Установка Proxmox в Debian на raid 1

[Zerox](https://serveradmin.ru/author/zerox/) Обновлено: 29.04.2021 [Debian](https://serveradmin.ru/category/linux/debian/), [Proxmox](https://serveradmin.ru/category/virtual/proxmox/), [Виртуализация](https://serveradmin.ru/category/virtual/) [367 комментариев](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#comments) 378,339 Просмотры

На сегодняшний день существует несколько наиболее популярных гипервизоров для построения виртуальной информационной системы. В данной статье я рассмотрю установку и настройку бесплатной системы виртуализации proxmox 6 на базе ОС Debian 10, установленной на RAID 1. В качестве гипервизоров она использует опенсорсные KVM и LXC, позволяя виртуализировать наиболее популярные ОС.

Научиться настраивать MikroTik с нуля или систематизировать уже имеющиеся знания можно на [углубленном курcе по администрированию MikroTik](https://serveradmin.ru/mikrotik-mtcna-txt). Автор курcа, сертифицированный тренер MikroTik Дмитрий Скоромнов, лично проверяет лабораторные работы и контролирует прогресс каждого своего студента. В три раза больше информации, чем в вендорской программе MTCNA, более 20 часов практики и доступ навсегда.

**Содержание**

[Введение](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Vvedenie)

[Установка proxmox на mdadm raid 1 в debian 10](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Ustanovka_proxmox_na_mdadm_raid_1_v_debian_10)

[Базовая настройка proxmox](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Bazovaa_nastrojka_proxmox)

[Настройка сети](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Nastrojka_seti)

[Как создать bridge в proxmox](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Kak_sozdat_bridge_v_proxmox)

[Настройка NAT для виртуальных машин](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Nastrojka_NAT_dla_virtualnyh_masin)

[Routed режим сети](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Routed_rezim_seti)

[Пример организации сети для виртуальных машин](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Primer_organizacii_seti_dla_virtualnyh_masin)

[Хранилище для виртуальных машин](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Hranilise_dla_virtualnyh_masin)

[Добавление и подключение диска](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Dobavlenie_i_podklucenie_diska)

[Хранилище по NFS](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Hranilise_po_NFS)

[Создание и настройка виртуальных машин](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Sozdanie_i_nastrojka_virtualnyh_masin)

[Автозапуск виртуальной машины](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Avtozapusk_virtualnoj_masiny)

[Backup виртуальных машин](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Backup_virtualnyh_masin)

[Часто задаваемые вопросы по теме статьи (FAQ)](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Casto_zadavaemye_voprosy_po_teme_stati_FAQ)

[Заключение](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Zaklucenie)

[Помогла статья? Подписывайся на telegram канал автора](https://serveradmin.ru/ustanovka-i-nastroyka-proxmox/#Pomogla_stata_Podpisyvajsa_na_telegram_kanal_avtora)

Введение

Некоторое время назад я узнал про систему виртуализации proxmox на базе KVM. Ранее с этим гипервизором я был знаком, но он мне не понравился из-за отсутствия удобных инструментов управления под windows. Это было давно, лет 5 назад. Мне пришлось администрировать уже настроенный гипервизор и мне это не понравилось, слишком много действий приходилось делать в консоли. Не скажу, что мне это прям не нравится, но я не вижу смысла в консоли делать то, что в других гипервизорах мышкой ты делаешь в 5 раз быстрее в gui. Свое время стараюсь экономить и использовать рационально.

Все изменилось, когда я решил посмотреть на [Proxmox](http://www.proxmox.com/en/). Простая установка и удобная панель управления через браузер привлекли сразу. Попробовал, потестировал, вроде все неплохо работает, управление удобное и понятное. Особенно понравились бэкапы из коробки. Не решался использовать в реальной работе, потому что не имею опыта работы c zfs, а ставить гипервизор на одиночные диски плохая идея. Раньше я использовал XenServer, установленный на mdadm raid1, пока он не перестал поддерживать такой режим работы. Proxmox почему-то не поддерживает установку на простой и понятный mdadm, но при этом есть zfs. Этот момент мне искренне не понятен, если учитывать, что proxmox работает на базе системы Debian, которая без проблем устанавливается на программный рейд.

В итоге я решил установить, настроить и протестировать proxmox, установленный на программный raid 1 mdadm. Диски отключал, вынимал, вставлял обратно. Все прекрасно работает. Отказоустойчивость на уровне дисков обеспечена, значит можно использовать в реальной работе. Я и использую последние пару лет. Отдельно упомяну, что меня сразу привлекло в KVM - возможность пробрасывать USB. До сих пор Hyper-V и XenServer не умеют это делать. Первый совсем не умеет, второй вроде как-то пробрасывает, но без гарантий и не все устройства работают. А в KVM без проблем получилось пробросить USB в виртуалку и воткнуть туда HASP ключ от 1С. Для малых и средних офисов это актуальная потребность.

Для того, чтобы установить proxmox на raid 1 пойдем окольным путем. Стандартный инсталлятор не дает нам необходимой возможности установки на рейд. У нас есть 2 варианта решения проблемы:

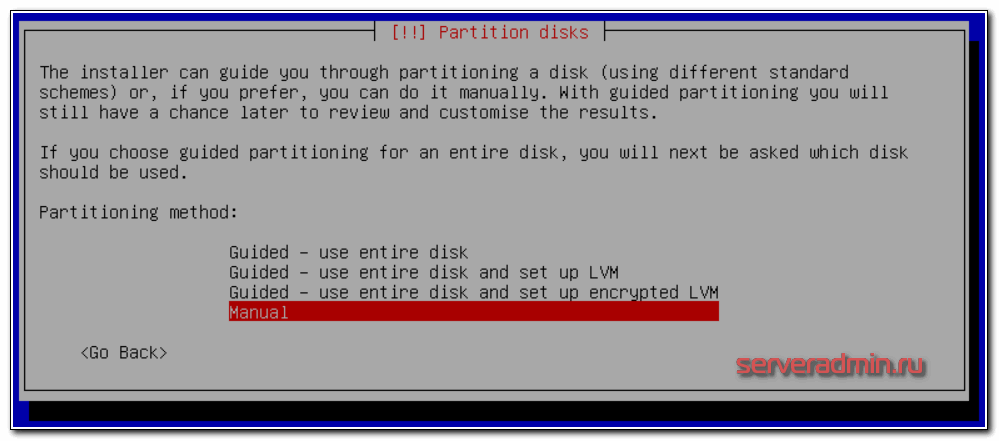
1. Установка сначала голой системы Debian 10 на raid1, а затем на нее устанавливается proxmox. Конечный результат не будет отличаться от инсталляции со стандартного диска.
2. Установить систему proxmox на одиночный диск, а потом перенести ее на raid1.

Я пробовал оба способа, первый мне показался более простым и понятным, поэтому займемся его реализацией.

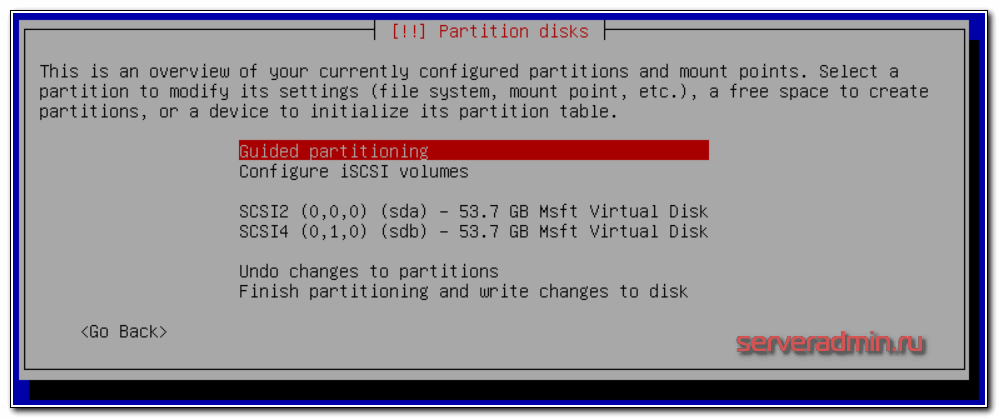
Установка proxmox на mdadm raid 1 в debian 10

Первым делом нам нужно установить чистую систему **Debian**. На данный момент это версия **10 Buster**. У меня есть отдельная статья по [установке Debian](https://serveradmin.ru/kak-skachat-i-ustanovit-debian-10-buster/). Там подробно описан процесс установки системы на софтовый рейд mdadm. В конце я на примере показал, что делать, когда диск выходит из строя, как его менять. Я на практическом примере продемонстрировал отказоустойчивость такого решения, поэтому не буду здесь подробно на этом останавливаться.

Скачивайте дистрибутив последней версии Debian. Взять его можно, к примеру, на [Yandex.Mirror](https://serveradmin.ru/ustanovka-repozitoriya-epel-rpmforge-v-centos/#Yandex_mirror_CentOS), конкретно [здесь](http://mirror.yandex.ru/debian-cd/current/amd64/iso-cd/). Для установки подойдет образ **CD-1** либо **netinst**. Начинайте стандартную установку и доходите до пункта настройки жесткого диска. Я буду использовать консольное отображение, не графическое. Мне так удобнее. Принципиальных отличий нет, можете делать по аналогии, если начали установку в графическом режиме. Когда дело дойдет до настройки диска, выбирайте вариант ручной разметки.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-01.png)

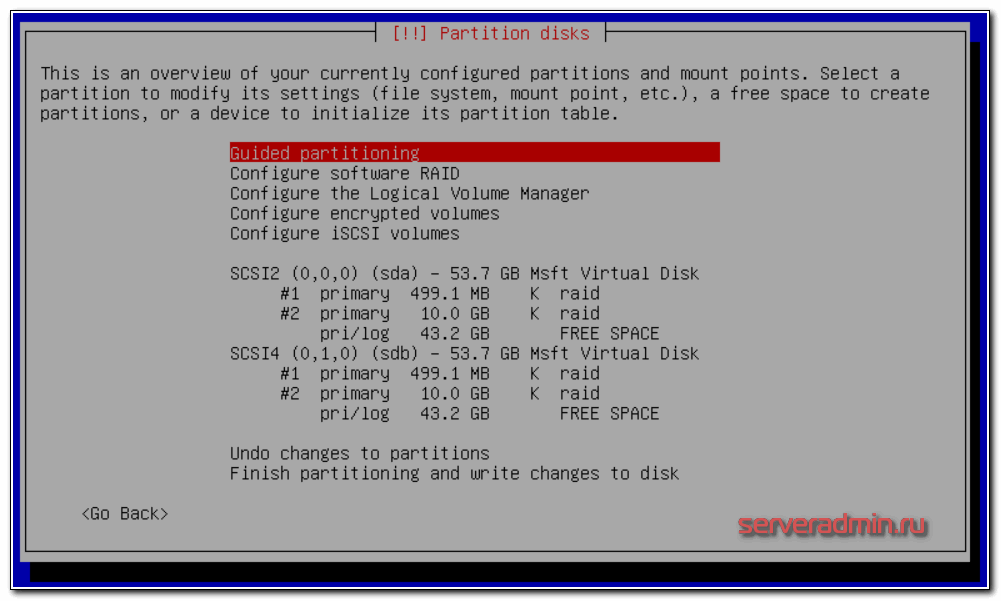
Если вы делаете установку на чистые диски, то вас должна встретить такая картина состояния дисков:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-02.png)

