**Виртуальные машины KVM**

**Содержание**

[Глава 6. Виртуальные машины KVM](file:///C:\Users\Adm\Documents\PVE\Ch06.html)

[Обзор KVM](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "01)

[Создание KVM](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "02)

[Создание KVM с помощью образа ISO](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0201)

[Закладка General](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "020101)

[Node](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010101)

[VM ID](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010102)

[Name](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010103)

[Resource Pool](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010104)

[Help](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010105)

[Закладка OS](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020102)

[Закладка CD/DVD](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020103)

[Закладка Hard Disk](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020104)

[Bus/Device](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010401)

[Storage](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010402)

[Disk Size (GB)](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010403)

[Format](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010404)

[Cache](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010405)

[No backup](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010406)

[Discard](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010407)

[IO thread](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010408)

[Закладка CPU](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020105)

[Sockets](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010501)

[Cores](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010502)

[Enabling NUMA](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010503)

[Type](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010504)

[Закладка Memory](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020106)

[Закладка Network](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020107)

[Bridged mode](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010701)

[Firewall](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010702)

[NAT mode](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010703)

[No network device](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010704)

[Model](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010705)

[MAC address](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010706)

[Rate limit (MB/s)](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010707)

[Multiqueues](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010708)

[Disconnect](file:///C:\Users\Adm\Documents\PVE\Ch06.html#02010709)

[Создание ВМ клонированием](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0202)

[Создание ВМ из шаблона](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0203)

[Узел назначения](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "020301)

[Режим](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "020302)

[Расширенные параметры настройки для ВМ](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "03)

[Настройка звуковых устройств](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0301)

[Настройка проброса PCI](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0302)

[Настройка проброса GPU](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0303)

[Подготовка к подключению во время работы](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0304)

[Настройка подключения во время работы](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "0305)

[Подключение vCPU в горячем режиме](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "020501)

[Подключение оперативной памяти в горячем режиме](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "020502)

[Подключение дисков/ vNIC в горячем режиме](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "020503)

[Миграция виртуальных машин KVM](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "04)

[Выводы](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch06.html" \l "05)

На текущий момент мы познакомились с графическим интерфейсом пользователя Proxmox, файлами настройки, а также структурой каталога. Мы также изучили различные типы поддерживаемых Proxmox хранилищ и то, как их интегрировать в кластер. В этой главе мы собираемся сделать шаг вперёд и рассмотреть **KVM** (**Kernel-based Virtual Machine**) и всё, что они должны предложить. Мы собираемся рассмотреть следующие темы:

* Обзор виртуальных машин KVM
* Создание виртуальных машин KVM
* Настройку виртуальных машин KVM
* Миграцию виртуальных машин KVM
* Встраиваемые виртуальные среды
* Систему резервного копирования/ восстановления Proxmox
* Моментальные снимки виртуальных машин

**Обзор KVM**

Как и подразумевает их название, KVM добавляют возможности гипервизора в ядро Linux. KVM позволяют вам создавать полностью изолированные виртуальные машины, которые не зависят от операционной системы или ядра хоста. Изолированность создаётся путём эмуляции определённых элементов оборудования, таких как ЦПУ, ОЗУ, звуковые/ видео/ сетевые карты, мосты PCI, а также устройства ввода. Чтобы создавать виртуальные машины KVM, ЦПУ в его машине хоста должен иметь **аппаратные расширения виртуализации** (**HWE**, **hardware virtualization extensions**). KVM/Qemu создаёт некий уровень, который виртуализирует физическое оборудование, что делает возможной полную виртуализацию системы в отличие от виртуализации уровня ядра, как это имеет место в случае контейнеров OpenVZ и LXC. Это делает возможной виртуализацию широкого диапазона операционных систем, таких как Linux, BSD, Windows и macOS. Одно из основных отличий KVM и виртуальных машин на основе контейнеров состоит в том, что ресурсы KVM совместно используются на аппаратном уровне, в то время как виртуальные машины уровня контейнера разделяются на уровне ядра. Таким образом, плотность общего числа ВМ KVM на узле намного меньше чем у контейнеров. KVM являются единственным выбором для не- Linux операционных систем и для ориентированных на выполнение определённой задачи операционных систем, таких как ClearOS, FreeNAS, Zentyal.

