/pve-ssl.pem` и `/etc/pve/local/pve-ssl.key` в противном случае. Закрытый ключ не может использовать парольную фразу.

Можно переопределить местоположение закрытого ключа сертификата `/etc/pve/local/pveproxy-ssl.key`, установив `TLS_KEY_FILE` в `/etc/default/pveproxy`, например:

```
TLS_KEY_FILE="/secrets/pveproxy.key"
```

8.20.2.7. Сжатие ответа

По умолчанию pveproxy использует сжатие gzip HTTP-уровня для сжимаемого контента, если клиент его поддерживает. Это поведение можно отключить в `/etc/default/pveproxy`:

```
COMPRESSION=0
```

8.20.3. pvestatd – служба PVE Status

Служба pveproxy запрашивает статус ВМ, хранилищ и контейнеров с регулярными интервалами. Результат отправляется на все узлы кластера.

8.20.4. spiceproxy – служба SPICE Proxy

Служба spiceproxy прослушивает TCP-порт 3128 и реализует HTTP-прокси для пересылки запроса CONNECT от SPICE-клиента к ВМ PVE. Она работает от имени пользователя www-data и имеет минимальные разрешения.

8.20.4.1. Управление доступом на основе хоста

Можно настраивать apache2-подобные списки контроля доступа. Значения считываются из файла `/etc/default/pveproxy`. Подробнее см. п. 8.20.2.

8.20.5. pvescheduler – служба PVE Scheduler

Служба pvescheduler отвечает за запуск заданий по расписанию, например, заданий репликации и vzdump.

Для заданий vzdump служба получает свою конфигурацию из файла `/etc/pve/jobs.cfg`.

9. СИСТЕМА РЕЗЕРВНОГО КОПИРОВАНИЯ PROXMOX BACKUP SERVER

Proxmox Backup Server (PBS) – клиент-серверное решение для резервного копирования и восстановления виртуальных машин, контейнеров и данных с физических узлов. Решение оптимизировано для проекта Proxmox VE (PVE).

Все взаимодействия между клиентом и сервером шифруются с использованием протокола TLS, кроме того, данные могут быть зашифрованы на стороне клиента перед отправкой на сервер. Это позволяет сделать резервное копирование более безопасным.

Сервер резервного копирования хранит данные резервного копирования и предоставляет API для создания хранилищ данных и управления ими. С помощью API также можно управлять дисками и другими ресурсами на стороне сервера.

Клиент резервного копирования использует API для доступа к резервным копиям. С помощью инструмента командной строки proxmox-backup-client можно создавать резервные копии и восстанавливать данные (в PVE клиент встроен).

Для управления настройкой резервного копирования и резервными копиями используется веб-интерфейс. Все административные задачи можно выполнять в веб-браузере. Веб-интерфейс также предоставляет встроенную консоль.

9.1. Установка PBS

9.1.1. Сервер PBS

Установить сервер PBS:

```
# apt-get install proxmox-backup-server
```

Запустить и добавить в автозагрузку Proxmox Backup API Proxy Server:

```
# systemctl enable --now proxmox-backup-proxy.service
```

Служба proxmox-backup-proxy предоставляет API управления PBS по адресу 127.0.0.1:82. Она имеет разрешение на выполнение всех привилегированных операций.

9.1.2. Клиент PBS

Установить клиент PBS:

```
# apt-get install proxmox-backup-client
```

9.2. Веб-интерфейс PBS

Веб-интерфейс PBS доступен по адресу `https://<имя-компьютера>:8007`. Потребуется пройти аутентификацию (рис. 401) (логин по умолчанию: `root`, пароль указывается в процессе установки ОС). Веб-интерфейс PBS показан на рис. 402.

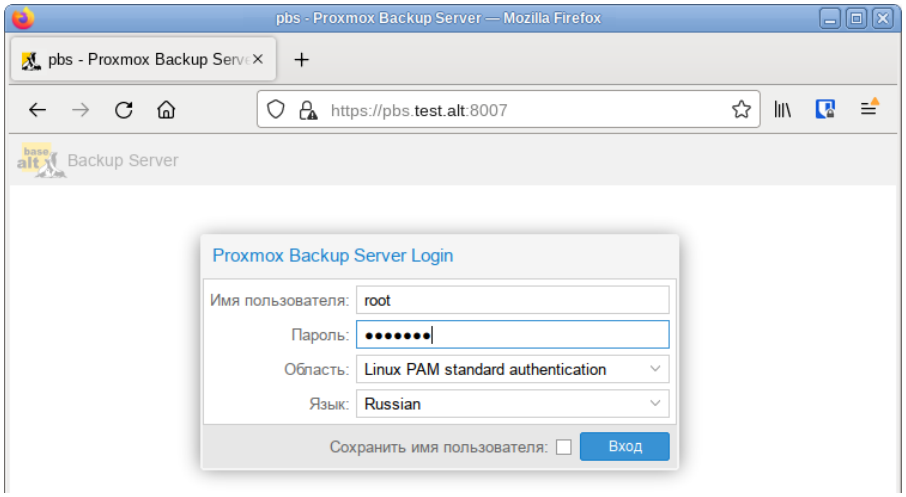

Рис. 401 – Аутентификация в веб-интерфейсе PBS

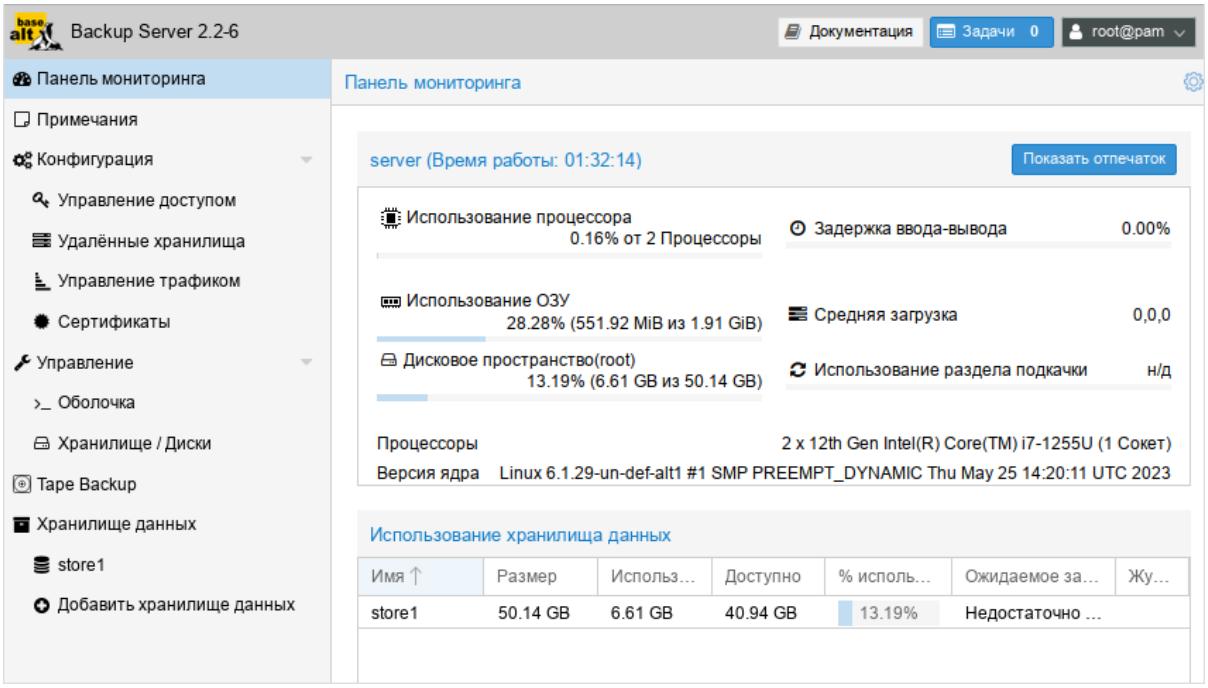

Рис. 402 – Веб-интерфейс PBS

9.3. Настройка хранилища данных

9.3.1. Управление дисками

В веб-интерфейсе на вкладке «Управление» → «Хранилище/Диски» можно увидеть диски, подключенные к системе (рис. 403).

Просмотр списка дисков в командной строке:

```
# proxmox-backup-manager disk list
```

Создание файловой системы ext4 или xfs на диске в веб-интерфейсе показано на рис. 404.

Пример создания файловой системы в командной строке (будет создана файловая система ext4 и хранилище данных на диске nvme0n3, хранилище данных будет создано по адресу `/mnt/datastore/store2`):

```
# proxmox-backup-manager disk fs create store2 --disk nvme0n3\
--filesystem ext4 --add-datastore true
create datastore 'store2' on disk nvme0n3
Chunkstore create: 1%
Chunkstore create: 2%
…
Chunkstore create: 99%
TASK OK
```