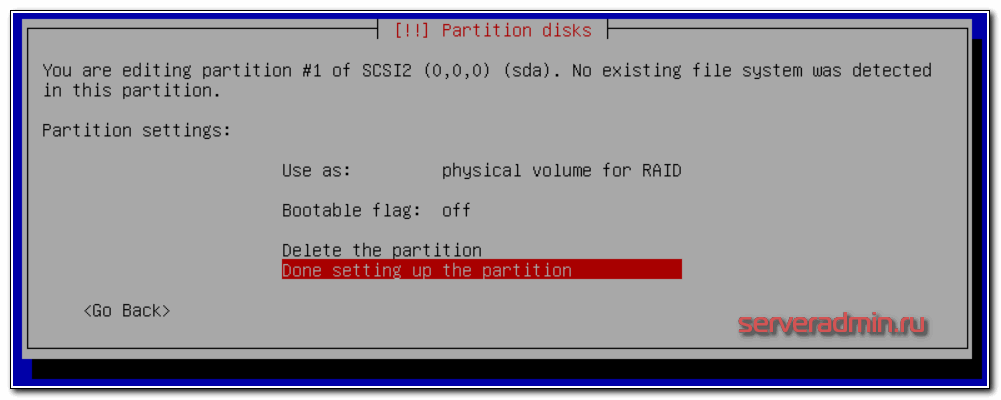
Если есть какие-то разделы, то удалите их все, чтобы были чистые диски. Для написания статьи я использую виртуальную тестовую среду, диски выделены небольшого объема. Для учебных целей этого достаточно. Я разобью диски следующим образом:

|  |  |
| --- | --- |
| /boot | Загрузочный раздел 500 мб |
| / | Корневой раздел будущей системы, 10 гб выделяю в тесте, в реальной работе рекомендую сделать 30-50 гб на всякий случай |
| Свободное место | Все остальное пространство диска оставляю неразмеченным. Как его разделить и использовать будет зависеть от ваших потребностей. Позже покажу как его все задействовать под различные хранилища (виртуальных машин, образов или бэкапов) |

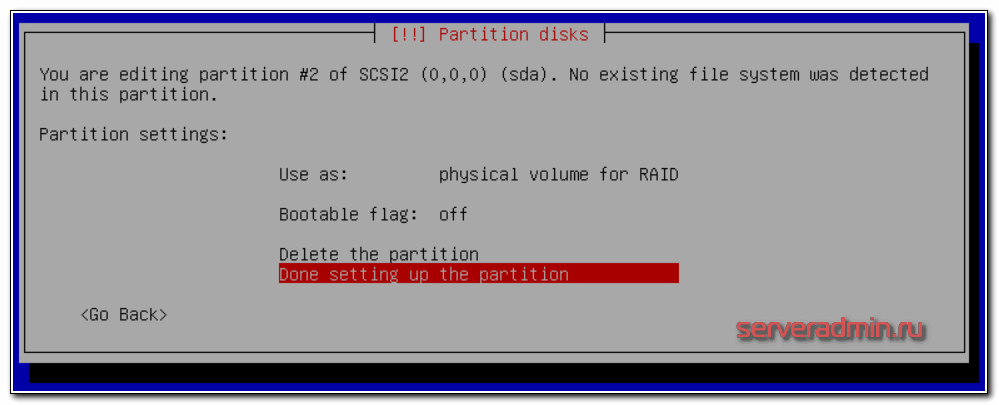
Создаем в инсталляторе указанные разделы на обоих дисках. Не буду расписывать по шагам как их сделать, надеюсь сами разберетесь, это не сложно. Должна получиться такая картина:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-03.png)

Параметры первого раздела на 500 Мб:

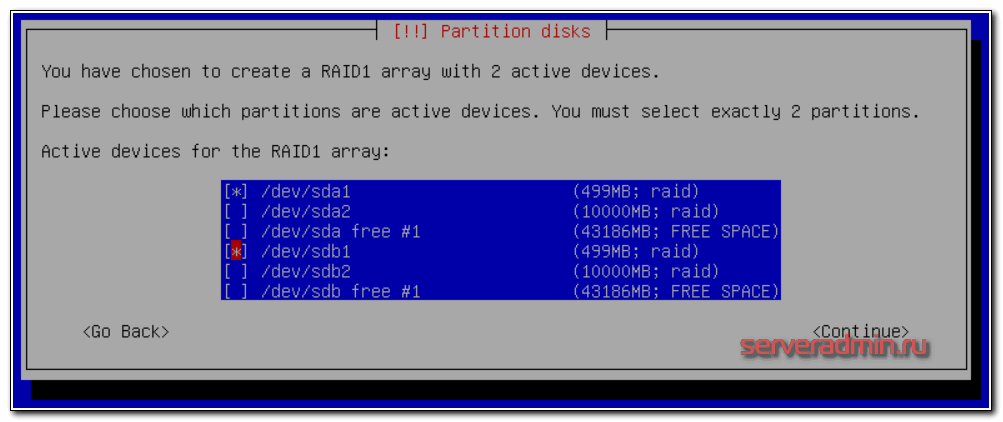
[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-04.png)

Параметры второго раздела на 10 Гб:

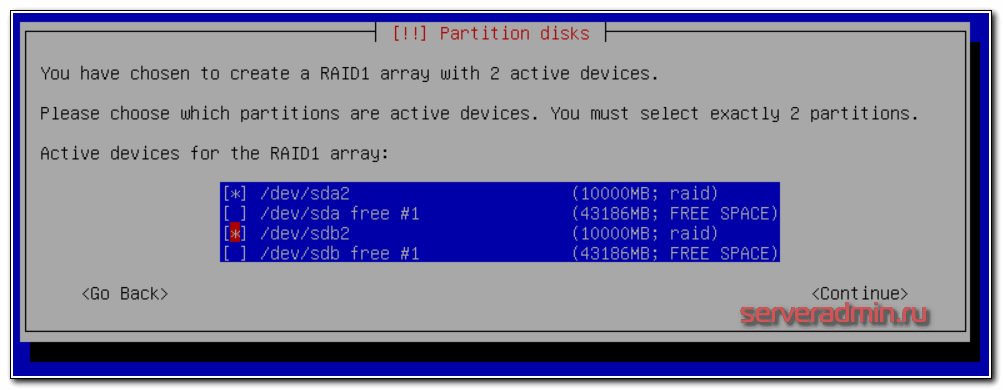
[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-05.png)

Оба раздела должны быть **Primary**.

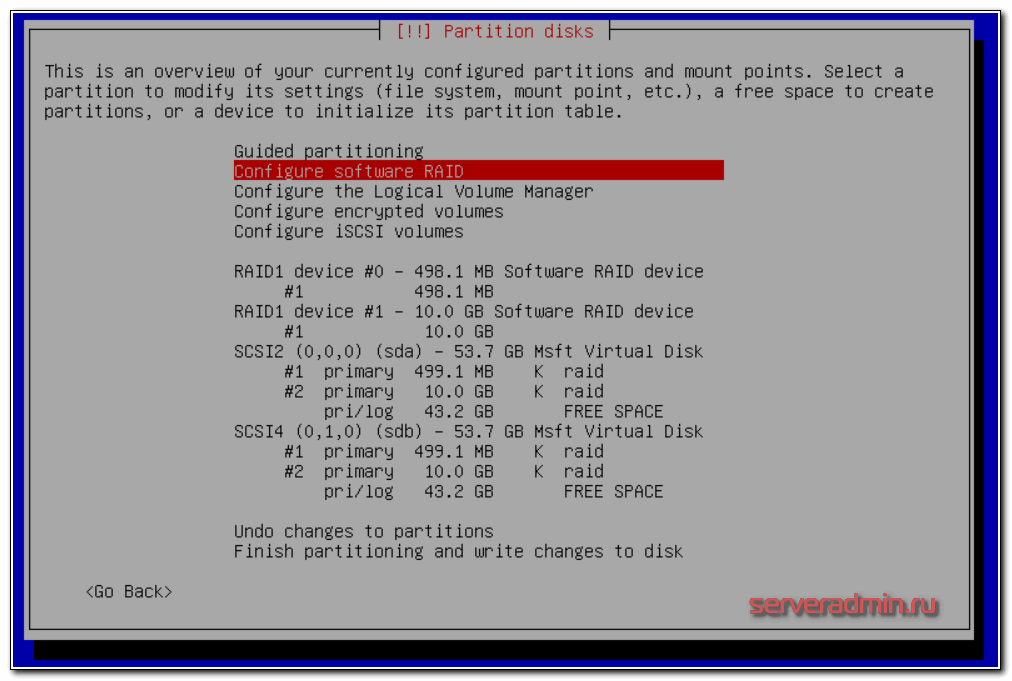
Теперь создадим 2 отдельных raid массива на 500 мб и 10 гб. Выбираем раздел **Configure Software RAID**, дальше Create MD device, потом RAID1, 2 диска в массиве, и в завершении выбираем 2 наших раздела по 500 мб:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-06.png)

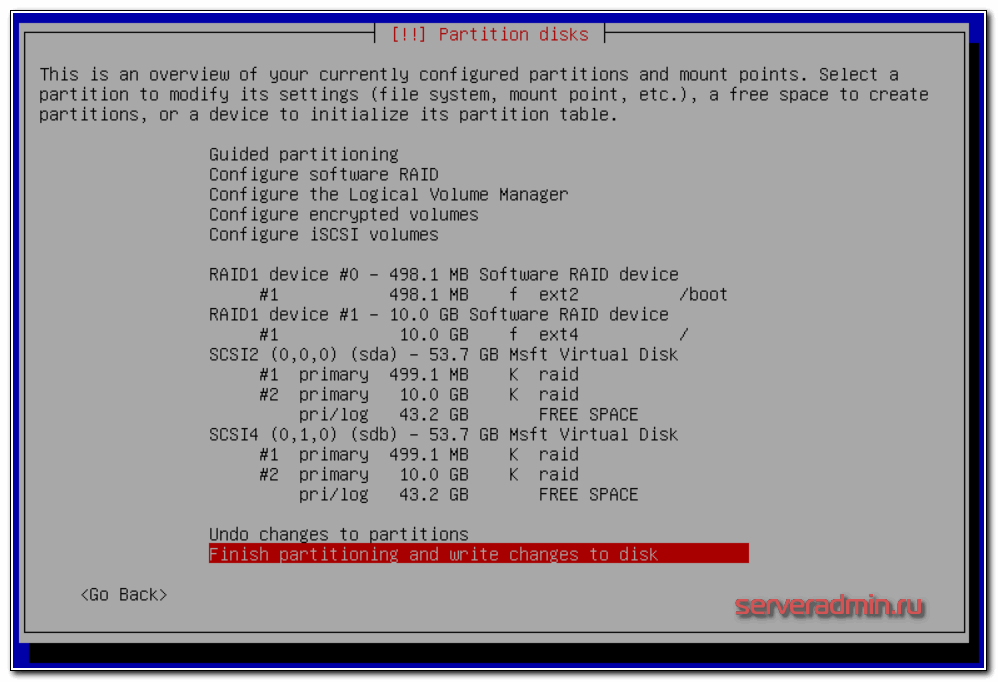
То же самое делаем с разделами по 10гб - объединяем их в рейд:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-07.png)

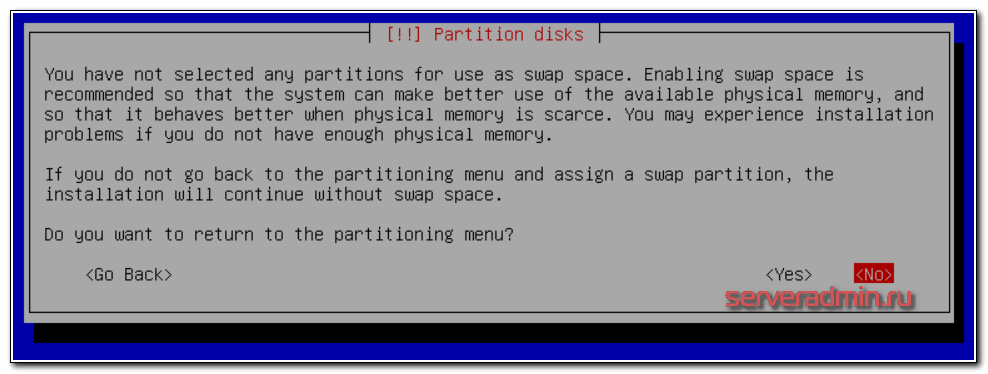
Как закончите, жмете Finish. У вас должна получиться такая картина:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-08.png)