Для получения дополнительной информации по KVM обратитесь к [https://en.wikipedia.org/wiki/Kernel-based\_Virtual\_Machine](javascript:if(confirm(%27https://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine%20%20\n\nThis%20file%20was%20not%20retrieved%20by%20Teleport%20Pro,%20because%20it%20is%20addressed%20using%20an%20unsupported%20protocol%20(e.g.,%20gopher).%20%20\n\nDo%20you%20want%20to%20open%20it%20from%20the%20server?%27))window.location=%27https://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine%27) {*Прим. пер.: или к русскоязычной странице*[*https://ru.wikipedia.org/wiki/KVM*](javascript:if(confirm(%27https://ru.wikipedia.org/wiki/KVM%20%20\n\nThis%20file%20was%20not%20retrieved%20by%20Teleport%20Pro,%20because%20it%20is%20addressed%20using%20an%20unsupported%20protocol%20(e.g.,%20gopher).%20%20\n\nDo%20you%20want%20to%20open%20it%20from%20the%20server?%27))window.location=%27https://ru.wikipedia.org/wiki/KVM%27)*.*}

**Создание KVM**

В Proxmox мы можем создавать ВМ KVM следующими способами:

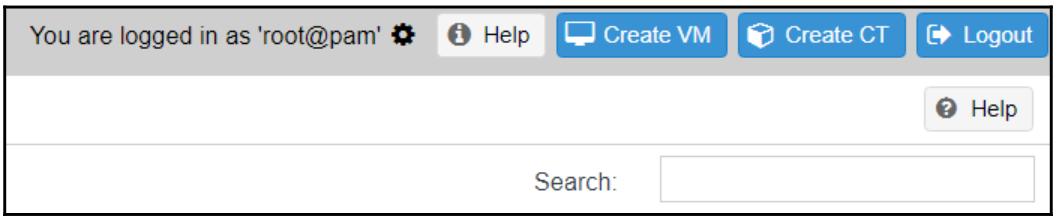
* Из рабочей области с применением ISO
* Из шаблона
* Применяя загрузку PXE

В этой главе мы намереваемся рассмотреть только создание ВИ из образов ISO и шаблонов.