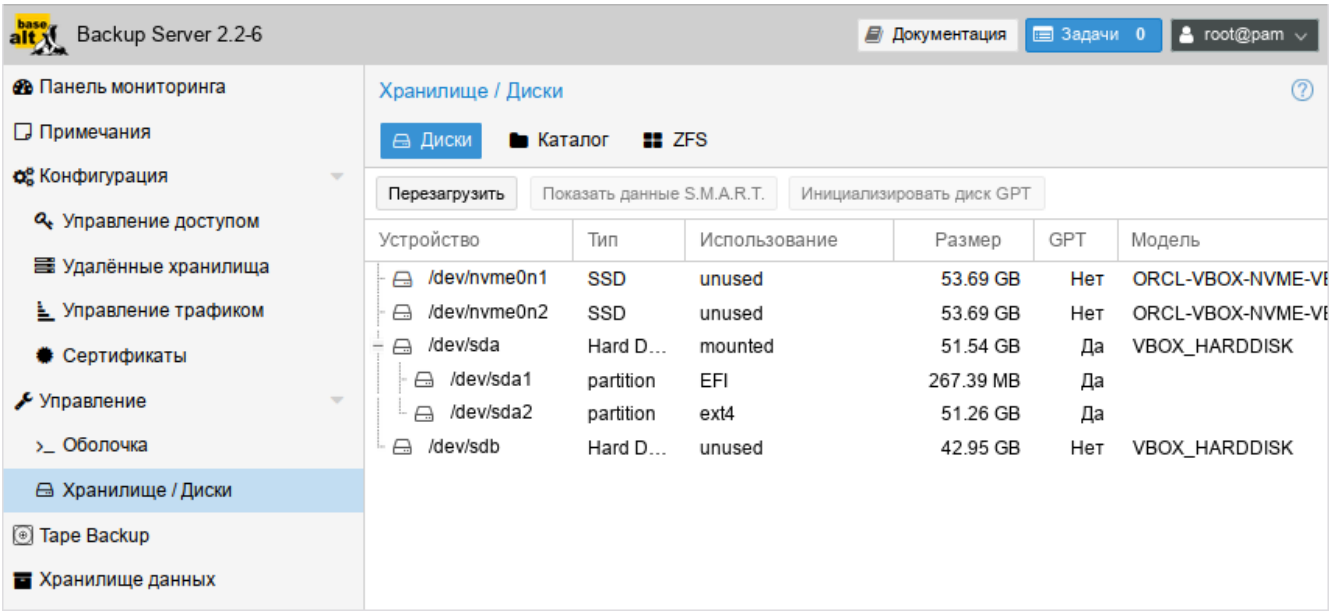

Рис. 403 – PBS. Диски, подключенные к системе

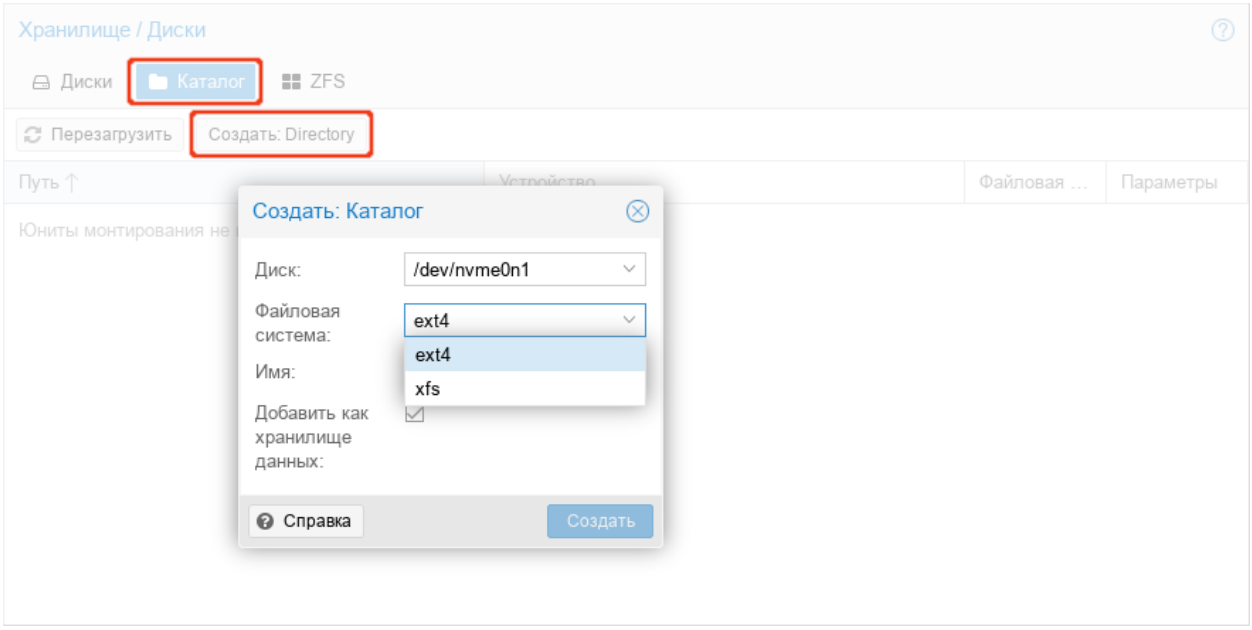

Рис. 404 – PBS. Создание файловой системы на диске

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. Если диск поддерживает S.M.A.R.T. и поддержка S.M.A.R.T. для диска включена, просмотреть данные S.M.A.R.T. можно в веб-интерфейсе или с помощью команды:

```
# proxmox-backup-manager disk smart-attributes sdX
```

9.3.2. Создание хранилища данных

Хранилище данных – это место, где хранятся резервные копии. Текущая реализация PBS использует каталог внутри стандартной файловой системы (ext4, xfs) для хранения данных резервного копирования. Информация о конфигурации хранилищ данных хранится в файле `/etc/proxmox-backup/datastore.cfg`.

Необходимо настроить как минимум одно хранилище данных. Хранилище данных идентифицируется именем и указывает на каталог в файловой системе.

С каждым хранилищем связаны настройки хранения, определяющие, сколько снимков резервных копий для каждого интервала времени (ежечасно, ежедневно, еженедельно, ежемесячно, ежегодно) хранить в этом хранилище.

Для создания хранилища в веб-интерфейсе необходимо нажать на кнопку «Добавить хранилище данных» в боковом меню (в разделе «Хранилище данных»).

В открывшемся окне необходимо указать (рис. 405):

1) «Имя» – название хранилища данных;
2) «Путь к каталогу хранилища» – путь к каталогу, в котором будет создано хранилище данных;
3) «Расписание сборщика мусора» – частота, с которой запускается сборка мусора;
4) «Расписание удаления» – частота, с которой происходит удаление ранее созданных резервных копий;
5) «Параметры удаления» – количество резервных копий, которые необходимо хранить.

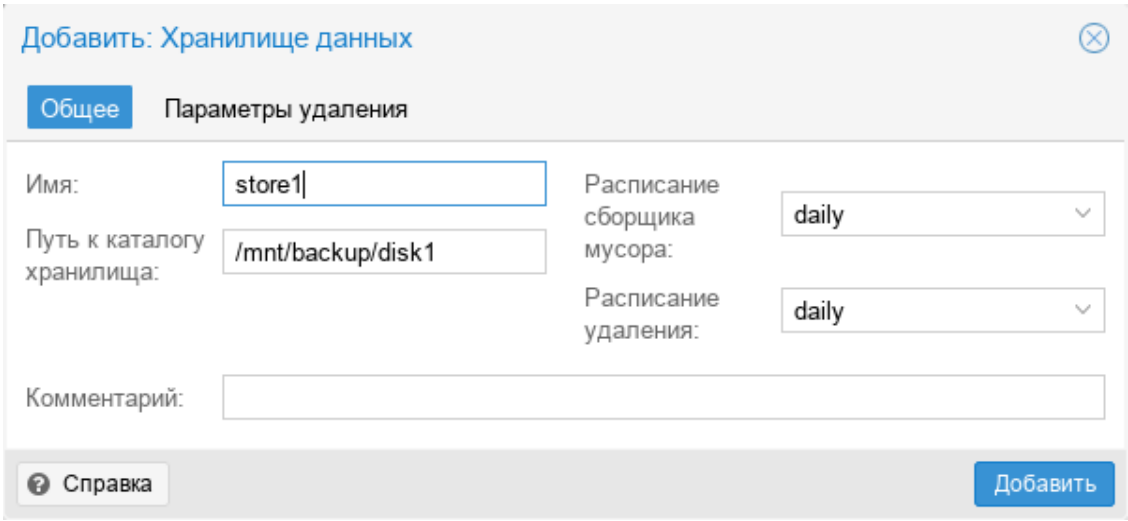

Рис. 405 – PBS. Создание хранилища данных

Создание хранилища данных в командной строке:

```
# proxmox-backup-manager datastore create store1 /mnt/backup/disk1
```

Вывести список существующих хранилищ:

```
# proxmox-backup-manager datastore list
```

После создания хранилища данных в каталоге появляется следующий макет:

```
# ls -arilh /mnt/backup/disk1/
итого 1,1M
2269541 -rw-r--r-- 1 backup backup    0 ноя 15 15:17 .lock
2269540 drwxr-x--- 1 backup backup 1,1M ноя 15 15:17 .chunks
2269538 drwxr-xr-x 3 root   root   4,0K ноя 15 15:17 ..
2269539 drwxr-xr-x 3 backup backup 4,0K ноя 15 15:17 .
```

где:

1) `.lock` – пустой файл, используемый для блокировки процесса;

2) каталог `.chunks` – содержит подкаталоги с именами от `0000` до `ffff`.

В этих каталогах будут храниться фрагментированные данные, полученные после выполнения операции резервного копирования.

9.4. Управление трафиком

Создание и восстановление резервных копий может привести к большому трафику и повлиять на работу других пользователей сети или общих хранилищ.

PBS позволяет ограничить входящий (например, резервное копирование) и исходящий (например, восстановление) сетевой трафик из набора сетей. При этом можно настроить определенные периоды, в которые будут применяться ограничения.

П р и м е ч а н и е . Ограничение скорости не влияет на задания синхронизации. Чтобы ограничить входящий трафик, создаваемый заданием синхронизации, необходимо настроить ограничение скорости входящего трафика для конкретного задания.

Настройка правила управления трафиком в веб-интерфейсе показана на рис. 406.

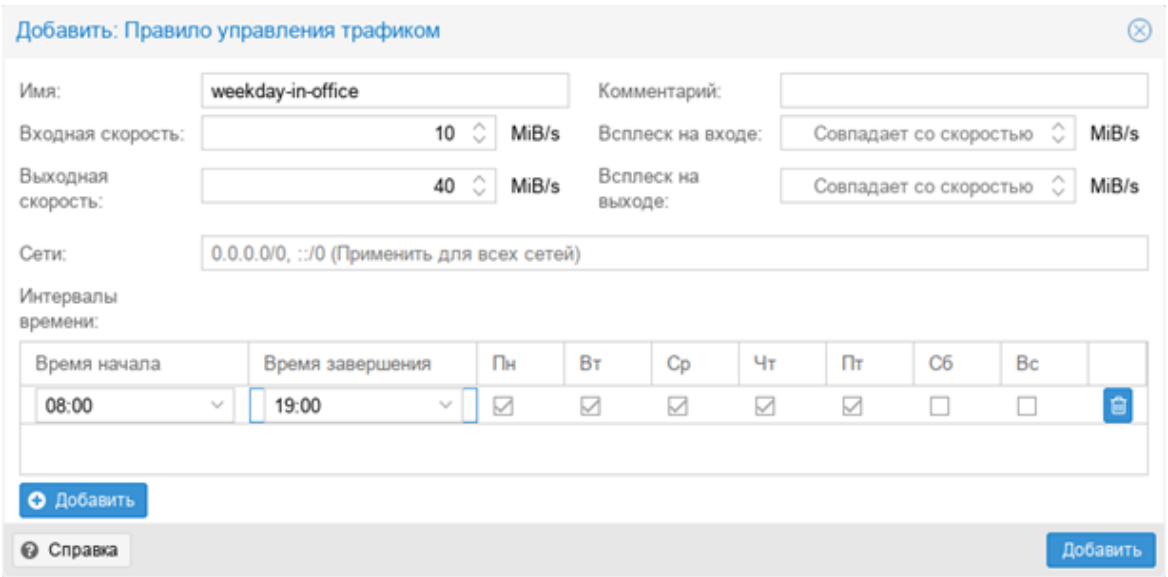

Рис. 406 – PBS. Настройка правила управления трафиком

Управление трафиком в консоли:

1) создать правило управления трафиком для ограничения всех клиентов IPv4 (сеть 0.0.0.0/0) до 100 Мбит/с:

```
# proxmox-backup-manager traffic-control create rule0 \
--network 0.0.0.0/0 \
--rate-in 100MB --rate-out 100MB \
--comment "Default rate limit (100MB/s) for all clients"
```

2) ограничить правило временными рамками:

```
# proxmox-backup-manager traffic-control update rule0 \
--timeframe "mon..fri 8-19"
```

3) вывести список текущих правил:

```
# proxmox-backup-manager traffic-control list
```

4) удалить правило:

```
# proxmox-backup-manager traffic-control remove rule0
```

5) показать состояние (текущую скорость передачи данных) всех настроенных правил:

```
# proxmox-backup-manager traffic-control traffic
```

9.5. Управление пользователями

PVE поддерживает несколько источников аутентификации (рис. 407).