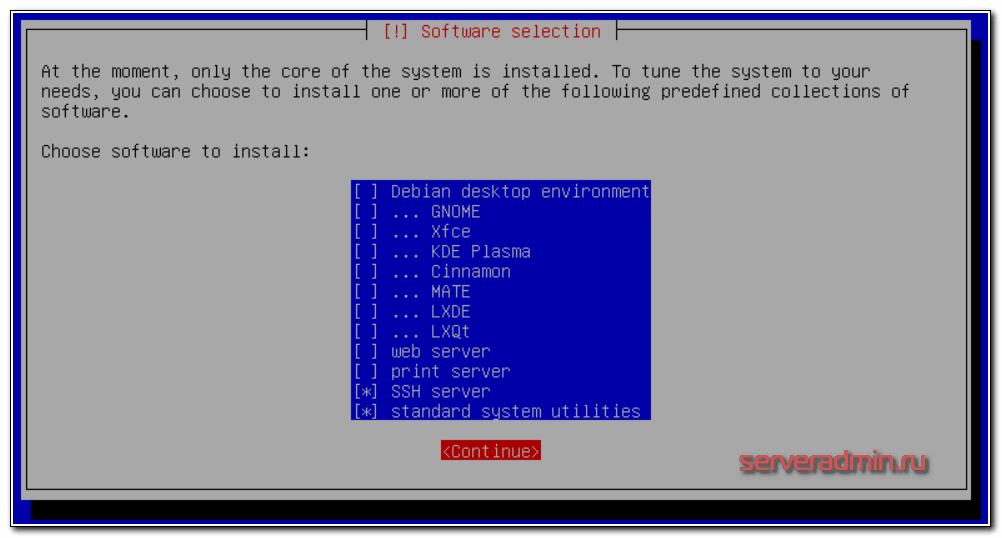
Теперь нужно создать файловую систему на рейд массивах. Сделаем на первом раздел /boot ext2, а на втором корень системы - / и файловую систему ext4. В итоге должно получиться вот так:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-09.png)

Применяем изменения и продолжаем стандартную установку. У вас будет предупреждение о том, что не указан раздел для swap.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-10.png)

Игнорируйте его, потом подключим swap отдельно в виде файла. В процессе установки вам будет предложено выбрать набор пакетов. Не ставьте ничего лишнего, только ssh сервер и системные утилиты.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-11.png)

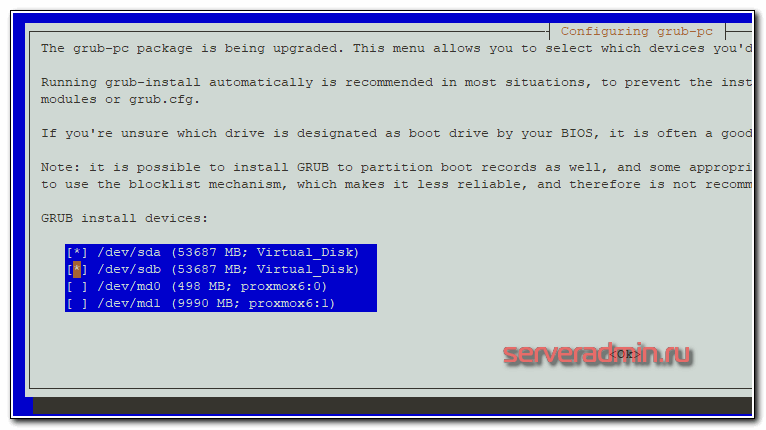
В конце будет предложено установить загрузчик на один из дисков. Можете выбрать любой диск, позже мы установим загрузчик на второй диск и убедимся, что он будет установлен на обоих дисках. Это нужно для того, чтобы система могла загружаться с любого из дисков, в случае выхода одного из строя.

После установки рекомендую выполнить [базовую настройку Debian](https://serveradmin.ru/debian-nastroyka-servera/). Это не обязательно, посмотрите на мои рекомендации и делайте то, что посчитаете нужным. Если гипервизор будет внутри локальной сети, то фаервол можно выключить, порт ssh не трогать, а логин под root разрешить. Обновиться, [настроить сеть](https://serveradmin.ru/nastroyka-seti-v-debian/), время и полезные утилиты. Так же на всякий случай [создайте swap раздел](https://serveradmin.ru/dobavit-swap-razdel-v-linux/) хотя бы на 2-4 гигабайта.

Теперь установим grub на оба жестких диска. Подключаемся к серверу по ssh и выполняем команду:

# dpkg-reconfigure grub-pc

На все вопросы оставляете дефолтные значения, в конце выбираете оба жестких диска для установки загрузчика:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-12.png)

На всякий случай проверим как встала система на жесткие диски:

# df -h

Filesystem Size Used Avail Use% Mounted on

udev 3.9G 0 3.9G 0% /dev

tmpfs 798M 17M 782M 3% /run

/dev/md1 9.1G 1020M 7.7G 12% /

tmpfs 3.9G 0 3.9G 0% /dev/shm

tmpfs 5.0M 0 5.0M 0% /run/lock

tmpfs 3.9G 0 3.9G 0% /sys/fs/cgroup

/dev/md0 460M 47M 390M 11% /boot

tmpfs 798M 0 798M 0% /run/user/0

# cat /proc/mdstat

Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]

md0 : active raid1 sdb1[1] sda1[0]

486400 blocks super 1.2 [2/2] [UU]

md1 : active raid1 sda2[0] sdb2[1]

9756672 blocks super 1.2 [2/2] [UU]

Все в порядке, получилось так, как и задумывали. На данном этапе можно повынимать жесткие диски и проверить как система работает без одного из них, научиться заменять сбойный диск. Команды и рекомендации не буду приводить, чтобы не раздувать статью. Мануалов по работе с mdadm в интернете полно. К тому же, я об этом рассказываю в статье про установку debian, ссылку на которую уже давал ранее. Рекомендую сразу настроить [мониторинг mdadm в zabbix](https://serveradmin.ru/monitoring-programmnogo-reyda-mdadm-v-zabbix/).

Переходим непосредственно к установке proxmox. Для этого редактируем файл */etc/hosts* и приводим его строго к следующему виду. Если что-то будет не так, как указано у меня, получите ошибку установки с очень большой долей вероятности. Я на этом моменте прилично застрял, когда разбирался.

# mcedit /etc/hosts

127.0.0.1 localhost.localdomain localhost

192.168.155.104 proxmox6.local proxmox6 pvelocalhost

|  |  |
| --- | --- |
| proxmox6 | Имя сервера, указанное во время установки |
| local | Домен, указанный во время установки |
| 192.168.155.104 | IP адрес сервера |

Проверить правильность настроек можно командой:

# hostname --ip-address

192.168.155.104

В ответ должны получить свой ip адрес.

Добавляем в список репозиториев репу proxmox и стандартные репозитории debian. Я буду использовать зеркало яндекса, все остальное можно закомментировать:

# mcedit /etc/apt/sources.list

deb http://mirror.yandex.ru/debian/ buster main non-free contrib

deb-src http://mirror.yandex.ru/debian/ buster main non-free contrib

deb http://download.proxmox.com/debian/pve buster pve-no-subscription

Если не подходит репозиторий яндекса (заблокирован), можно воспользоваться любым другим, например http://mirror.corbina.net/debian/

Добавляем цифровую подпись proxmox репозитория:

# wget http://download.proxmox.com/debian/proxmox-ve-release-6.x.gpg -O /etc/apt/trusted.gpg.d/proxmox-ve-release-6.x.gpg

Если раньше не обновили систему, то обновитесь и на всякий случай перезагрузите сервер после этого:

# apt update && apt full-upgrade

# reboot

Теперь устанавливаем саму систему виртуализации proxmox:

# apt install proxmox-ve postfix open-iscsi

Если получите ошибку во время установки:

dpkg: error processing package proxmox-ve (--configure):

dependency problems - leaving unconfigured

Errors were encountered while processing:

pve-cluster

libpve-access-control

librados2-perl

pve-firewall

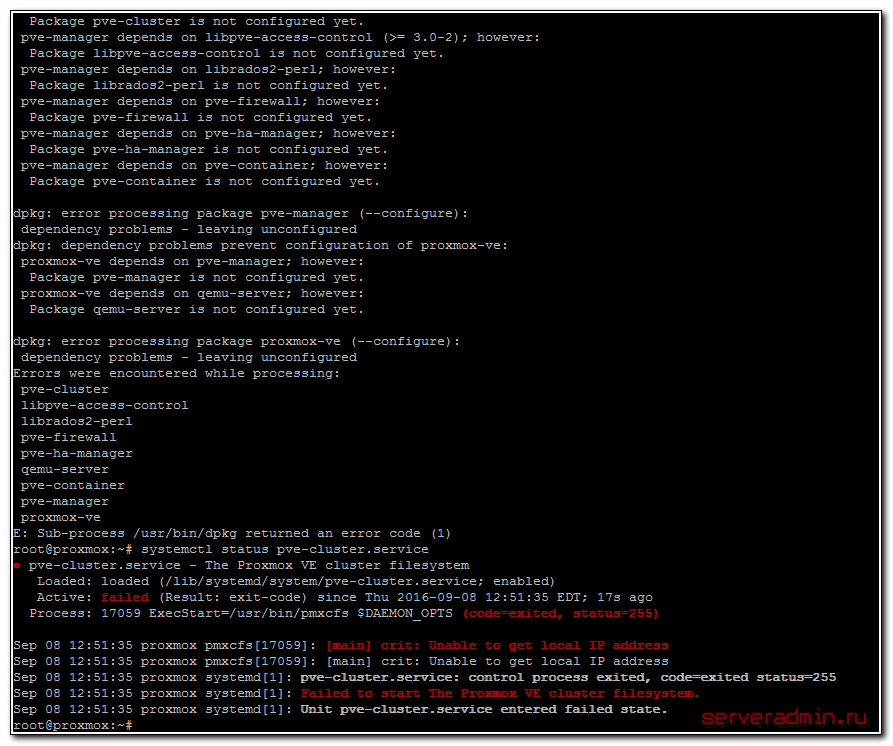
pve-ha-manager

qemu-server

pve-container

pve-manager

proxmox-ve

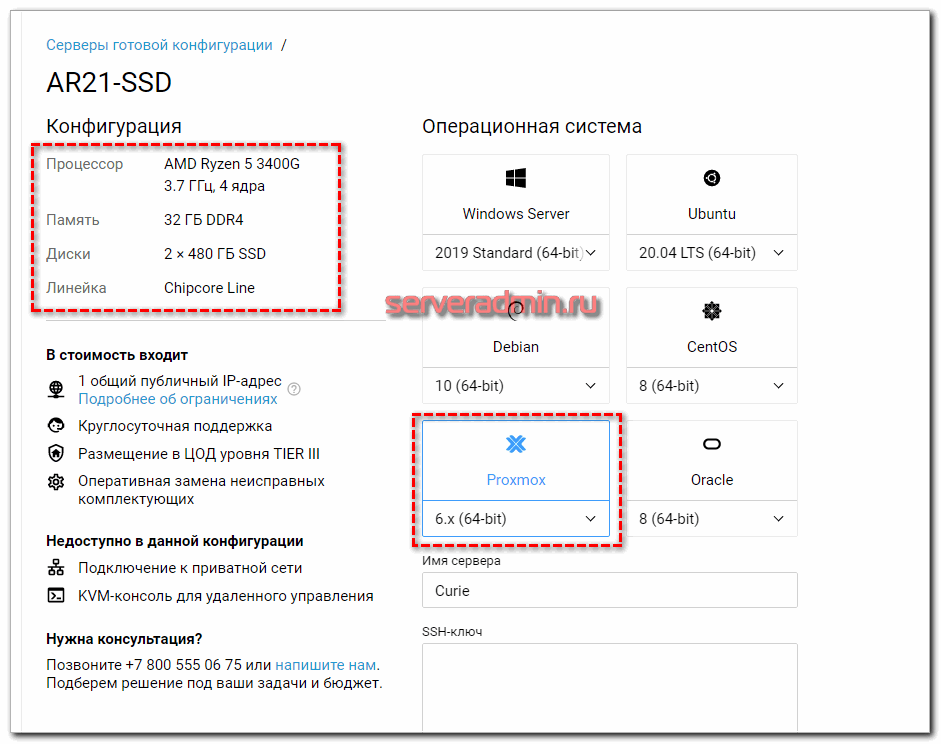


Проверяйте файл hosts. У меня там лишние строки были, кроме тех, что я указал ранее. Когда исправил, установка прошла без ошибок. После окончания установки перезагрузите сервер, чтобы загрузилось новое ядро. Если все в порядке, то увидите окно приветствия на мониторе сервера:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-13.png)

Открывайте браузер по указанному адресу и заходите в web интерфейс. Напоминаю, что **web порт proxmox по-умолчанию - 8006**. Не забывайте его указывать в строке адреса в браузере. Вы должны увидеть предупреждение браузера насчет сертификата. Так и должно быть, по-умолчанию используется самоподписанный сертификат.