**Создание KVM с помощью образа ISO**

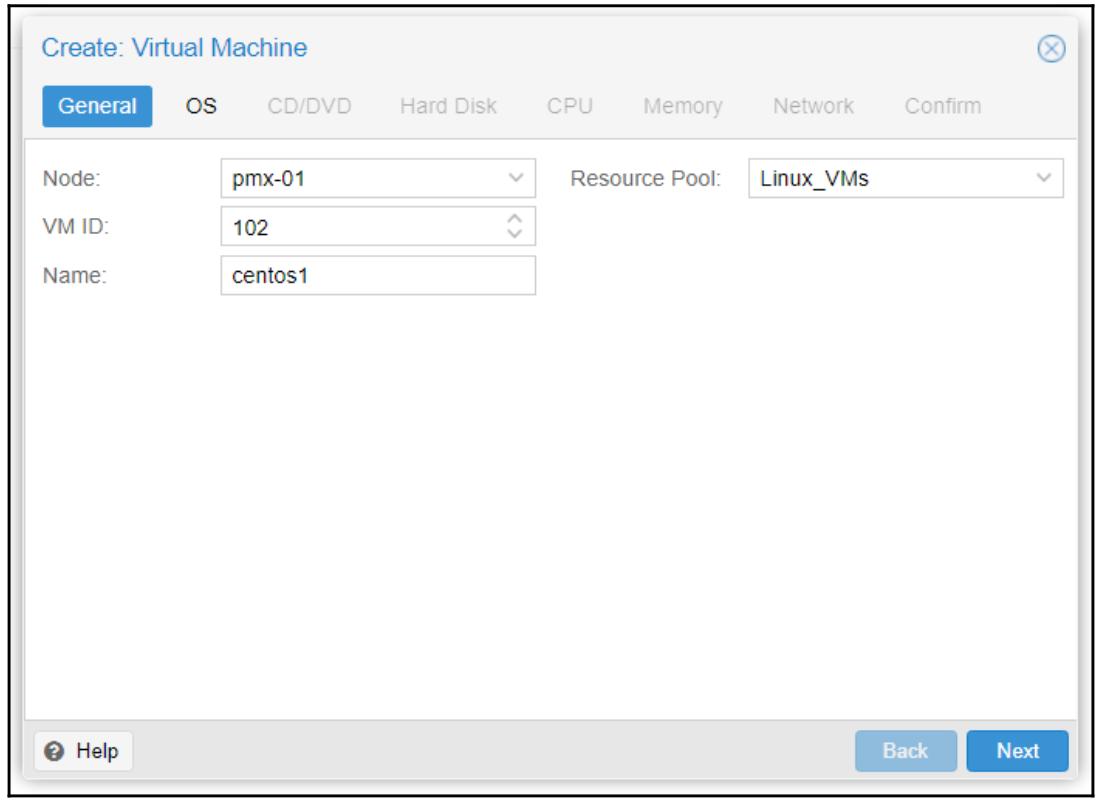
Создание ВМ основано на простых блоках диалога на базе закладок. На протяжении этого процесса мы должны назначать ресурсы и тип в необходимой информации, принадлежащей определённой ВМ. К блоку диалога можно осуществить доступ кликнув кнопку **Create VM**, расположенную в верхнем правом углу вашего экрана, как это показано на следующем экранном снимке. К нему также можно получить доступ кликнув правой кнопкой по узлу и затем выбрав **Create VM** из меню контекста:

**Рисунок 6-1**



В своём примере кластера мы собираемся создать KVM с названием **centos1** в узле **pmx-01**. Чтобы выполнить процесс построения необходимой ВМ просто кликните по кнопке **Next** или кликните по кнопке **Back** чтобы вернуться обратно в предыдущую закладку. Следующий экранный снимок отображает соответствующий блок диалога после того как мы нажали на кнопку **Create VM** в GUI Proxmox:

**Рисунок 6-2**



**Закладка General**

Закладка **General** предназначается главным образом для определения идентификационной информации. Давайте взглянем на неё.

***Node***

Это ниспадающий список для выбора того, на каком из узлов Proxmox следует создать данную ВМ.

***VM ID***

Этот текстовый блок применяется для ввода численного идентификатора данной ВМ. Мы также можем увеличивать или уменьшать текущее значение **VM ID** применяя стрелки. Если вы назначили некий идентификатор, который уже имеется в данном кластере, текущий блок отобразит красное окаймление, указывающее что уже имеется некая ВМ с тем же самым идентификатором.

***Name***

Данный текстовый блок используется для ввода необходимого названия создаваемой ВМ. Мы можем вводить строку из букв и цифр с единственным разрешённым специальным символом дефиса (**-**).

***Resource Pool***

Это ниспадающее меню применяется для выбора созданного ранее пула. Оно требуется только если вам необходимо назначить данную ВМ в определённый пул. В нашем примере ВМ мы назначаем её в пул с названием **Linux\_VMs**.