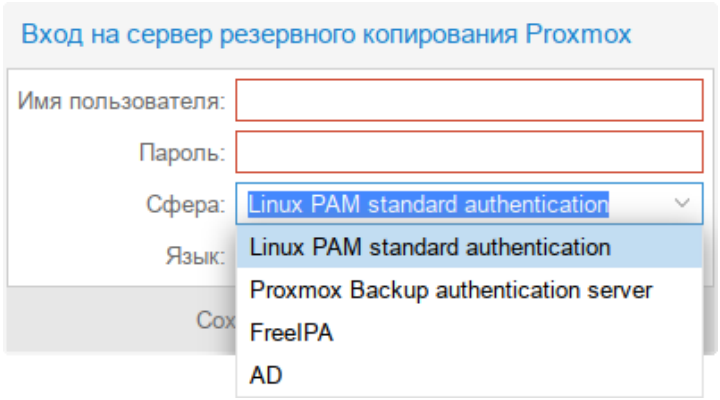

Рис. 407 – Выбор типа аутентификации в веб-интерфейсе

PBS хранит данные пользователей в файле `/etc/proxmox-backup/user.cfg`.

Пользователя часто внутренне идентифицируют по его имени и области аутентификации в форме `<user>@<realm>`.

После установки PBS существует один пользователь `root@pam`, который соответствует суперпользователю ОС. Суперпользователь имеет неограниченные права, поэтому рекомендуется добавить других пользователей с меньшими правами.

9.5.1. Области аутентификации

PBS поддерживает следующие области (методы) аутентификации:

1) «Стандартная аутентификация Linux PAM» («Linux PAM standard authentication») – при использовании этой аутентификации системный пользователь должен существовать (должен быть создан, например, с помощью команды `adduser`). Пользователь аутентифицируется с помощью своего обычного системного пароля;
2) «Сервер аутентификации Proxmox Backup» («Proxmox Backup authentication server») – аутентификация Proxmox Backup Server. Хешированные пароли хранятся в файле `/etc/proxmox-backup/shadow.json`;
3) «Сервер LDAP» – позволяет использовать внешний LDAP-сервер для аутентификации пользователей (например, OpenLDAP);
4) «Сервер OpenID Connect» – уровень идентификации поверх протокола OATH 2.0. Позволяет аутентифицировать пользователей на основе аутентификации, выполняемой внешним сервером авторизации.

9.5.1.1. Стандартная аутентификация Linux PAM

При использовании «Стандартная аутентификация Linux PAM», системный пользователь должен существовать (должен быть создан, например, с помощью команды `adduser`) на всех узлах, на которых пользователю разрешено войти в систему.

Область Linux PAM создается по умолчанию и не может быть удалена.

9.5.1.2. Сервер аутентификации Proxmox Backup

Область «Сервер аутентификации Proxmox Backup» представляет собой хранилище паролей в стиле Unix (`/etc/proxmox-backup/shadow.json`). Пароль шифруется с использованием метода хеширования SHA-256.

Область создается по умолчанию.

Для добавления пользователя в веб-интерфейсе следует в разделе «Конфигурация» → «Управление доступом» перейти на вкладку «Управление пользователями» и нажать на кнопку «Добавить» (рис. 408).

Примеры использования командной строки для управления пользователями PBS:

1) просмотреть список пользователей:

```
# proxmox-backup-manager user list
```

2) создать пользователя:

```
# proxmox-backup-manager user create backup_u@pbs --email backup_u@test.alt
```

3) обновить или изменить любые свойства пользователя:

```
# proxmox-backup-manager user update backup_u@pbs --firstname Дмитрий --lastname Иванов
```

4) отключить учетную запись пользователя:

```
# proxmox-backup-manager user update backup_u@pbs --enable 0
```

5) удалить учетную запись пользователя:

```
# proxmox-backup-manager user remove backup_u@pbs
```

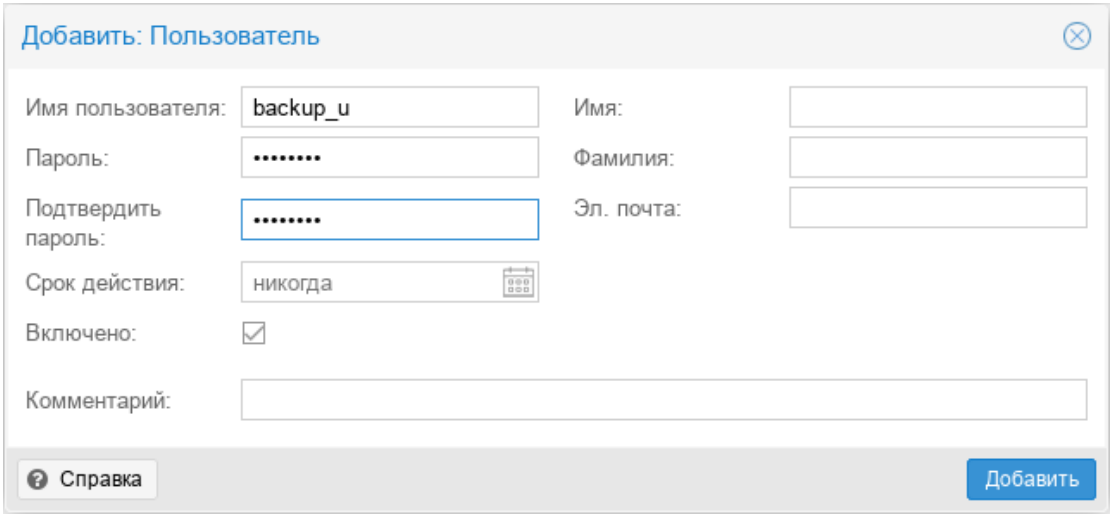

Рис. 408 – PBS. Добавление пользователя

9.5.1.3. LDAP аутентификация (FreeIPA)

В данном разделе приведен пример настройки LDAP аутентификации для аутентификации на сервере FreeIPA. В примере используются следующие исходные данные:

1) ipa.example.test, 192.168.0.113 – сервер FreeIPA;
2) admin@example.test – учетная запись с правами чтения LDAP;
3) 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 «Конфигурация» → «Управление доступом» → «Сферы» нажать на кнопку «Добавить» → «Сервер LDAP» (рис. 409);

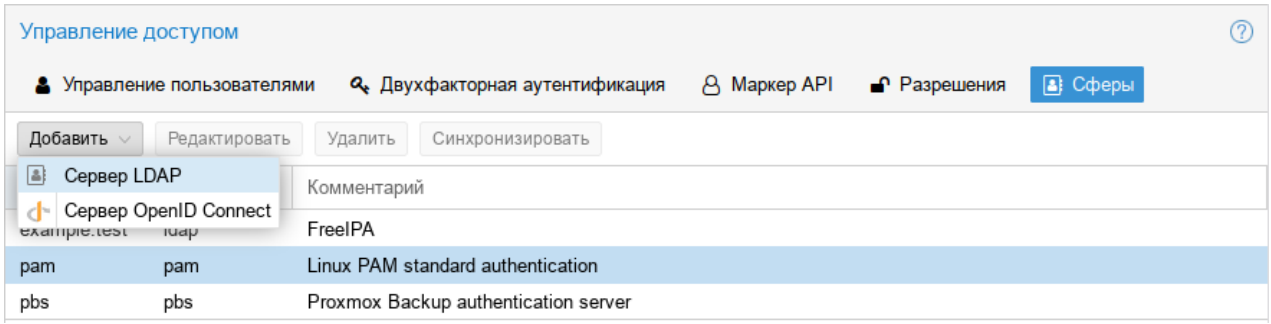

Рис. 409 – Создать область аутентификации LDAP

2) на вкладке «Общее» (рис. 410) указать следующие данные:
   - «Сфера» – идентификатор области;
   - «Имя основного домена» (base_dn) – каталог, в котором выполняется поиск пользователей (`cn=accounts, dc=example, dc=test`);
   - «Имя пользовательского атрибута» (user_attr) – атрибут LDAP, содержащий имя пользователя, с которым пользователи будут входить в систему (uid);
   - «Bind Domain Name» – имя пользователя (`uid=admin, cn=users, cn=accounts, dc=example, dc=test`);
   - «Пароль (bind)» – пароль пользователя;

- «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);

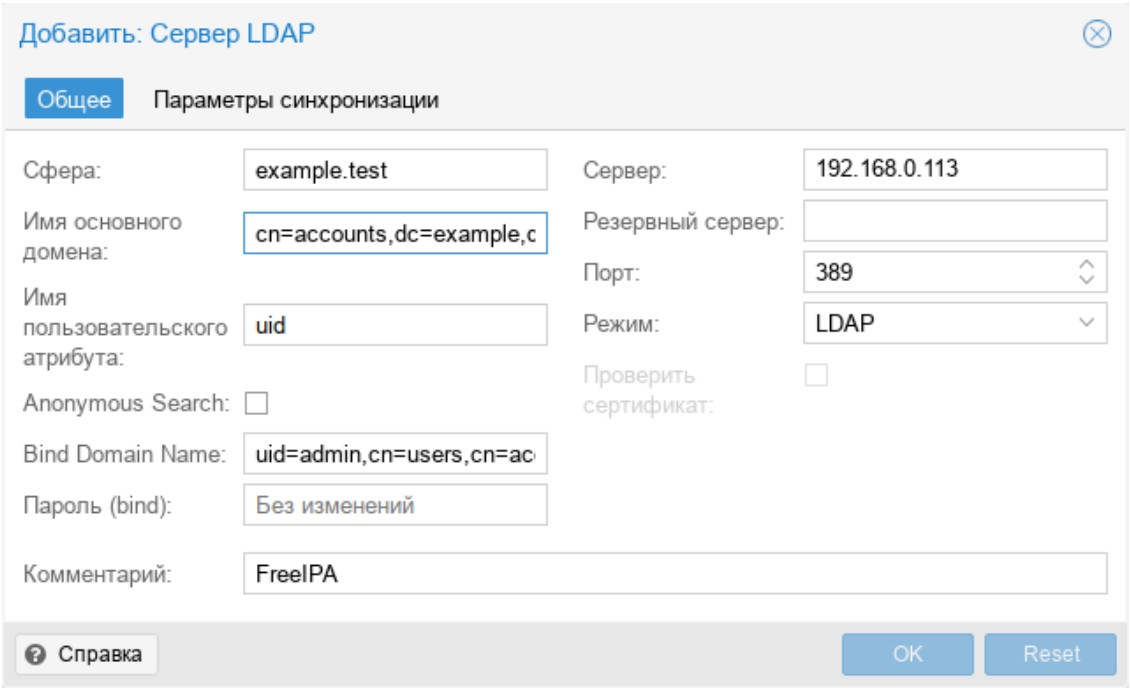

Рис. 410 – Настройка LDAP аутентификации FreeIPA (вкладка «Общее»)