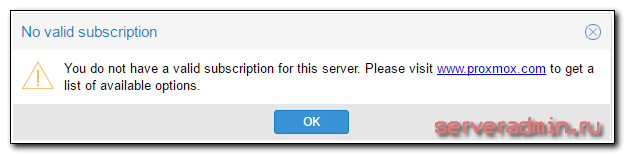
Если у вас нет своего железа для установки proxmox, можно арендовать недорогой сервер у [Selectel](https://serveradmin.ru/lklTref4). Линейка выделенных севреров Chipcore стартует с абонентки в 1500 р. в месяц. Через панель управления во время заказа сервера можно сразу выбрать proxmox, установленный на софтовый рейд mdadm. Самим ничего делать не нужно будет. Сразу получите отказоустойчивый настроенный гипервизор.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-33.png)

Базовая настройка proxmox

Заходим через браузер по указанному адресу. В окне логина вводите системный root и пароль от него. Нас сразу же встречает предупреждение:

You do not have a valid subscription for this server. Please visit www.proxmox.com to get a list of available options.

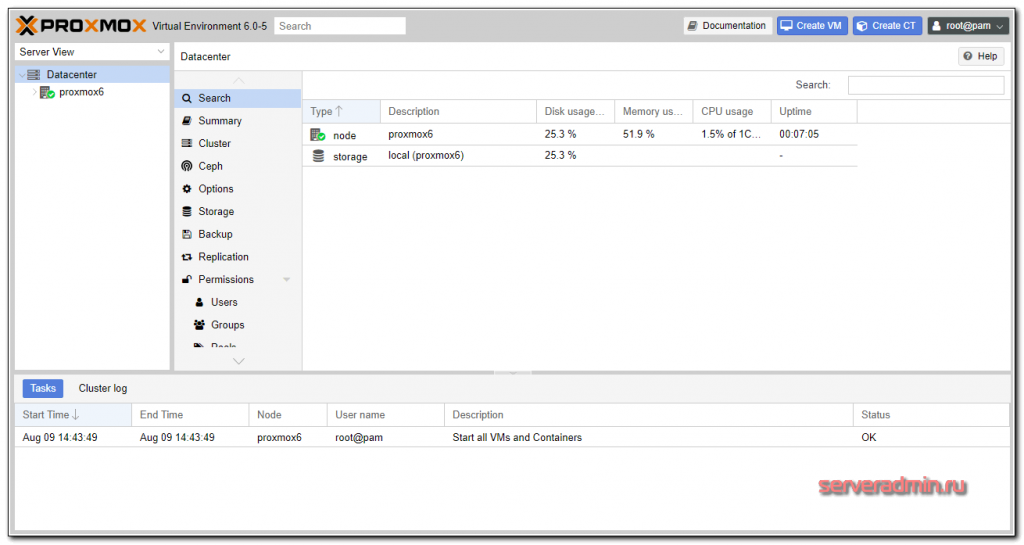


У нас нет платной подписки, поэтому нужно удалить из списка репозиториев платный. Для этого открываем консоль сервера и закомментируем репозиторий:

# mcedit /etc/apt/sources.list.d/pve-enterprise.list

#deb https://enterprise.proxmox.com/debian/pve buster pve-enterprise

Дальше видим главную страницу управления гипервизором proxmox 6.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-14.png)

Настройка сети

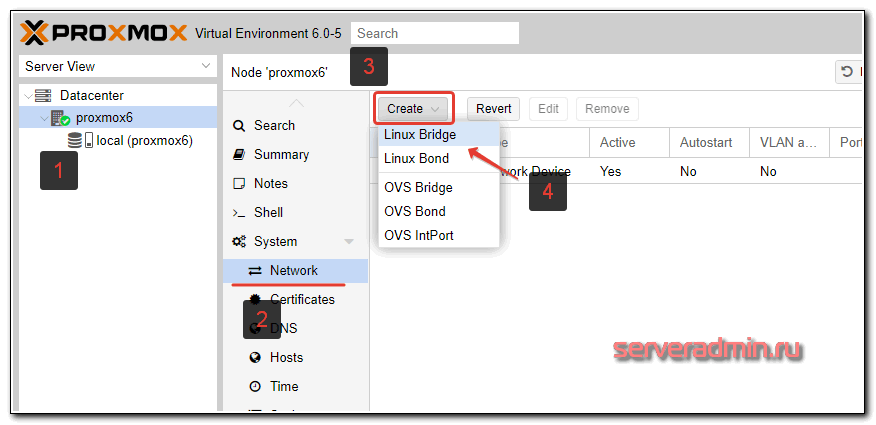
Обычно для виртуальных машин достаточно 3 режима работы сети:

1. **Режим Bridge**. В этом режиме виртуальные машины получают ip адрес из одной подсети с гипервизором и имеют в нее прямой доступ.
2. **Режим NAT**. Виртуальные машины получают ip адреса в своей виртуальной подсети, во внешнюю сеть выходят через гипервизор и настроенный на нем NAT.
3. **Routed режим**, когда шлюзом в интернет является одна из виртуальных машин.

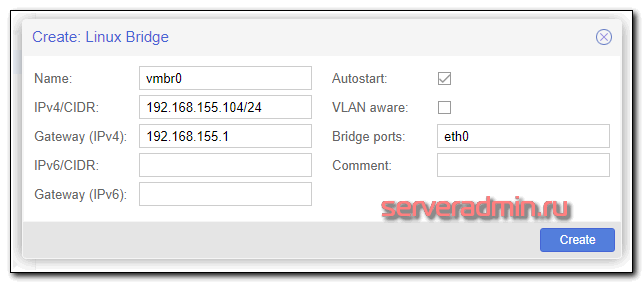
Я использую все три режима работы сети, в зависимости от ситуации. Покажу это на примерах.

Как создать bridge в proxmox

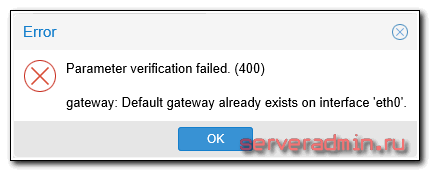
Для создания сетевого интерфейса типа bridge в proxmox, в web интерфейсе перейдите в раздел Network и нажмите Create -> Linux Bridge.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-15.png)

Заполняете необходимые поля. Обязательными являются поле IP address, Gateway, Bridge Ports.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-16.png)

С такими настройками вы сможете в качестве локального интерфейса оставить только vmbr0, а eth0 отключить. Это на ваше усмотрение. Если оставить оба интерфейса, то ваш proxmox будет доступен по двум разным ip адресам. Если получите ошибку:



Удалите настройку шлюза на eth0, добавьте его на vmbr0.

Сразу обращаю внимание, что после создания бриджа, в системе создается новый файл с сетевыми настройками - */etc/network/interfaces.new*. Там отражены сделанные нами изменения. Информация из него будет добавлена в основной файл *interfaces*после перезагрузки. Перезагрузите сервер и проверьте. Если все в порядке, то вы сможете подключаться и по ssh и по web доступу к обоим ip адресам - eth0 и vmbr0.

Очень внимательно отнеситесь к настройке сети в proxmox. Убедитесь, что у вас есть доступ к консоли гипервизора. Мне часто приходится настраивать выделенные серверы. Тем не менее, иногда теряю доступ к серверу из-за какой-то ошибки или невнимательности. Да и в самом proxmox могут быть проблемы.

У меня были ситуации, когда выполняешь описанные выше действия, перезагружаешь сервер - он недоступен. Захожу локально, открываю файл */etc/network/interfaces*, и удаляю какую-то лишнюю закомментированную строку, из-за которой не поднималась сеть. Подробностей уже не помню, что именно было не так, но пару раз ловил такие ошибки.

Расскажу про еще один нюанс. Во время создания бриджа вас обязательно просят указать статические сетевые настройки. А иногда нужно использовать dhcp сервер. Через web интерфейс это сделать не получится, либо я не понял как. Поступаю другим образом. Пишу произвольные настройки через gui, а потом исправляю файл */etc/network/interfaces*. Вот пример настройки сети, когда bridge получает сетевые настройки по dhcp.

source /etc/network/interfaces.d/\*

auto lo

iface lo inet loopback

allow-hotplug eth0

#iface eth0 inet dhcp

auto vmbr0

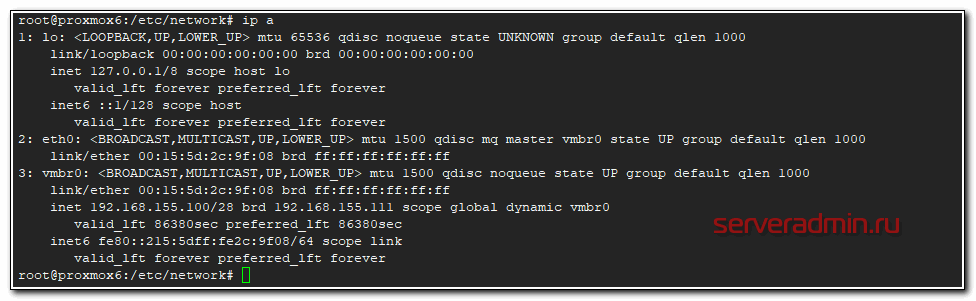
iface vmbr0 inet dhcp

bridge-ports eth0

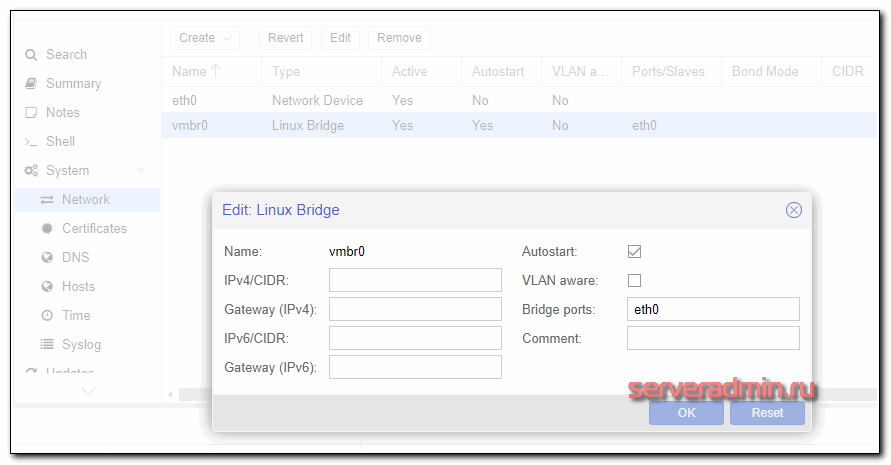
bridge-stp off

bridge-fd 0

Перезагружаю сервер и убеждаюсь, что все работает как надо.

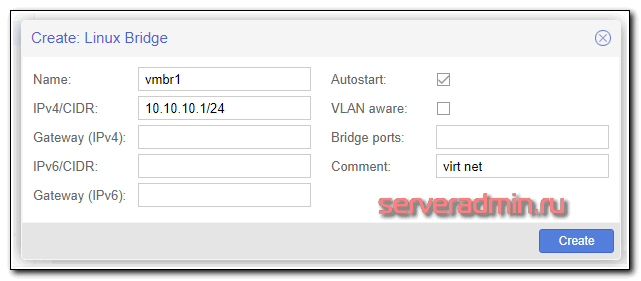
[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-17.png)

При этом в web интерфейсе будет вот так.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-18.png)

Настройка NAT для виртуальных машин

Для того, чтобы настроить удобный способ сетевого подключения виртуальных машин с использованием NAT, когда все виртуалки вместе с гипервизором будут в единой виртуальной сети видеть друг друга, потребуется создать еще один **bridge** и настроить трансляцию адресов с помощью **iptables** на самом гипервизоре. Сделаем это.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-19.png)

Заходим по ssh на гипервизор и добавляем в файл с новыми сетевыми настройками несколько строк:

# mcedit /etc/network/interfaces.new

auto vmbr1

iface vmbr1 inet static

address 10.10.10.1

netmask 255.255.255.0

bridge\_ports none

bridge\_stp off

bridge\_fd 0

**post-up echo 1 > /proc/sys/net/ipv4/ip\_forward**

**post-up iptables -t nat -A POSTROUTING -s '10.10.10.0/24' -o vmbr0 -j MASQUERADE**

**post-down iptables -t nat -D POSTROUTING -s '10.10.10.0/24' -o vmbr0 -j MASQUERADE**

Объясняю, что мы сделали:

1. Разрешили форвард пакетов между сетевыми интерфейсами. Без этого гипервизор не сможет работать в роли шлюза.
2. Добавили правила [iptables](https://serveradmin.ru/nastroyka-iptables-v-centos-7/) для настройки NAT.