3) на вкладке «Параметры синхронизации» (рис. 411) заполнить следующие поля (в скобках указаны значения, используемые в данном примере):
   - «Атрибут имени пользователя» (опционально) – атрибут LDAP, содержащий имя пользователя (givenname);
   - «Атрибут фамилии пользователя» (опционально) – атрибут LDAP, содержащий фамилию пользователя (sn);
   - «Атрибут электронной почты» (опционально) – атрибут LDAP, содержащий электронную почту пользователя (mail);
   - «Классы пользователей» – класс пользователей LDAP (inetOrgPerson);
   - «Фильтр пользователей» – фильтр пользователей (`memberOf=cn=pve, cn=groups, cn=accounts, dc=example, dc=test`);

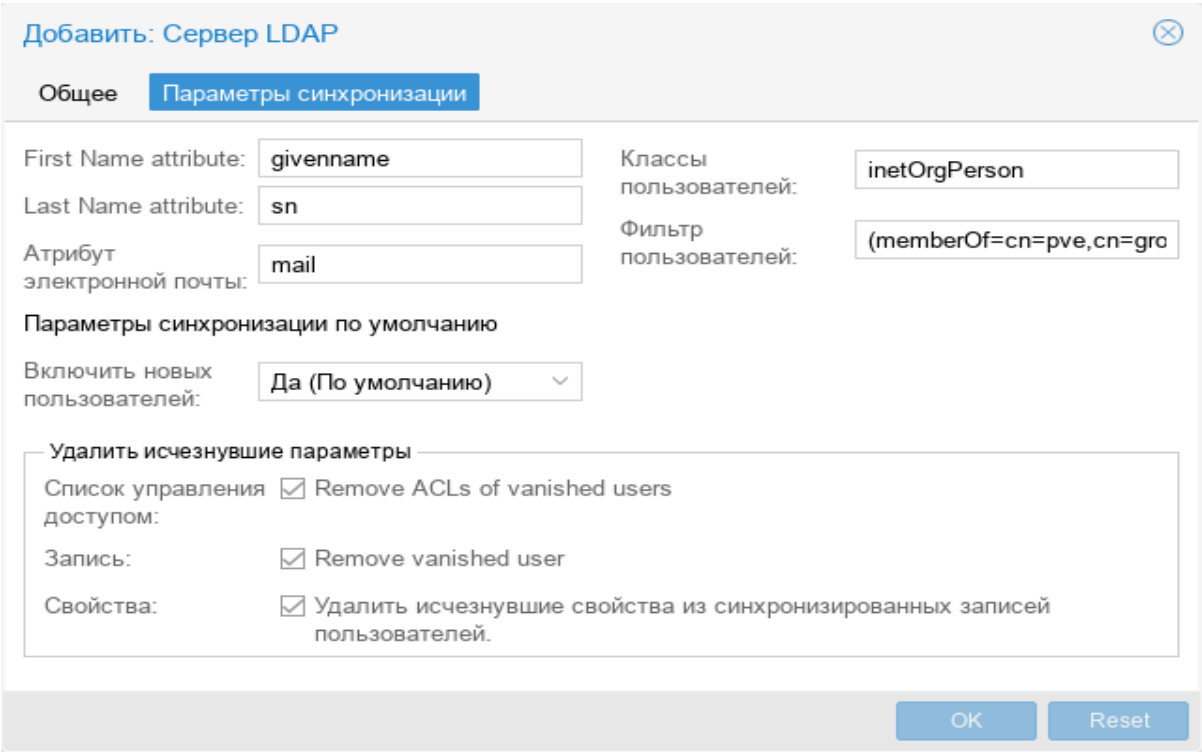

Рис. 411 – Настройка LDAP аутентификации FreeIPA (вкладка «Параметры синхронизации»)

4) нажать на кнопку «OK»;

5) выбрать добавленную область и нажать на кнопку «Синхронизировать» (рис. 412);

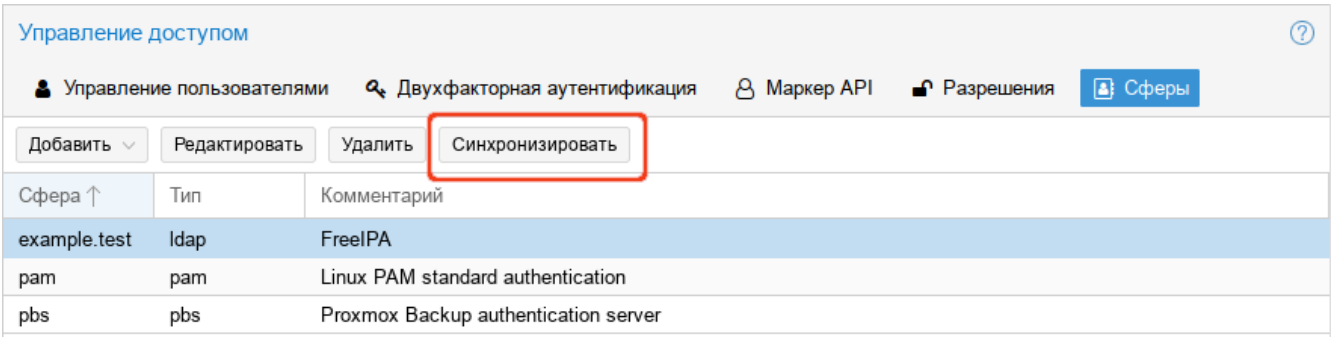

Рис. 412 – Кнопка «Синхронизировать»

6) указать, если необходимо, параметры синхронизации и нажать на кнопку «Синхронизировать» (рис. 413).

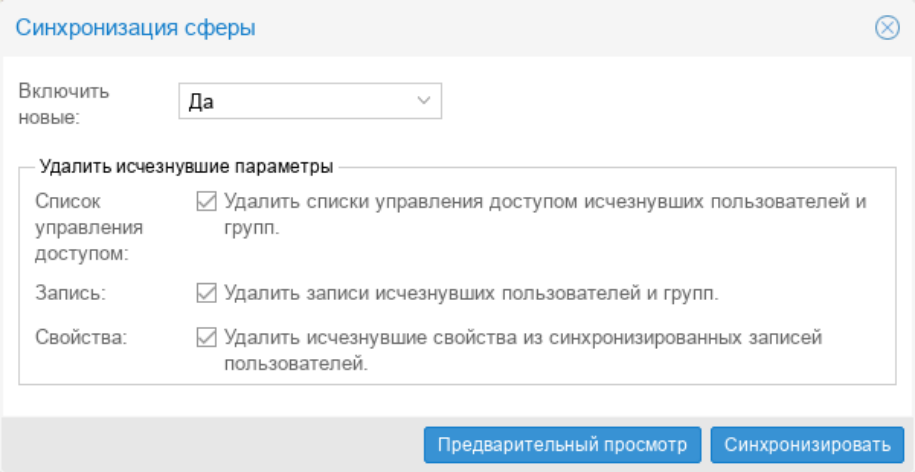

Рис. 413 – Параметры синхронизации области аутентификации

П р и м е ч а н и е . Команда синхронизации пользователей:

```
# proxmox-backup-manager ldap sync example.test
```

Для автоматической синхронизации пользователей можно добавить команду синхронизации в планировщик задач.

9.5.1.4. LDAP аутентификация (AD)

В данном разделе приведен пример настройки аутентификации на сервере AD. В примере используются следующие исходные данные:

1) dc.test.alt, 192.168.0.122 – сервер AD;
2) administrator@test.alt – учетная запись администратора (для большей безопасности рекомендуется создать отдельную учетную запись с доступом только для чтения к объектам домена и не использовать учетную запись администратора);
3) 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 «Конфигурация» → «Управление доступом» → «Сферы» нажать на кнопку «Добавить» → «Сервер LDAP» (см. рис. 410);
2) на вкладке «Общее» (рис. 414) указать следующие данные:
   - «Сфера» – идентификатор области;

- «Имя основного домена» (base_dn) – каталог, в котором выполняется поиск пользователей (`dc=test, dc=alt`);
- «Имя пользовательского атрибута» (user_attr) – атрибут LDAP, содержащий имя пользователя, с которым пользователи будут входить в систему (sAMAccountName);
- «По умолчанию» – установить область в качестве области по умолчанию для входа в систему;
- «Bind Domain Name» – имя пользователя (`cn=Administrator, cn=Users, dc=test, dc=alt`);
- «Пароль (bind)» – пароль пользователя;
- «Сервер» – IP-адрес или имя AD-сервера (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);

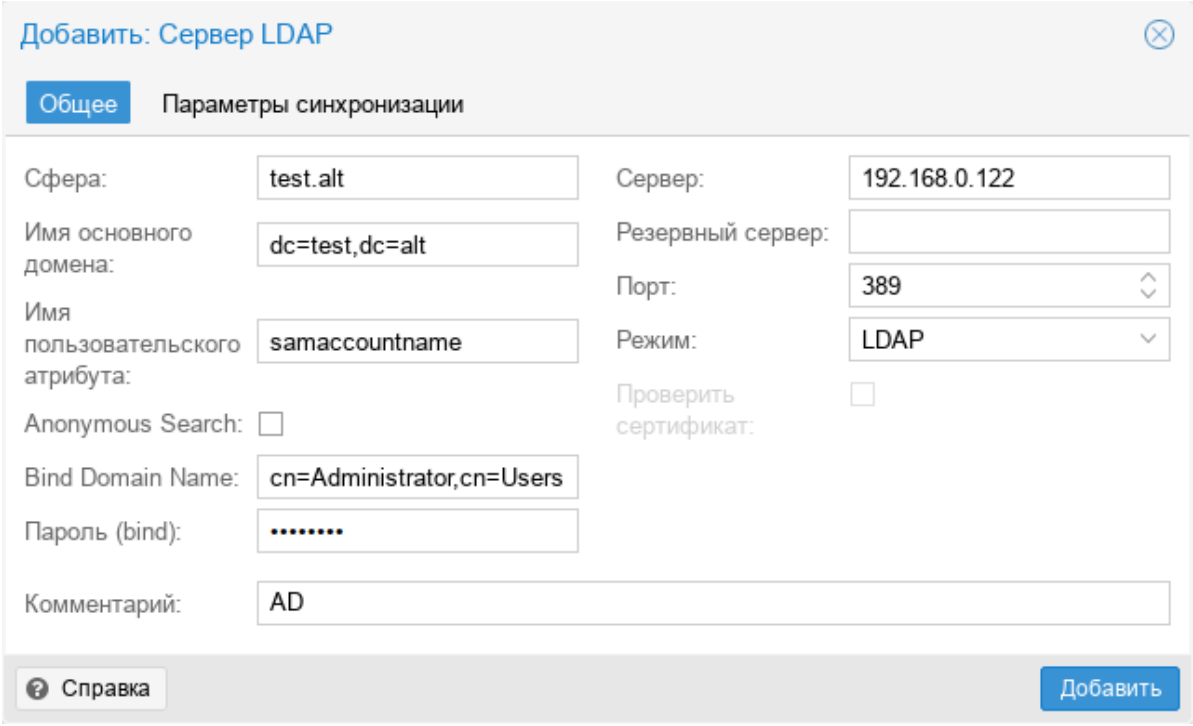

Рис. 414 – Настройка LDAP аутентификации AD (вкладка «Общее»)

3) на вкладке «Параметры синхронизации» (рис. 415) заполнить следующие поля (в скобках указаны значения, используемые в данном примере):
   - «Атрибут имени пользователя» (опционально) – атрибут LDAP, содержащий имя пользователя (givenname);
   - «Атрибут фамилии пользователя» (опционально) – атрибут LDAP, содержащий фамилию пользователя (sn);
   - «Атрибут электронной почты» (опционально) – атрибут LDAP, содержащий электронную почту пользователя (mail);
   - «Классы пользователей» – класс пользователей LDAP (user);
   - «Фильтр пользователей» – фильтр пользователей (`(&(objectclass=user) (samaccountname=*) (MemberOf=CN=UDS, cn=Users, dc=TEST, dc=ALT))`);
4) нажать на кнопку «OK»;
5) выбрать добавленную область и нажать на кнопку «Синхронизировать»;
6) указать, если необходимо, параметры синхронизации и нажать на кнопку «Синхронизировать» (см. рис. 414).

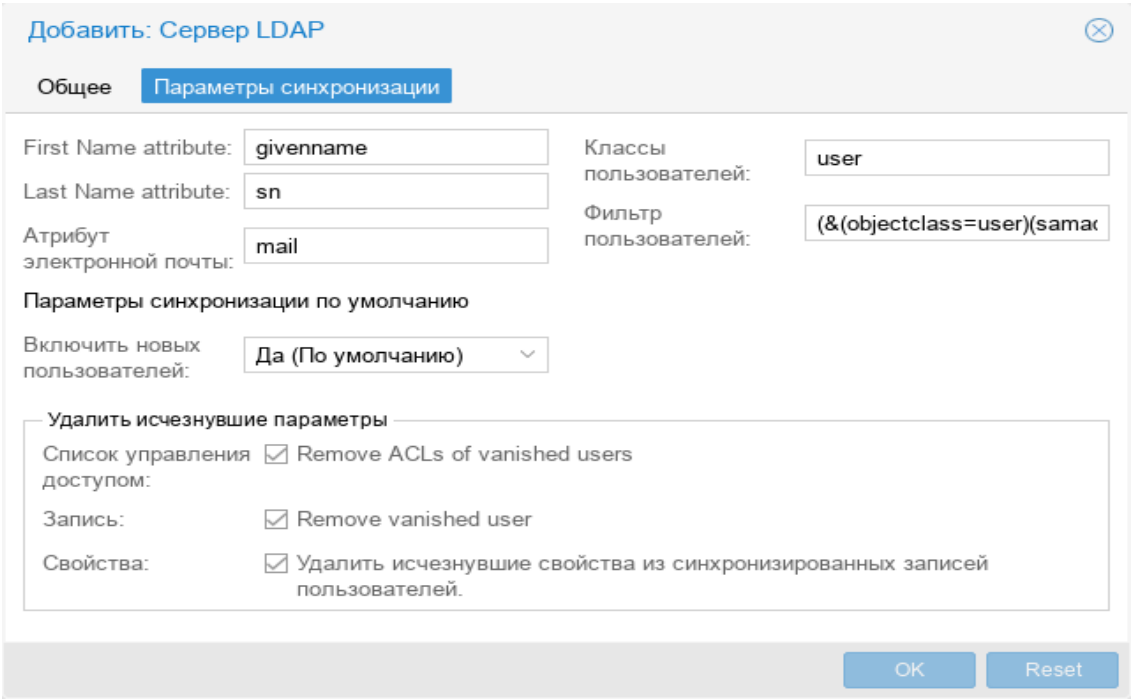

Рис. 415 – Настройка LDAP аутентификации AD
(вкладка «Параметры синхронизации»)

В результате синхронизации пользователи PBS будут синхронизированы с сервером AD. Сведения о пользователях можно проверить на вкладке «Управление пользователями».

Настроить разрешения для пользователя на вкладке «Разрешения».

Примечание. Команда синхронизации пользователей и групп:

```
# proxmox-backup-manager ldap sync test.alt
```

Для автоматической синхронизации пользователей и групп можно добавить команду синхронизации в планировщик задач.

9.5.2. API-токены

Любой аутентифицированный пользователь может генерировать API-токены, которые, в свою очередь, можно использовать для настройки клиентов резервного копирования вместо прямого указания имени пользователя и пароля.

Назначение API-токенов:

1) простой отзыв в случае компрометации клиента;
2) возможность ограничить разрешения для каждого клиента/токена в рамках разрешений пользователей.

Добавление API-токена в веб-интерфейсе показано на рис. 416.

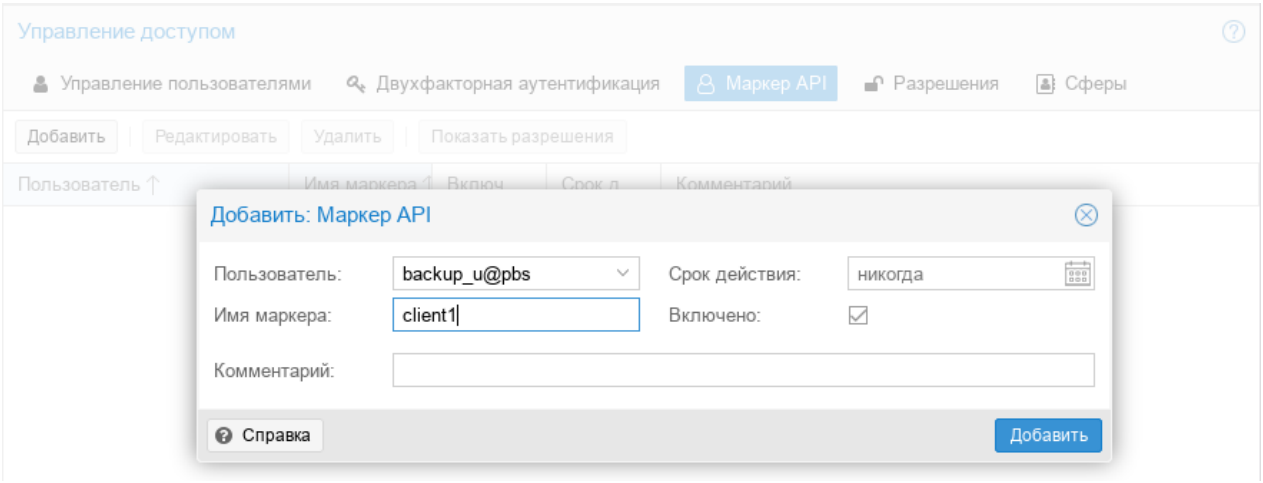

Рис. 416 – PBS. Добавление API-токена

API-токен состоит из двух частей (рис. 417):

1) идентификатор (Token ID), который состоит из имени пользователя, области и имени токена (`user@realm!имя токена`);
2) секретное значение.

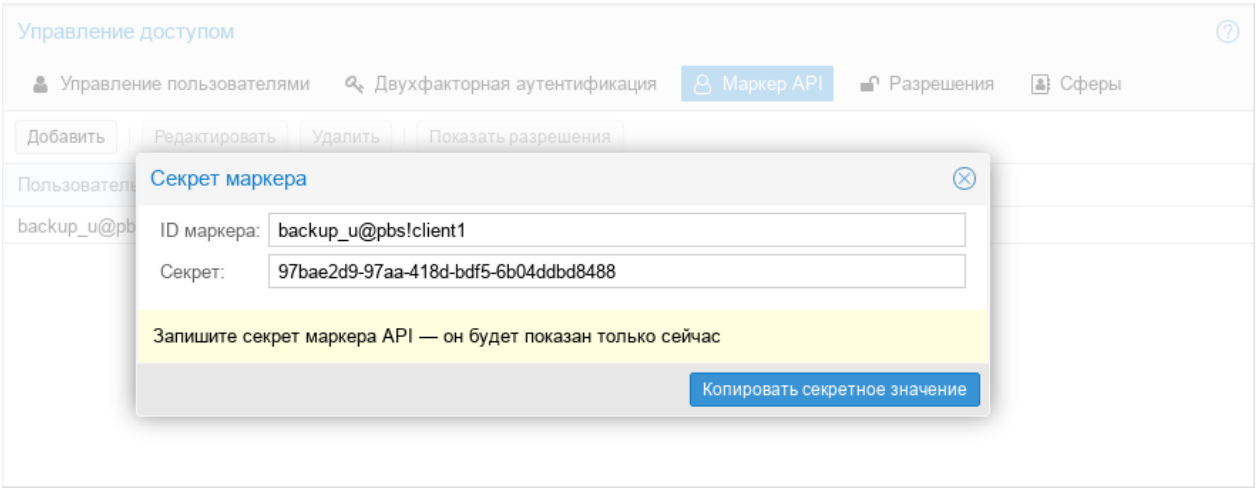

Рис. 417 – PBS. API-токен

Обе части должны быть предоставлены клиенту вместо идентификатора пользователя и его пароля.

Примечание. Отображаемое секретное значение необходимо сохранить, так как после создания токена его нельзя будет отобразить снова.

Создание API-токена в консоли:

```
# proxmox-backup-manager user generate-token backup_u@pbs client1
Result: {
  "tokenid": "backup_u@pbs!client1",
  "value": "ff13e5e0-30df-4a70-99f1-c62b13803769"
}
```

9.5.3. Управление доступом

По умолчанию новые пользователи и API-токены не имеют никаких разрешений. Добавить разрешения можно, назначив роли пользователям/токенам для определенных объектов, таких как хранилища данных или удаленные устройства.

Роль – это список привилегий. В PBS предопределен ряд ролей:

1) NoAccess – нет привилегий (используется для запрета доступа);
2) Admin – все привилегии;
3) Audit – доступ только для чтения;
4) DatastoreAdmin – все привилегии для хранилищ данных;
5) DatastoreAudit – просмотр настроек хранилищ и их содержимых без возможности чтения фактических данных;

6) DatastoreBackup – создание и восстановление собственных резервных копий;
7) DatastorePowerUser – создание, восстановление и удаление собственных резервных копий;
8) DatastoreReader – просмотр содержимого хранилища, восстановление данных;
9) RemoteAdmin – все привилегии для удаленных PBS;
10) RemoteAudit – просмотр настроек удаленных PBS;
11) RemoteSyncOperator – чтение данных с удаленных PBS;
12) TapeAdmin – все привилегии для резервного копирования на ленту;
13) TapeAudit – просмотр настроек, показателей и состояния ленты;
14) TapeOperator – создание и восстановление резервных копий на ленте без возможности изменения конфигурации;
15) TapeReader – чтение и проверка конфигурации ленты.

PBS использует систему управления разрешениями на основе ролей и путей. Запись в таблице разрешений позволяет пользователю играть определенную роль при доступе к объекту или пути. Такое правило доступа может быть представлено как тройка (путь, пользователь, роль) или (путь, API-токен, роль), причем роль содержит набор разрешенных действий, а путь представляет цель этих действий.