Сохраняем файл и перезагружаем сервер. В настройках сети виртуальных машин указываете интерфейс **vmbr1**, а в самой системе виртуалки вручную прописываете сетевые настройки, где адрес шлюза равен адресу интерфейса vmbr1 - 10.10.10.1, а ip адрес самой машины будет в подсети *10.10.10.0/24*. Все виртуальные машины, у которых установлен бридж vmbr1 будут видеть друг друга.

Если вам не хочется на каждой ВМ вручную прописывать IP, можно настроить dhcp сервер либо на одной из виртуалок, либо на самом гипервизоре. Главное назначить ему нужный интерфейс - vmbr1. Настройка dhcp сервера выходит за рамки этой статьи, не буду на этом останавливаться. Для общего использования достаточно того, что я уже указал.

Routed режим сети

Последний вариант, который я иногда использую. В качестве шлюза в интернет для локальных машин, а если необходимо и офиса, выступает виртуальная машина. Этот режим ничем особо не отличается от режима NAT, кроме того, что сам нат на гипервизоре не нужен, так как натить трафик будет шлюз на виртуальной машине. Нам нужны будут 2 бриджа: один для передачи сети провайдера в виртуальную машину-шлюз, второй для сети виртуальных машин. Настройка может немного отличаться, в зависимости от того, что вы хотите получить. Тут могут быть 2 варианта:

1. У вас гипервизор с 1 физической сетевой картой, в которую воткнут шнурок с интернетом от провайдера. Вся ваша инфраструктура виртуальная, доступ только через интернет. Например, вы арендуете выделенный сервер где-то и используете для различных целей.
2. На вашем физическом сервере 2 и более сетевых карт. В первую приходит интернет от провайдера, вторая смотрит в локальную сеть. На гипервизоре виртуальные машины используются пользователями из локальной сети и находятся с ними в едином адресном пространстве.

В обоих случаях есть еще варианты настройки в зависимости от количества ip адресов, которые вам выдает провайдер. Если у вас только 1 внешний IP адрес, вы должны его пробросить в виртуальный шлюз. Только он будет иметь доступ в интернет. Для этого вы должны создать бридж вместе с сетевой картой, в которую приходит интернет, но при этом сам бридж не должен иметь IP адреса. Вот пример, как это должно выглядеть в первом варианте:

iface eth0 inet manual

auto vmbr0

iface vmbr0 inet manual

bridge\_ports eth0

bridge\_stp off

bridge\_fd 0

auto vmbr1

iface vmbr1 inet static

address 10.0.0.1

netmask 255.255.255.0

gateway 10.0.0.2

bridge\_ports none

bridge\_stp off

bridge\_fd 0

В **eth0** входит провод от провайдера. Этот интерфейс включен в **vmbr0** и ему не назначен ip адрес. Второй бридж **vmbr1** имеет виртуальный ip адрес и создан для локальной сети виртуальных машин. Дальше вы создаете виртуальную машину для шлюза и добавляете ему оба бриджа - vmbr0 и vmbr1. На первом настраиваете ip в соответствии с настройками провайдера, на втором в данном случае указываете статический ip адрес 10.0.0.2, который будет являться шлюзом для всех виртуальных машин и самого гипервизора в том числе. Это отражено в параметре gateway в свойствах vmbr1. Потом настраиваете виртуальный [шлюз](https://serveradmin.ru/tag/gate/) и все будет работать.

Виртуальному шлюзу нужно не забыть поставить автозапуск при старте гипервизора, чтобы можно было перезагружать гипервизор удаленно и не терять доступ. Схема кажется немного запутанной, надо просто вникнуть и разобраться. На самом деле это удобно в том случае, если у вас один единственный сервер и на нем надо развернуть всю инфраструктуру. Шлюз замечательно живет на виртуальной машине, легко бэкапится и переносится. Настроить можно любой функционал. Единственный нюанс - настраивать нужно локально, тщательно проверить и только убедившись, что все корректно работает и перезагружается, ставить на постоянную работу.

Таким образом, у вас будет гипервизор, на нем шлюз в виде виртуальной машины. Все настройки сети выполняются на шлюзе, гипервизор трогать вообще не надо. Если у вас несколько гипервизоров в разных местах, вы их объединяете в единую сеть с помощью, к примеру, [openvpn](https://serveradmin.ru/nastroyka-openvpn-na-centos-7/), который настраивается на самих шлюза. Виртуальные машины отдельно настраивать не надо. Они замечательно будут видеть друг друга через свои шлюзы на гипервизорах.

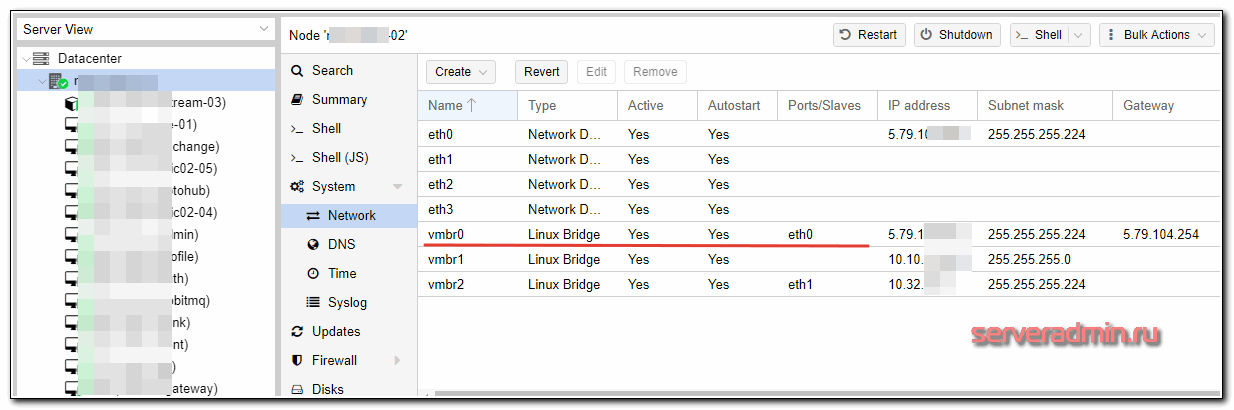
Если у вас две сетевые карты, то все то же самое. Первый бридж для интернета от провайдера без ip, второй бридж для виртуальных машин и локальной сети. Нужно только указать в нем bridge\_ports eth1, если eth1 используется для физического подключения в локальную сеть.

Я часто использовал такую схему подключения, каких-то особых проблем и подводных камней тут нет. Если сервер под гипервизор надежный, то работает такое решение зачастую лучше, чем отдельный бюджетный роутер. А функционал в разы больше. Последнее время стараюсь под шлюз использовать [Mikrotik](https://serveradmin.ru/mikrotik/) там, где это возможно и оправданно.

Пример организации сети для виртуальных машин

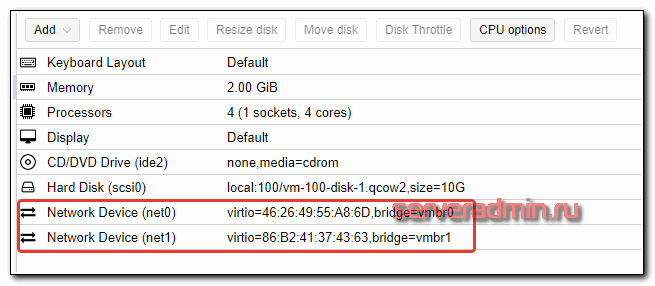
Я сейчас покажу на конкретном примере как настроить сеть в proxmox по последнему описанному способу. Для доступа в интернет виртуальных машин будет использоваться шлюз в виде отдельной виртуальной машины. На гипервизоре ничего делать не нужно будет, кроме создания бриджа.

Итак, вот настройки сети на гипервизоре.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-31.png)

eth0 смотрит в интернет, vmbr0 бридж, в который включен интерфейс eth0. На обоих интерфейсах настроены внешние ip адреса провайдера, но это не обязательно. Их может и не быть. Тут просто у сервера 8 внешних ip, хватает для всех. vmbr1 полностью виртуальный бридж для организации сети виртуальных машин.

Далее создаем виртуальную машину под шлюз. Добавляем ей оба бриджа.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-32.png)

Диска, как видите, хватит и 10 гб для шлюза. Он уже пару лет так работает. Теперь смотрим на настройки сети у этой виртуалки.

auto ens18

iface ens18 inet static

address 5.79.111.222

netmask 255.255.255.224

gateway 5.79.111.1

dns-nameservers 1.1.1.1 8.8.4.4

post-up iptables-restore < /etc/iptables.rules

auto ens19

iface ens19 inet static

address 10.10.11.1

netmask 255.255.255.0

Ens18 соответствует vmbr0, а ens19 - vmbr1. На ens18 настроен внешний ip и шлюз провайдера, который выделил этот ip адрес и выдал сетевые настройки. Настраиваем на этой виртуальной машине шлюз так, как нам надо - [iptables](https://serveradmin.ru/nastroyka-iptables-v-centos-7/), [dns](https://serveradmin.ru/nastroyka-dns-servera-bind-v-centos-7/), dhcp и т.д. Далее этот шлюз указываем в качестве default gateway для остальных виртуальных машин гипервизора.

Вот пример настроек сети с одной из виртуальных машин этого гипервизора.

auto eth0

iface eth0 inet static

address 10.10.11.17

netmask 255.255.255.0

gateway 10.10.11.1

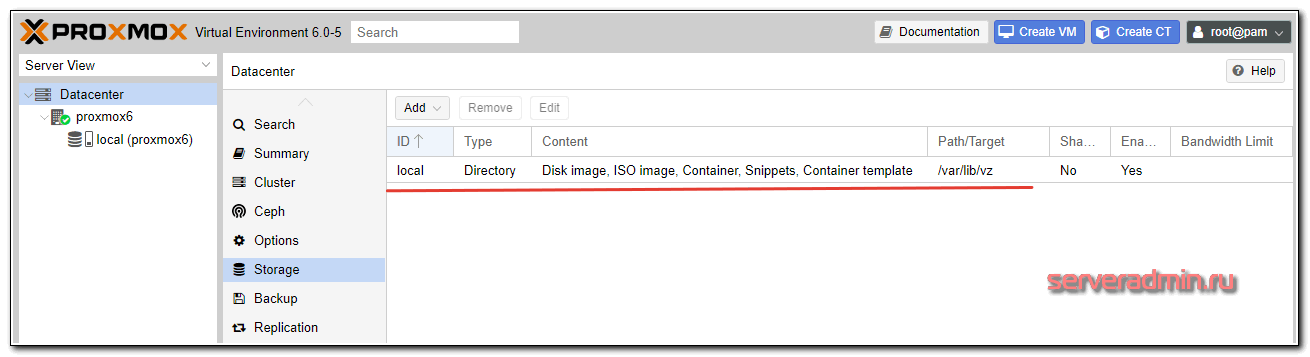
В данном случае eth0 это бридж vmbr1 гипервизора. Виртуальная машина подключена только к виртуальной сети гипервизра и имеет выход в интернет через шлюза, настроенный выше. Все доступы к виртуальной машине, проброс портов и т.д. настраивается на шлюзе. Если таких гипервизоров много, то они через vpn на шлюзах объединяются.

Такой подход удобен тем, что не надо вообще трогать гипервизор. Все настройки сети (а они могут быть сложными и их может быть много) хранятся на маленьком шлюзе, который удобно бэкапить и разворачивать. В случае необходимости, из бэкапов быстро поднимается вся инфраструктура на чистом гипервизоре.