Информация о правах доступа хранится в файле `/etc/proxmox-backup/acl.cfg`. Файл содержит 5 полей, разделенных двоеточием «:»:

```
acl:1:/datastore:backup_u@pbs!client1:DatastoreAdmin
```

В каждом поле представлены следующие данные:

1) идентификатор acl;
2) 1 или 0 – включено или отключено;
3) объект, на который установлено разрешение;
4) пользователи/токены, для которых установлено разрешение;
5) устанавливаемая роль.

Добавление разрешения можно выполнить в веб-интерфейсе («Конфигурация» → «Управление доступом» вкладка «Разрешения») (рис. 418).

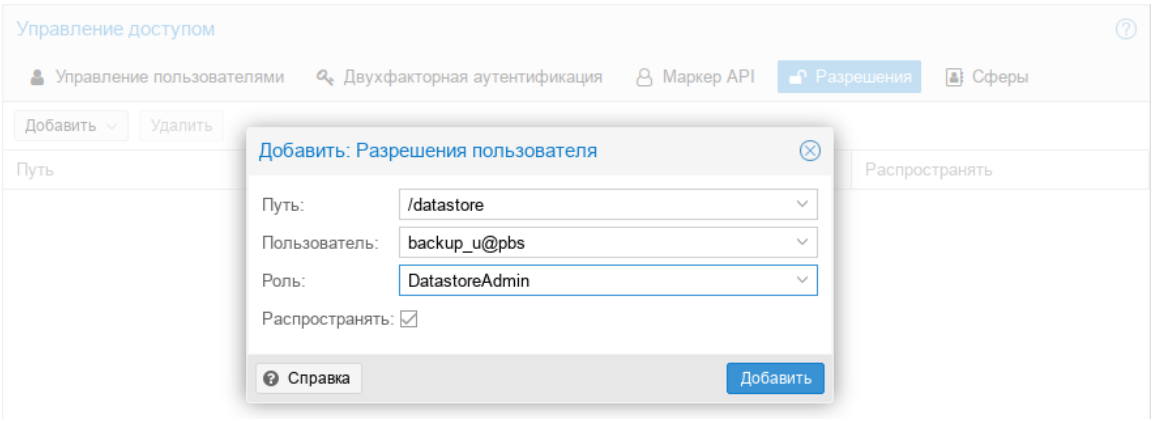

Рис. 418 – PBS. Добавление разрешения

Управление разрешениями в консоли:

1) добавить разрешение (добавить пользователя `backup_u@pbs` в качестве администратора хранилища данных для хранилища данных `store1`, расположенного в `/mnt/backup/disk1/store1`):

```
# proxmox-backup-manager acl update /datastore/store1
DatastoreAdmin --auth-id backup_u@pbs
```

2) вывести список разрешений:

```
# proxmox-backup-manager acl list
```

3) отобразить действующий набор разрешений пользователя или API-токена:

```
# proxmox-backup-manager user permissions backup_u@pbs --path
/datastore/store1
   Privileges with (*) have the propagate flag set

   Path: /datastore/store1
   - Datastore.Audit (*)
   - Datastore.Backup (*)
   - Datastore.Modify (*)
   - Datastore.Prune (*)
   - Datastore.Read (*)
   - Datastore.Verify (*)
```

Примечание. Для токенов требуются собственные записи ACL. Токены не могут делать больше, чем их соответствующий пользователь.

9.5.4. Двухфакторная аутентификация

Примечание. Двухфакторная аутентификация реализована только для веб-интерфейса.

PBS поддерживает три метода двухфакторной аутентификации (рис. 419):

1) «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);
2) «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;
3) «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.

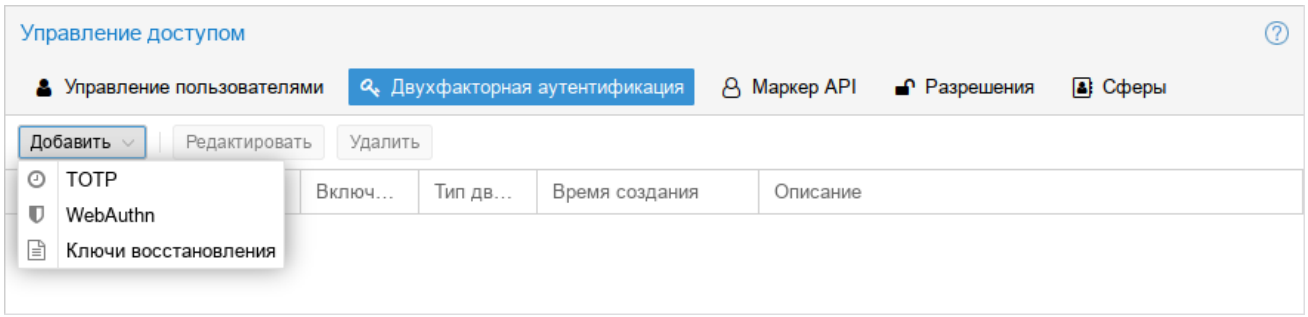

Рис. 419 – PBS. Двухфакторная аутентификация

Процедура добавления аутентификации «TOTP» показана на рис. 420.

Рис. 420 – PBS. Настройка аутентификации TOTP

При аутентификации пользователя будет запрашиваться второй фактор, показанный на рис. 421.

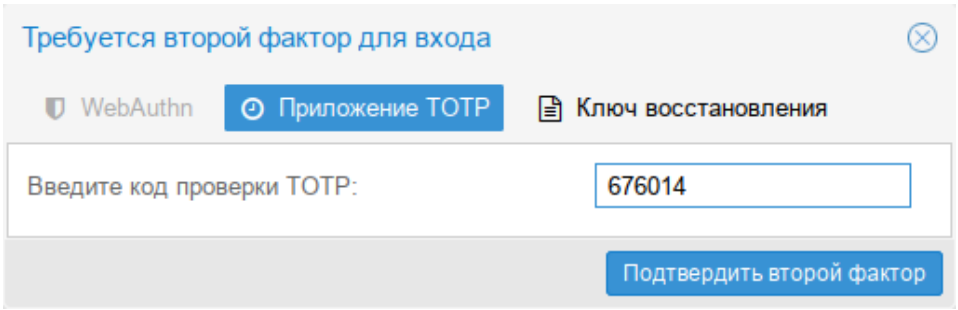

Рис. 421 – Запрос второго фактора (TOTP) при аутентификации пользователя в веб-интерфейсе

При настройке аутентификации «Ключи восстановления» необходимо создать набор ключей (рис. 422).

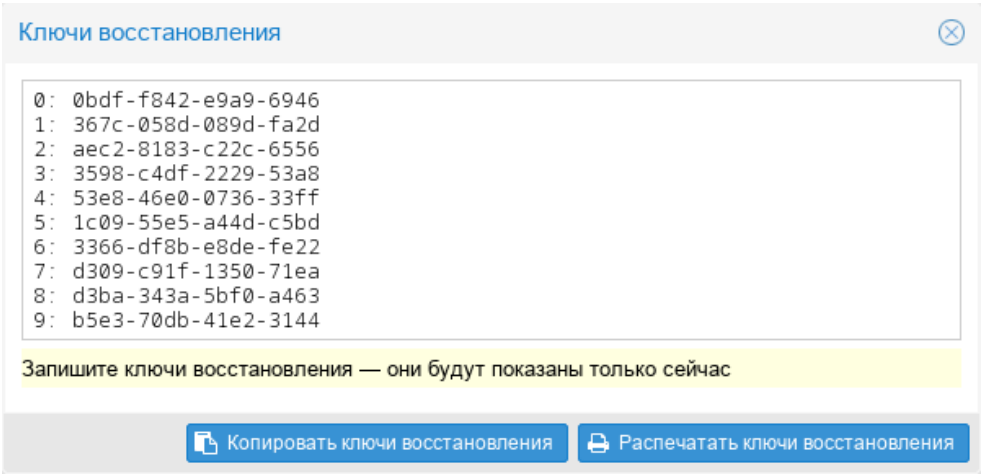

Рис. 422 – PBS. Настройка аутентификации «Ключи восстановления»

При аутентификации пользователя будет запрашиваться второй фактор (рис. 423).

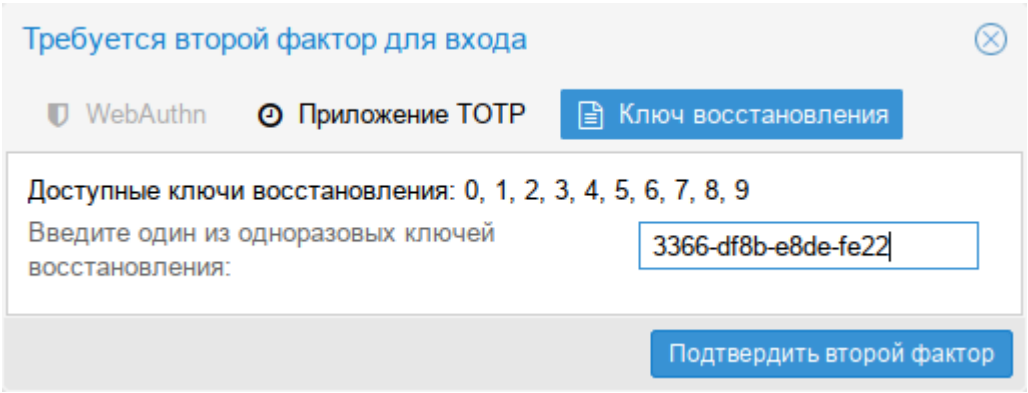

Рис. 423 – Запрос второго фактора («Ключи восстановления») при аутентификации пользователя в веб-интерфейсе

9.6. Управление удаленными PBS

Хранилища данных с удаленного сервера можно синхронизировать с локальным хранилищем с помощью задания синхронизации.

Информация о конфигурации удаленных PBS хранится в файле `/etc/proxmox-backup/remote.cfg`.

Для добавления удаленного PBS в веб-интерфейсе следует перейти в раздел «Конфигурация» → «Удаленные хранилища» и нажать на кнопку «Добавить» (рис. 424).