Если кому-то не понятна описанная схема, задавайте вопросы в комментариях. Я в таком режиме эксплуатирую гипервизоры уже давно. Мне это кажется удобным. Если у вас есть предложения, как можно организовать сеть виртуальных машин в proxmox более удобно, делитесь соображениями.

Хранилище для виртуальных машин

После чистой установки proxmox на debian, автоматически создается хранилище **local**, которое располагается на одном разделе с самой операционной системой по адресу */var/lib/vz*. Оно уже предназначено для хранения образов дисков, виртуальных машин и контейнеров.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-20.png)

Для удобства лучше этот раздел не занимать, оставить необходимое пространство для нормальной работы ОС, а для виртуальных машин создать отдельное хранилище в незанятой области жестких дисков, которую мы оставили во время установки, либо на отдельном жестком диске или рейд массиве. Сначала рассмотрим вариант создания хранилища на неразмеченной области диска.

Мы создадим еще один **raid 1 mdadm** из незанятого места на дисках. Для этого нам нужно будет там создать разделы и объединить их в программный рейд с помощью mdadm.

Обращаю сразу внимание, что это тестовая среда, в реальности так диски разбивать не надо. Оптимально сделать raid 1 из двух отдельных под систему, заняв 30-50 гигов, остальное место на этом рейде отдать под iso образы и возможно бэкапы. Для виртуальных машин подключить отдельные диски и собрать из них рейд. Его уровень будет зависеть от количества дисков. Оптимально 4 штуки для raid 10

Смотрим имена дисков в нашей системе:

# ls -l /dev | grep sd

brw-rw---- 1 root disk 8, 0 Aug 9 15:26 sda

brw-rw---- 1 root disk 8, 1 Aug 9 15:26 sda1

brw-rw---- 1 root disk 8, 2 Aug 9 15:26 sda2

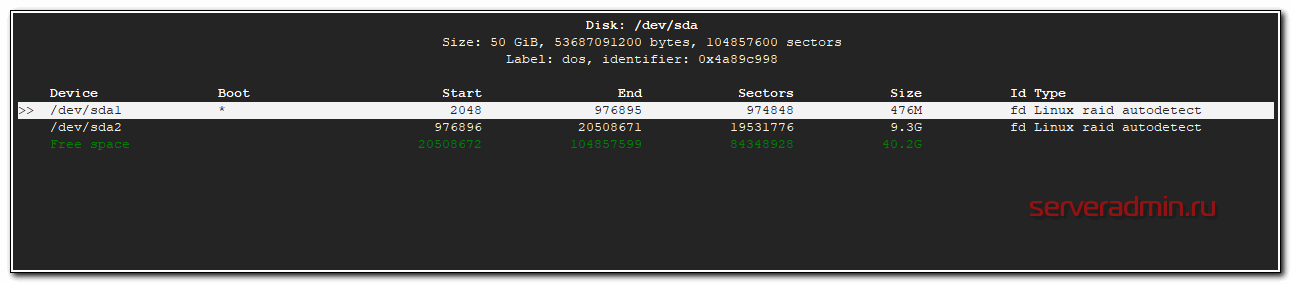
brw-rw---- 1 root disk 8, 16 Aug 9 15:26 sdb

brw-rw---- 1 root disk 8, 17 Aug 9 15:26 sdb1

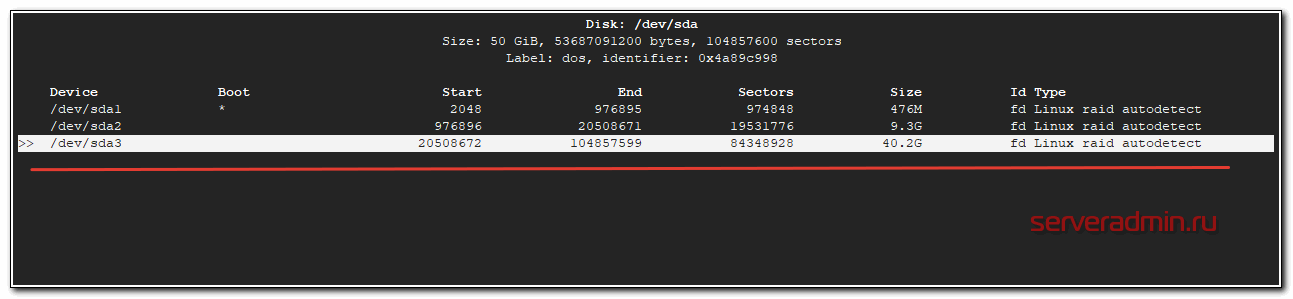
brw-rw---- 1 root disk 8, 18 Aug 9 15:26 sdb2

У меня 2 диска **sda**и **sdb**. На них уже есть по 2 раздела, которые мы сделали во время установки системы. Создадим на каждом из них еще по одному разделу из всего оставшегося свободного места.

# cfdisk /dev/sda

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-21.png)

Выбираем свободное пространство, создаем на нем раздел и указываем тип - **Linux raid autodetect**.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-22.png)

То же самое делаем со вторым диском. После этих действий у вас должны появиться новые разделы **/dev/sda3** и **/dev/sdb3**. Чтобы это произошло, нужно либо перезагрузить сервер, либо перечитать таблицу разделов командой:

# partprobe -s

Если команда не найдена, установите **parted**:

# apt install parted

Проверим новые разделы:

# ls -l /dev | grep sd

brw-rw---- 1 root disk 8, 0 Aug 9 15:40 sda

brw-rw---- 1 root disk 8, 1 Aug 9 15:40 sda1

brw-rw---- 1 root disk 8, 2 Aug 9 15:40 sda2

brw-rw---- 1 root disk 8, 3 Aug 9 15:40 sda3

brw-rw---- 1 root disk 8, 16 Aug 9 15:40 sdb

brw-rw---- 1 root disk 8, 17 Aug 9 15:40 sdb1

brw-rw---- 1 root disk 8, 18 Aug 9 15:40 sdb2

brw-rw---- 1 root disk 8, 19 Aug 9 15:40 sdb3

Теперь объединим только что созданные разделы в raid1:

# mdadm --create /dev/md2 --level=1 --raid-disks=2 /dev/sda3 /dev/sdb3

mdadm: Note: this array has metadata at the start and

may not be suitable as a boot device. If you plan to

store '/boot' on this device please ensure that

your boot-loader understands md/v1.x metadata, or use

--metadata=0.90

Continue creating array? y

mdadm: Defaulting to version 1.2 metadata

mdadm: array /dev/md2 started.

Проверим информацию о mdadm:

# cat /proc/mdstat

Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]

md2 : active raid1 sdb3[1] sda3[0]

42140672 blocks super 1.2 [2/2] [UU]

[=>...................] resync = 9.4% (4001792/42140672) finish=3.0min speed=210620K/sec

md0 : active raid1 sdb1[1] sda1[0]

486400 blocks super 1.2 [2/2] [UU]

md1 : active raid1 sdb2[1] sda2[0]

9756672 blocks super 1.2 [2/2] [UU]

Все в порядке, массив синхронизируется. Дождитесь окончания синхронизации и продолжайте. Хотя это не обязательно, массив и сейчас уже готов к работе, он он будет сильно тормозить во время ребилда. Добавим информацию о новом массиве в конфигурационный файл */etc/mdadm/mdadm.conf*, иначе после перезагрузки массив не будет виден в системе:

# mdadm --examine --scan | grep 'md/2' >> /etc/mdadm/mdadm.conf

Конфигурационный файл примет такой вид:

# cat /etc/mdadm/mdadm.conf

# mdadm.conf

# mdadm.conf

#

# !NB! Run update-initramfs -u after updating this file.

# !NB! This will ensure that initramfs has an uptodate copy.

#

# Please refer to mdadm.conf(5) for information about this file.

#

# by default (built-in), scan all partitions (/proc/partitions) and all

# containers for MD superblocks. alternatively, specify devices to scan, using

# wildcards if desired.

#DEVICE partitions containers

# automatically tag new arrays as belonging to the local system

HOMEHOST

# instruct the monitoring daemon where to send mail alerts

MAILADDR root

# definitions of existing MD arrays

ARRAY /dev/md/0 metadata=1.2 UUID=a214eb8b:c3ff0ab2:13197140:25a25924 name=proxmox6:0

ARRAY /dev/md/1 metadata=1.2 UUID=1d5a52f3:7caa923c:b8f13836:8a7bcf54 name=proxmox6:1

# This configuration was auto-generated on Fri, 09 Aug 2019 13:59:40 +0300 by mkconf

**ARRAY /dev/md/2 metadata=1.2 UUID=30be9c4e:6a351eca:6d66228d:ee695c19 name=proxmox6:2**

Дальше у нас 2 пути:

1. Создать файловую систему на md2, смонтировать раздел в систему и создать storage в виде обычной **директории**, такой же, как по-умолчанию.
2. Создать том **LVM** и на его основе storage.

Традиционно используют второй способ, так как он позволяет потом использовать в качестве дисков виртуальных машин lvm разделы. С ними удобно работать, плюс, теоретически, блочное устройство типа lvm должно работать чуть быстрее. Так что пойдем по второму пути. Выполним инициализацию раздела для работы с LVM:

# pvcreate /dev/md2

File descriptor 7 (pipe:[31619]) leaked on pvcreate invocation. Parent PID 2856: bash

Physical volume "/dev/md2" successfully created.

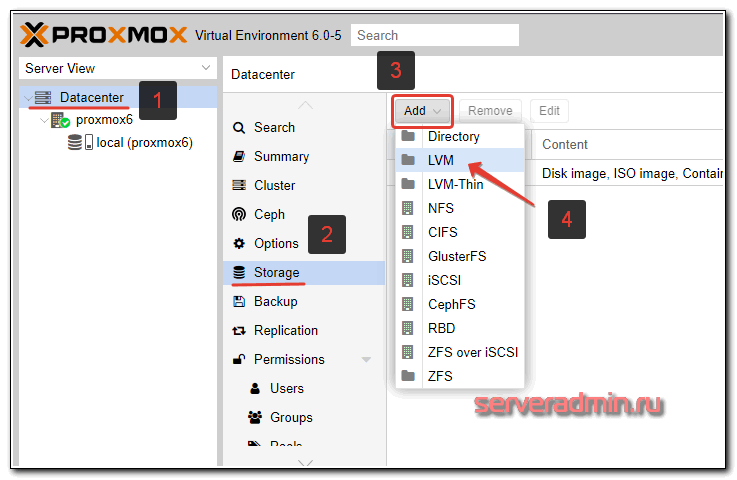
Создадим группу томов:

# vgcreate raid1-md2 /dev/md2

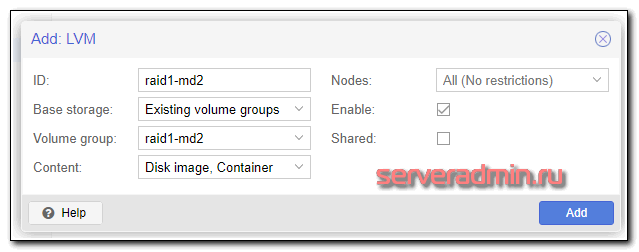
File descriptor 7 (pipe:[31619]) leaked on vgcreate invocation. Parent PID 2856: bash

Volume group "raid1-md2" successfully created

Теперь идем в web интерфейс, выбираем Datacenter, вкладку Storage, жмем Add и выбираем LVM:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-23.png)

Указываем параметры только что созданной группы томов.

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-24.png)

|  |  |
| --- | --- |
| ID | Любое название нового сторейджа. Я обычно называю так, чтобы потом можно было легко сопоставить с именем системного раздела. Так удобно. |
| Volume group | Имя группы томов LVM, которое вы указали при создании командой vgcreate |

Все готово. Новый storage добавили. Его можно использовать для хранения дисков виртуальных машин.

Добавление и подключение диска

Рассмотрим вариант добавления нового диска и использование его в качестве хранилища виртуальных машин. По большому счету, это то же самое, что мы проделали ранее с новым разделом, только теперь все указанные выше операции нужно будет проделать над новым диском. Быстро пройдем по этапам.