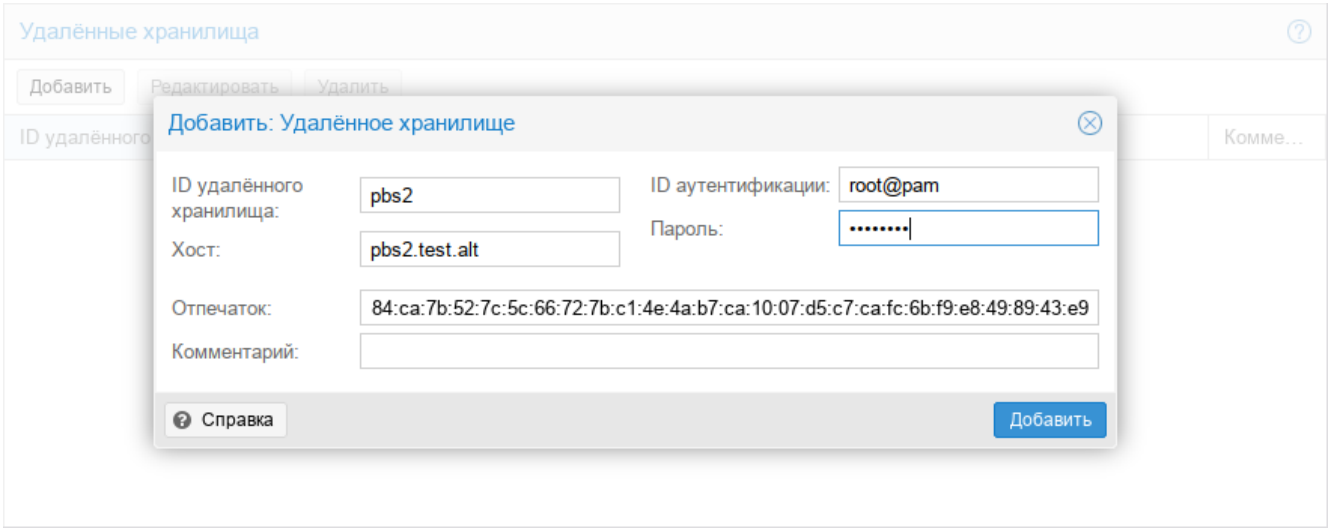

Рис. 424 – PBS. Добавление удаленного PBS

Примечание. Отпечаток TLS-сертификата можно получить в веб-интерфейсе удаленного PBS (рис. 425).

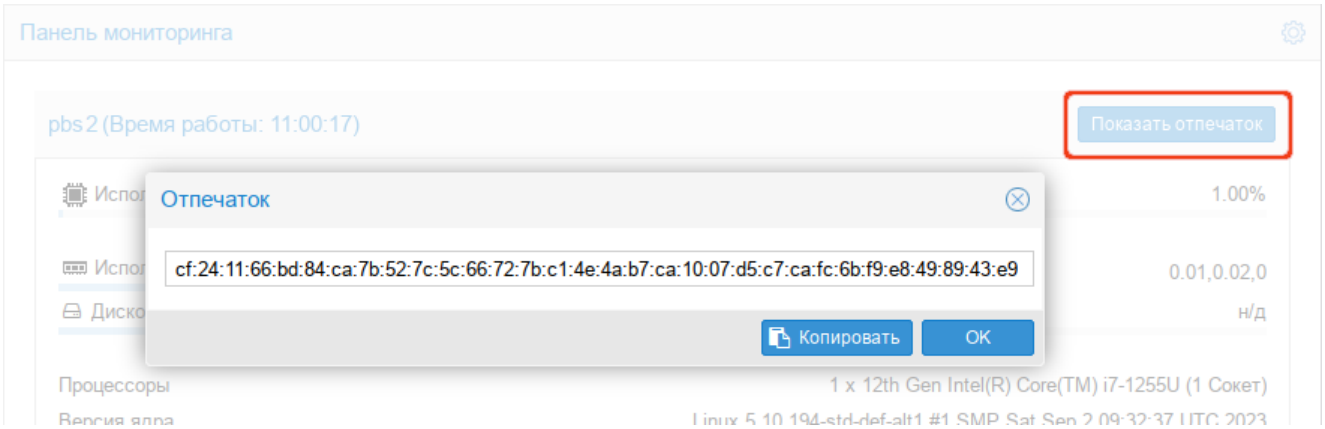

Рис. 425 – PBS. Отпечаток TLS-сертификата

Получить отпечаток в командной строке:

```
# proxmox-backup-manager cert info | grep Fingerprint
```

Для настройки задачи синхронизации, необходимо в разделе «Хранилище данных» перейти на вкладку «Задания синхронизации» и нажать на кнопку «Добавить» (рис. 426).

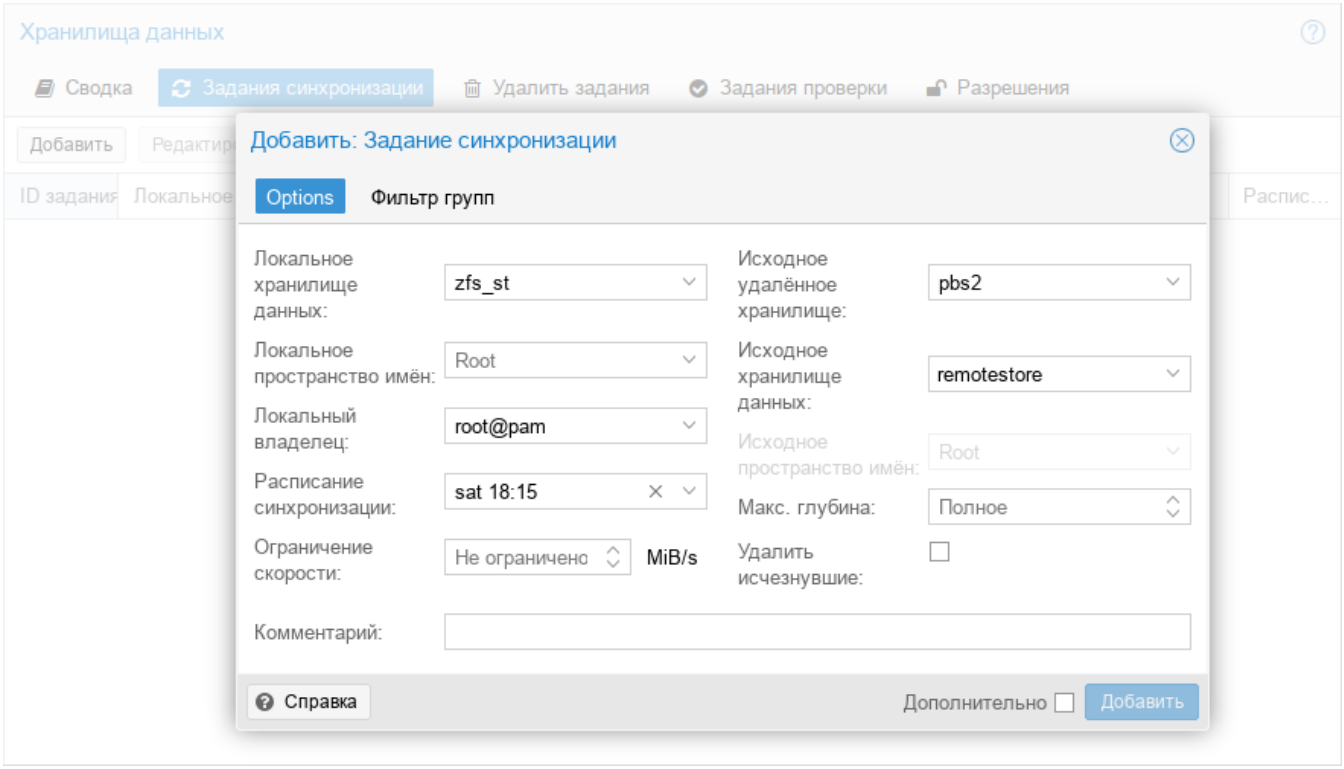

Рис. 426 – PBS. Добавление задачи синхронизации

После того, как задание синхронизации создано, оно будет запускаться по заданному расписанию, а также его можно запустить вручную из веб-интерфейса (кнопка «Запустить сейчас»).

9.7. Клиент резервного копирования

Клиент резервного копирования использует следующий формат для указания репозитория хранилища данных на сервере резервного копирования (где имя пользователя указывается в виде `user@realm`):

```
[[пользователь@]сервер[:порт]:]datastore
```

Значение по умолчанию для пользователя – `root@pam`. Если сервер не указан, используется `localhost`. Примеры репозиториев показаны в таблице 28.

Указать репозиторий можно, передав его в параметре `--repository`, или установив переменную окружения `PBS_REPOSITORY`, например:

```
# export PBS_REPOSITORY=pbs.test.alt:store1
```

Т а б л и ц а 28 – Примеры репозиториев

Пример	Пользователь	Хост:Порт	Хранилище
store1	root@pam	localhost:8007	store1
pbs.test.alt:store1	root@pam	pbs.test.alt:8007	store1
backup_u@pbs@pbs.test.alt:store1	backup_u@pbs	pbs.test.alt:8007	store1
backup_u@pbs!client1@pbs.test.alt:store1	backup_u@pbs!client1	pbs.test.alt:8007	store1
192.168.0.123:1234:store1	root@pam	192.168.0.123:1234	store1

9.7.1. Создание резервной копии

В этом разделе рассмотрено, как создать резервную копию внутри машины (физического хоста, ВМ или контейнера). Такие резервные копии могут содержать архивы файлов и образов.

Создать резервную копию домашнего каталога пользователя `user` (будет создан архив `user.pxar`):

```
$ proxmox-backup-client backup user.pxar:/home/user/ --repository
pbs.test.alt:store1
Starting backup: host/host-197/2023-09-17T13:12:05Z
Client name: host-01
Starting backup protocol: Sun Sep 17 15:12:05 2023
No previous manifest available.
Upload directory '/home/user/' to 'pbs.test.alt:store1' as
user.pxar.didx
user.pxar: had to backup 667.04 MiB of 667.04 MiB (compressed
190.182 MiB) in 26.22s
user.pxar: average backup speed: 25.436 MiB/s
Uploaded backup catalog (109.948 KiB)
Duration: 26.36s
End Time: Sun Sep 17 15:12:12 2023
```

Команда `proxmox-backup-client backup` принимает список параметров резервного копирования, включая имя архива на сервере, тип архива и источник архива на клиенте, в формате:

```
<archive-name>.<type>:<source-path>
```

Тип архива `.pxar` используется для файловых архивов, а `.img` – для образов блочных устройств.

Команда создания резервной копии блочного устройства:

```
$ proxmox-backup-client backup mydata.img:/dev/mylvm/mydata
```

9.7.2. Создание зашифрованной резервной копии

PBS поддерживает шифрование на стороне клиента с помощью AES-256 в режиме GCM.

Создание ключа шифрования:

```
$ proxmox-backup-client key create my-backup.key
Encryption Key Password: ******
Verify Password: ******
```

Создание зашифрованной резервной копии:

```
$ proxmox-backup-client backup user_s.pxar:/home/user/ --repository
pbs.test.alt:store1 --keyfile ./my-backup.key
Password for "root@pam": ***
Starting backup: host/host-197/2023-09-17T12:17:16Z
Client name: host-01
Starting backup protocol: Sun Sep 17 14:17:19 2023
Using encryption key from './my-backup.key'..
Encryption Key Password: ******
Encryption key fingerprint: 0d:aa:4f:9b:ef:63:31:47
fingerprint:
cf:24:11:66:bd:84:ca:7b:52:7c:5c:66:72:7b:c1:4e:4a:b7:ca:10:07:d5:c7
:ca:fc:6b:f9:e8:49:89:43:e9
Are you sure you want to continue connecting? (y/n): y
Downloading previous manifest (Sun Sep 17 14:14:27 2023)
Upload directory '/home/user/' to '192.168.0.123:store1' as
user_s.pxar.didx
user_s.pxar: had to backup 667.04 MiB of 667.04 MiB (compressed
190.028 MiB) in 21.16s
user_s.pxar: average backup speed: 31.518 MiB/s
Uploaded backup catalog (109.971 KiB)
Duration: 31.17s
End Time: Sun Sep 17 14:17:31 2023
```

Содержимое хранилища `store1` показано на рис. 427.