Я подключил новый жесткий диск, проверим его имя в системе:

# ls -l /dev | grep sd

brw-rw---- 1 root disk 8, 0 Aug 9 15:40 sda

brw-rw---- 1 root disk 8, 1 Aug 9 15:40 sda1

brw-rw---- 1 root disk 8, 2 Aug 9 15:40 sda2

brw-rw---- 1 root disk 8, 3 Aug 9 15:40 sda3

brw-rw---- 1 root disk 8, 16 Aug 9 15:40 sdb

brw-rw---- 1 root disk 8, 17 Aug 9 15:40 sdb1

brw-rw---- 1 root disk 8, 18 Aug 9 15:40 sdb2

brw-rw---- 1 root disk 8, 19 Aug 9 15:40 sdb3

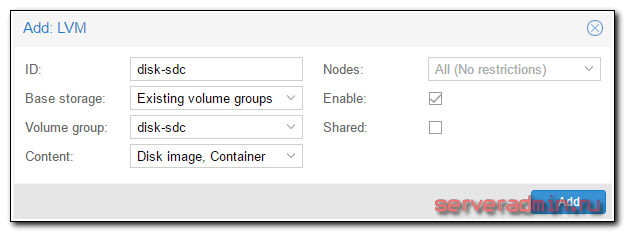
brw-rw---- 1 root disk 8, 32 Aug 9 15:40 **sdc**

Можно не создавать на диске раздел, а сразу его инициализировать для работы в lvm и создать группу томов:

# pvcreate /dev/sdc

# vgcreate disk-sdc /dev/sdc

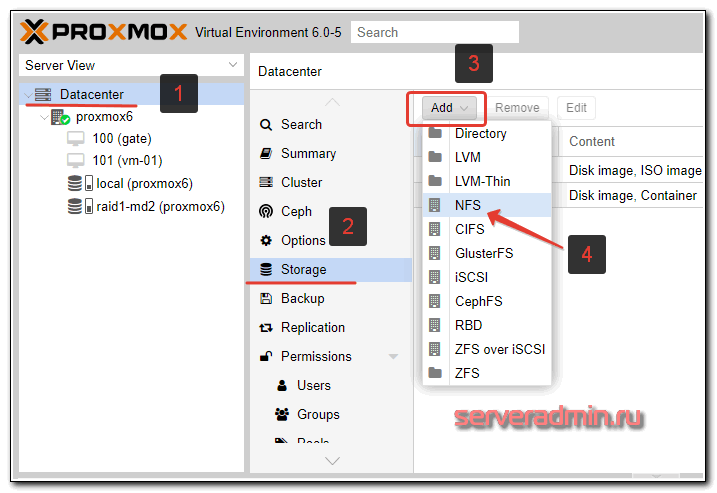
Идем в веб интерфейс и создаем новый storage:



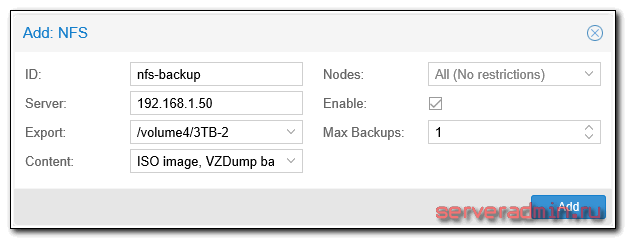
Если у вас будет не одиночный диск, а несколько, то перед этим создайте рейд массив из них и так же поверх накатывайте lvm и добавляйте новый storage. На этом настройку локальных хранилищ завершаем. Все основные действия мы выполнили, можно начинать работать с системой - создавать виртуальные машины.

 Хранилище по NFS

Приведу простой пример подключения хранилища по NFS. Мне оно в будущем понадобится для организации бэкапа виртуальных машин. Об этом я расскажу позже. Сейчас только добавим новое хранилище. Сделать это очень просто. У меня уже есть настроенный сервер NFS. Описание его настройки выходит за рамки материала, поэтому будем считать, что сервер у вас есть. Идем в раздел Storage и добавляем новое хранилище NFS:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-25.png)

Я это хранилище буду использовать только для iso образов и backup'ов, поэтому выставляю соответствующие настройки.

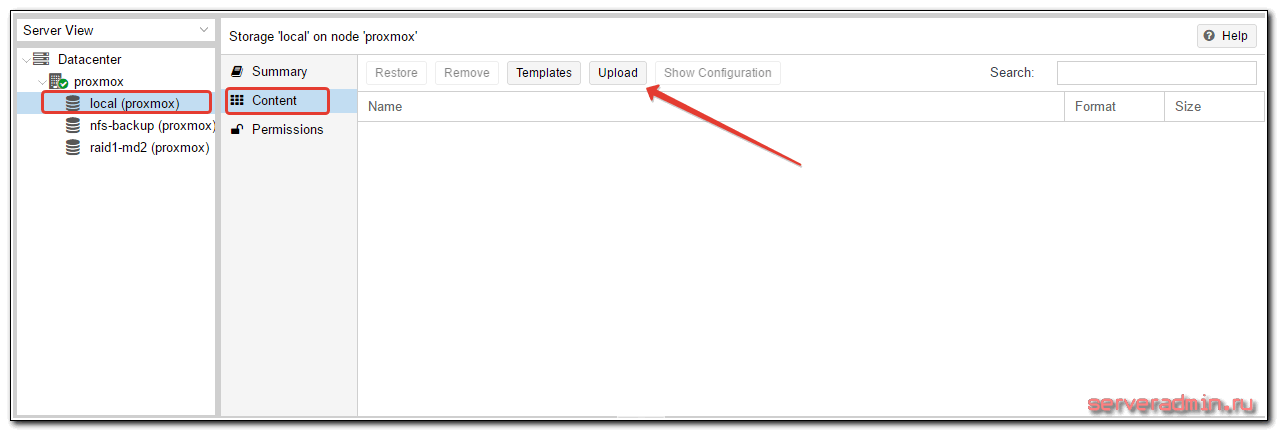


Обращаю внимание, что после указания адреса сервера, список доступных каталогов для экспорта будет автоматически запрошен с сервера. Так что нет необходимости указывать путь вручную, можно выбрать из выпадающего списка.

Создание и настройка виртуальных машин

В настройке виртуальных машин в proxmox нет ничего сложного и необычного. Я не буду по шагам рассказывать, как это сделать. Если вы дошли до этого пункта, преодолев все остальные, разобраться не составит труда. Обращаю внимание только на несколько ключевых моментов.

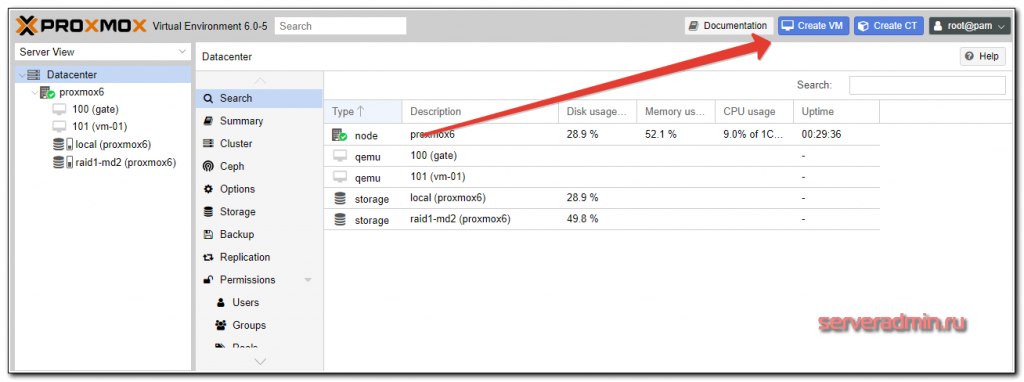
Чтобы установить виртуальную машину, нужно использовать **iso образ**. Его надо как-то загрузить на proxmox. Сделать это не сложно, нужно только понимать, что загрузить образ можно только на тот storage, который поддерживает такую возможность. Сторейджи LVM не позволяют загружать образы, для этого нужно использовать, например, storage в виде локальной директории. По умолчанию, один такой есть в системе. В него и загрузим образ виртуальной машины. Для этого открываем нужный storage и выбираем вкладку Content. Жмем на **Upload** и выбираем нужный образ.

[](https://serveradmin.ru/wp-content/uploads/2016/10/proxmox-install-24.png)

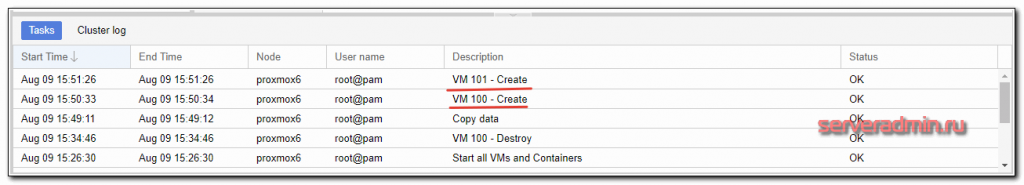
На этапе загрузки iso образа я иногда сталкиваюсь с проблемой. Образ больше 4Гб не загружается. Проверял несколько раз, в том числе в различных инсталляциях на разное железо. Ошибка время от времени проявляется. Если у вас образы больше, чем 4 Гб, ищите альтернативные способы загрузки в хранилище, благо это не составляет большого труда.

Альтернативный способ загрузки iso образа в proxmox - использовать scp или напрямую через wget качать образ из интернета. Положить его нужно в директорию */var/lib/vz/template/iso*, если используете дефолтный storage, добавленный по-умолчанию после установки.

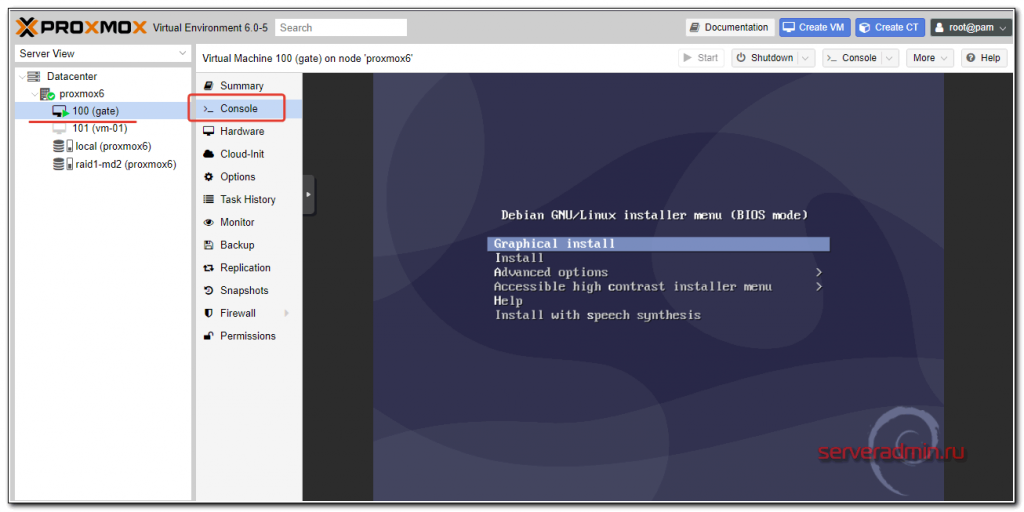
Дожидаемся окончания загрузки. Теперь этот образ можно использовать для установки ОС на виртуальную машину. Создание виртуальной машины выглядит следующим образом. Нажимаем на кнопку Create VM

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-26.png)

Заполняем все необходимые поля и стартуем машину. Некоторое время уйдет на создание машины (от нескольких секунд до 1-2 минут). В нижней части экрана ведется лог событий, там увидите информацию об окончании создания:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-27.png)

Выбирайте только что созданную машину и запускайте ее. Для того, чтобы попасть в консоль виртуальной машины, выбери соответствующий раздел:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-28.png)

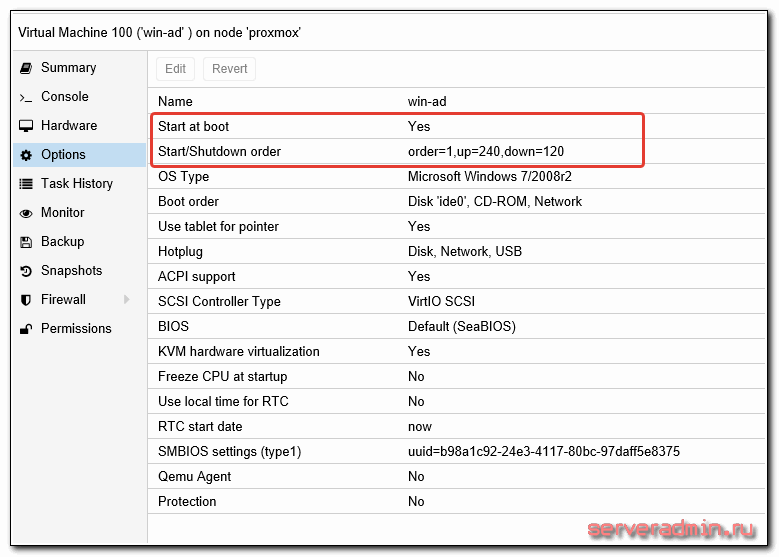
Виртуальная машина должна быть запущена, чтобы увидеть консоль. Рассказывать про настройку виртуальных машин особенно нечего. Все настройки видны в свойствах машины, выставляйте то, что вам необходимо и пользуйтесь. Отдельно рассмотрю вопрос автозапуска.