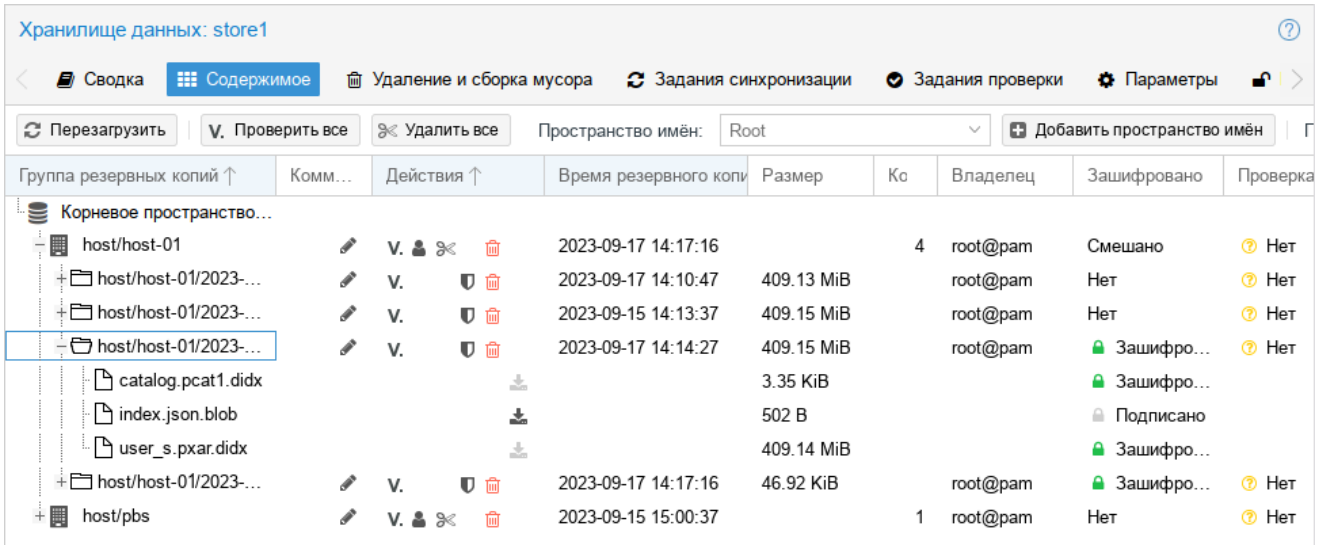

Рис. 427 – PBS. Содержимое хранилища `store1`

9.7.3. Восстановление данных

Просмотреть список всех снимков на сервере:

```
$ proxmox-backup-client snapshot list --repository pbs.test.alt:store1
```

Просмотреть содержимое снимка:

```
$ proxmox-backup-client catalog dump host/host-01/2022-04-28T12:27:01Z --repository pbs.test.alt:store1
```

Команда восстановления архива из резервной копии:

```
proxmox-backup-client restore <снимок> <имя-архива> <целевой-путь> [ОПЦИИ]
```

Восстановить архив `user.pxar` в каталог `/home/user/restore`:

```
$ proxmox-backup-client restore host/host-01/2023-09-17T12:10:47Z user.pxar /home/user/restore --repository pbs.test.alt:store1
```

Получить содержимое любого архива можно, восстановив файл `index.json` в репозитории по целевому пути «-». При этом содержимое архива будет выведено на стандартный вывод:

```
$ proxmox-backup-client restore host/host-01/2023-09-17T12:10:47Z index.json - --repository pbs.test.alt:store1
```

Если необходимо восстановить несколько отдельных файлов, можно использовать интерактивную оболочку восстановления:

```
$ proxmox-backup-client catalog shell host/host-01/2023-09-17T12:10:47Z user.pxar --repository pbs.test.alt:store1
Starting interactive shell
pxar:/ > ls
…
```

Пример поиска в содержимом архива и восстановление данных:

```
pxar:/ > find  *.txt --select
/test/connection_trace.txt
/Рабочий стол/1.txt
pxar:/ > list-selected
/test/connection_trace.txt
/Рабочий стол/1.txt
pxar:/ > restore-selected /home/user/restore/
pxar:/ > restore /home/user/conf/ --pattern *.conf
pxar:/ > exit
```

где:

1) `find *.txt --select` – найти все файлы с расширением `.txt` и добавить соответствующие шаблоны в список для последующего восстановления;
2) `list-selected` – вывести шаблоны на экран;
3) `restore-selected /home/user/restore/` – восстановить все файлы в архиве, соответствующие шаблонам в `/home/user/restore/` на локальном хосте;
4) `restore /home/user/conf/ --pattern *.conf` – восстановить все файлы с расширением `.conf` в `/home/user/conf/` на локальном хосте.

9.7.4. Вход и выход

При первой попытке получить доступ к серверу с использованием команды `proxmox-backup-client`, потребуется ввести пароль пользователя. Сервер проверяет учетные данные и отправляет билет, действительный в течение двух часов. Клиент использует этот билет для последующих запросов к этому серверу.

Можно вручную инициировать вход/выход. Команда входа:

```
$ proxmox-backup-client login --repository pbs.test.alt:store1
Password for "root@pam": ******
```

Удалить билет:

```
$ proxmox-backup-client logout --repository pbs.test.alt:store1
```

9.8. Интеграция с PVE

PBS можно интегрировать в автономную или кластерную установку PVE, добавив его в качестве хранилища (рис. 428).

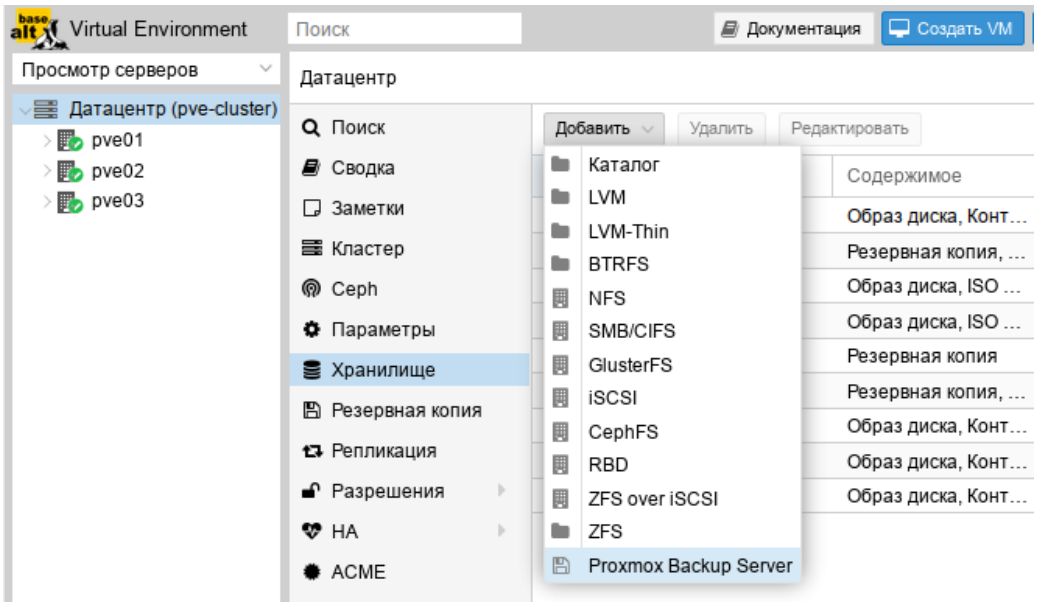

Рис. 428 – PVE. Добавление хранилища Proxmox Backup Server

Диалог создания хранилища `pbs_backup` типа «Proxmox Backup Server» для хранения резервных копий представлен на рис. 429.

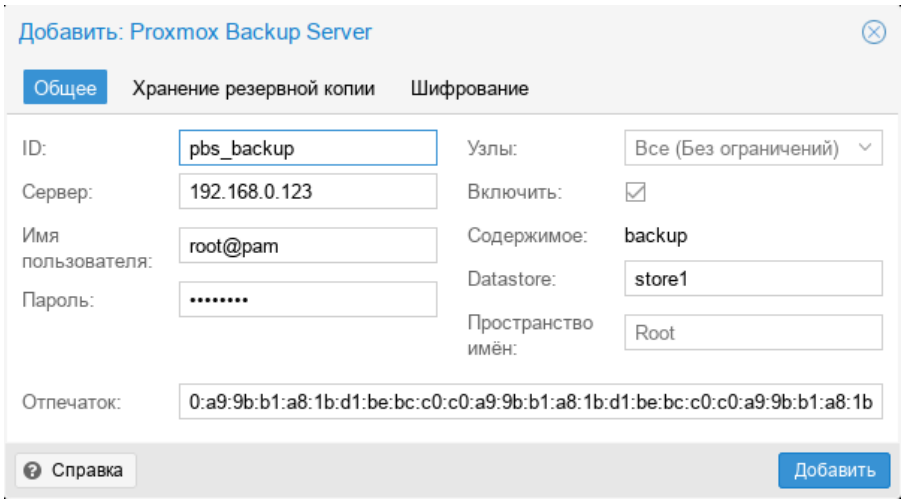

Рис. 429 – PVE. Диалог создания хранилища Proxmox Backup Server

Примечания:

1. Отпечаток TLS-сертификата можно получить в веб-интерфейсе сервера резервного копирования (см. рис. 425). Получить отпечаток также можно, выполнив следующую команду на сервере резервного копирования:

```
# proxmox-backup-manager cert info | grep Fingerprint
Fingerprint
(sha256):c8:26:af:4a:c3:dc:60:72:4a:0b:4d:c1:e6:58:02:62:90:39:cb:fc:7
5:5d:00:9a:57:ca:3d:28:a0:2c:99:a5
```

2. Добавление хранилища в командной строке:

```
# pvesm add pbs pbs_backup --server pbs.test.alt\
--datastore store2\
--fingerprint c8:26:af:4a:c3:dc:60:72:...:99:a5\
--username root@pam\
--password
```

3. Просмотреть состояние хранилища:

```
# pvesm status --storage pbs_backup Name Type Status Total Used Available % pbs_backup pbs active 30786448 3097752 26099504 10.06%
```

Если добавить хранилище данных типа Proxmox Backup Server в PVE, можно создавать резервные копии ВМ и контейнеров в это хранилище так же, как и в любые другие хранилища.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

ВМ – виртуальная машина;

ОЗУ – оперативное запоминающее устройство;

ОС – операционная система;

ПИ – программное изделие;

ПО – программное обеспечение;

ПЭВМ – персональная электронно-вычислительная машина;

ЦПУ – центральное процессорное устройство;

ЦУС – центр управления системой.

Лист регистрации изменений									
Номера листов (страниц)					*Всего листов (страниц) в докум.*	*№ документа*	*Входящий № сопроводительного докум. и дата*	*Подп.*	*Дата*
Изм.	*измененных*	*замененных*	*новых*	*аннулированных*					