Автозапуск виртуальной машины

По-умолчанию созданные виртуальные машины не запускаются со стартом гипервизора, это нужно делать вручную. Но есть настройка, которая отвечает за автозагрузку виртуальных машин, а так же за порядок их загрузки. Вот список параметров, которыми мы можем управлять:

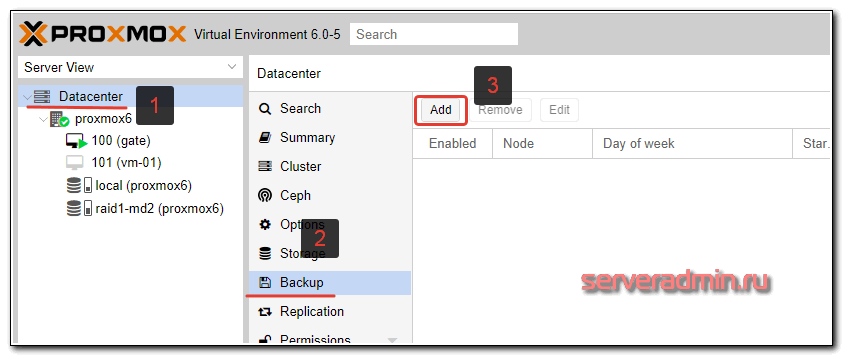
|  |  |
| --- | --- |
| Start at boot | Принимает значение Yes или No, соответственно, загружаемся автоматически или нет. |
| Start/Shutdown order | Порядок загрузки и выключения виртуальных машин, принимает целые цифровые значения (1, 2, и т.д.). |
| Startup delay | Задержка запуска остальных ВМ после запуска этой, принимает значения в секундах. |
| Shutdown timeout | Задержка в выключении ВМ. Сначала будет отдана команда на завершение работы. Если ВМ не завершит работу корректно, ей будет послан сигнал на принудительное выключение. |

Рассмотрим простой пример. У нас есть виртуальная машина, к примеру Windows Server с ролью Active Directory на борту. Все остальные серверы завязаны на корректную работу службы каталогов. Нам надо сначала запустить контроллер домена, а затем все остальные серверы за ним. Установите параметр **Start at boot** в положение Yes для всех виртуальных машин, которые хотите запускать автоматически. Укажите **Start/Shutdown order** на контроллере домена 1, **Startup delay** 240, **Shutdown timeout** 120. С такими настройками при запуске гипервизора первой стартует виртуальная машина с КД, через 240 секунд все остальные в соответствии с их настройками.

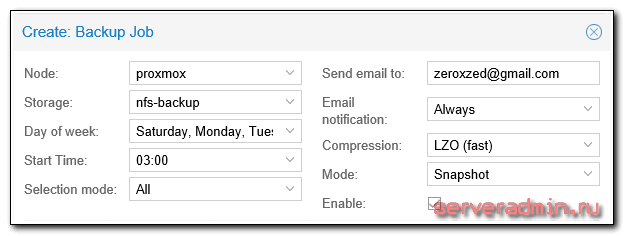


Backup виртуальных машин

Подошли к очень важному этапу настройки proxmox - организация бэкапа виртуальных машин. На этот момент нужно обратить особенное внимание, и все хорошенько проверить. Настраивается backup так же просто, как и все остальное в proxmox. Идем в раздел Backup и создаем задание:

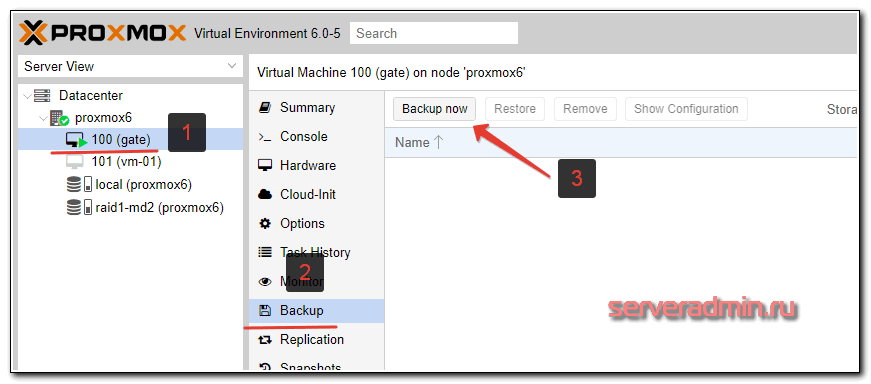
[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-29.png)

Выставляем необходимые параметры, например такие:

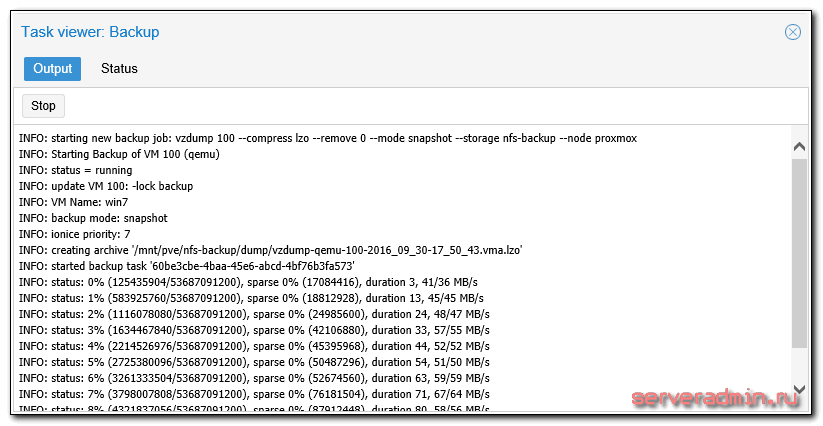


Ждем назначенного часа и проверяем исполнение задания.

Для того, чтобы выполнить разовый бэкап конкретной виртуальной машины, достаточно открыть виртуальную машины, выбрать в ней раздел Backup и нажать **Backup Now**:

[](https://serveradmin.ru/wp-content/uploads/2019/08/proxmox-configure-30.png)

Дальше указываете необходимые параметры и дожидаетесь завершения бэкапа. Обращаю внимание, что бэкап будет выполняться при работающей виртуальной машине. Останавливать ее не требуется.



Тут сразу встает следующий вопрос - как автоматически удалять старые бэкапы? Однозначно ответить не получится, все будет зависеть от того, где будут храниться бэкапы. Если это обычный linux сервер с доступом к консоли, то можно воспользоваться программой **find**:

# /usr/bin/find /mnt/backups-vm -type f -mtime +30 -exec rm {} \;

После выполнения этой команды, все файлы в папке */mnt/backups-vm* старше 30 дней будут удалены. Для автоматизации процесса команду можно поместить в **cron**. С помощью [zabbix](https://serveradmin.ru/ustanovka-i-nastroyka-zabbix-4-0/) вы можете [мониторить актуальность бэкапов](https://serveradmin.ru/monitoring-bekapov-s-pomoshhyu-zabbix/) и [следить за их размерами](https://serveradmin.ru/monitoring-razmera-bekapa-v-zabbix/).

После создания бэкапа рекомендую сразу убедиться, что из него можно восстановить виртуальную машину. Для этого откройте обзор storage, где у вас сделана копия и оттуда начните процесс восстановления. В таком случае вы сможете указать новое имя для виртуальной машины. Если вы будете пробовать восстановить из бэкапа в интерфейсе виртуальной машины, то будет предложена только замена существующей ВМ.

Не забудьте у восстановленной копии виртуальной машины, если восстановите на том же гипервизоре, поменять mac адрес и сетевые настройки. Если не поменять мак, то будете потом долго ловить и разбираться с непонятными сетевыми проблемами.

Часто задаваемые вопросы по теме статьи (FAQ)

**Как можно делать инкрементные бэкапы виртуальных машин в Proxmox?**

Проблема инкрементных бэкапов в proxmox очень актуальна. Универсального решения, к сожалению, нет. На текущий момент инкрементные бэкапы возможны только на уровне хранилища для виртуальных машин, если оно это поддерживает. К примеру, с помощью zfs это можно организовать через механизм снепшотов. Какого-то аналога для бэкапов, типа Veeam, для Proxmox не существует.

**Есть ли разница, разворачивать Proxmox на Debian либо через официальный ISO образ?**

Принципиальной разницы нет. На выходе будет одна и та же система. Другое дело, что установив сначала Debian, у вас возникает больше возможностей для преднастройки дисковой подсистемы. К примеру, вы можете установить все на массив mdadm. Штатный установщик из ISO образа не дает такой возможности.

**Чтло лучше использовать - виртуальные машины или контейнеры?**

Однозначного ответа на этот вопрос нет. Все зависит от решаемых задач. Контейнеры позволяют ускорить разворачивание и бэкап систем. Снижают потребление ресурсов за счет использования общего ядра с хостовой системой. Но это накладывает и ограничения. Если вам нужно изменить параметры ядра, то придется это делать для всего гипервизора и всех систем, работающих на его ядре. Так что тут рекомендация может быть такая - если вас устраивает функционал и ограничения контейнеров, используйте их. В остальных случаях - виртуальные машины.

**Есть ли какая-то принципиальная разница в чистом гипервизоре KVM по сравнению с Proxmox?**

Принципиальной разницы нет. По сути Proxmox - это система управления гипервизором на базе KVM. Виртуальные машины, как настройки, так и сами диски с данными, без проблем переносятся на любой другой гипервизор на базе KVM. Для этого не нужна ни конвертация данных, ни преобразование настроек.

**Можно ли в виртуальные машины Proxmox пробрасывать внешние IP адреса?**

Да, в этом нет никаких проблем. Просто делаете бридж с интерфейсом гипервизора, на который приходят эти ip адреса и добавляете интерфейсы из этого бриджа в виртуальные машины. А на них настраиваете внешние IP адреса.

**Можно ли в Proxmox пробросить USB в виртуальную машину? Например, для подключения токена или лицензионного ключа.**

Да, то можно сделать. Гипервизор KVM, на базе которого построен Proxmox, давно поддерживает возможность проброса USB портов в виртуальные машины.

Заключение

Не понравилась статья и хочешь научить меня администрировать? Пожалуйста, я люблю учиться. Комментарии в твоем распоряжении. Расскажи, как сделать правильно!

Подведем итог тому, что сделали:

* Установили гипервизор proxmox на сервер под управлением Debian на программный raid1, обеспечив отказоустойчивость системы на уровне дисков.
* Настроили виртуальную сеть для виртуальных машин.
* Подключили различные storage для выполнения задач размещения и бэкапа виртуальных машин.
* Добавили и настроили виртуальные машины.
* Организовали автоматический бэкап всех ВМ.

Я постарался выбрать наиболее востребованные функции и описать их, чтобы систему можно было использовать в реальной работе и не бояться за надежность решения. Собрал необходимый минимум, но статься все равно получилась очень большая, объемная. Я не стал ее разделять на части, чтобы была целостная картина всего того, что сделали. Тут есть еще над чем поработать, особенно актуален вопрос бэкапа. Без инкрементного бэкапа неудобно хранить резервные копии на удаленном сервере, очень большой объем передаваемых данных получается. Мне не известен способ инкрементного бэкапа виртуальных машин в proxmox, буду рад, если кто-то посоветует хорошее решение для KVM. Снэпшоты zfs не в счет.