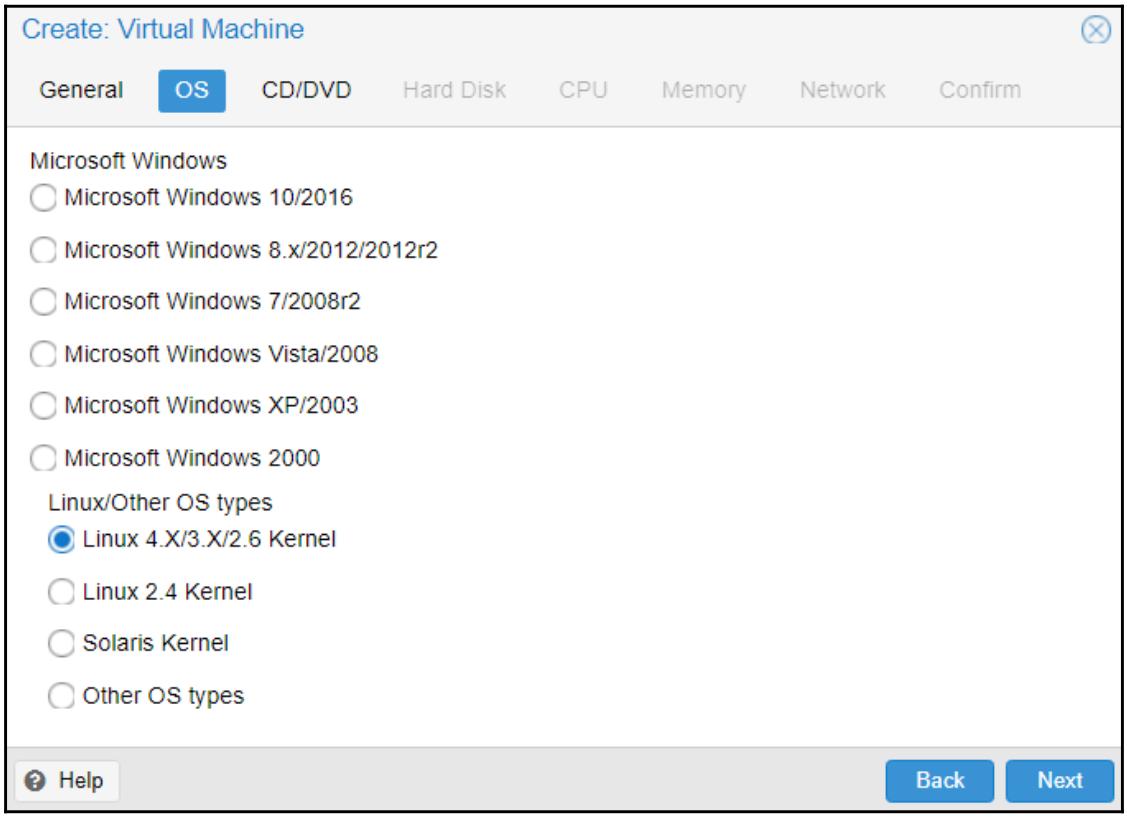
***Help***

Кнопка **Help** откроет новую закладку с установленной документацией, созданной разработчиками Proxmox. Данная документация содержит специфичную информацию, относящуюся к данной закладке. Каждая кнопка **Help** в различных закладках закреплена за определённым разделом имеющейся документации. URL данной документации является [https://ip\_addr:8006/pve-docs/chapter-qm.html](javascript:if(confirm(%27https://ip_addr:8006/pve-docs/chapter-qm.html%20%20\n\nThis%20file%20was%20not%20retrieved%20by%20Teleport%20Pro,%20because%20it%20is%20addressed%20using%20an%20unsupported%20protocol%20(e.g.,%20gopher).%20%20\n\nDo%20you%20want%20to%20open%20it%20from%20the%20server?%27))window.location=%27https://ip_addr:8006/pve-docs/chapter-qm.html%27).

**Закладка OS**

Закладка **OS** применяется для выбора типа гостевой операционной системы, которая будет установлена в данной ВМ. Этот вид выбора делает возможным уведомление строящейся ВМ о той операционной системе, которую мы собираемся устанавливать, и осуществлять регулировки текущей архитектуры на основе выбираемой ОС. В нашем примере ВМ мы выбираем **Linux 4.X/3.X/2.6 Kernel**, как это отображено на снимке экрана ниже:

**Рисунок 6-3**

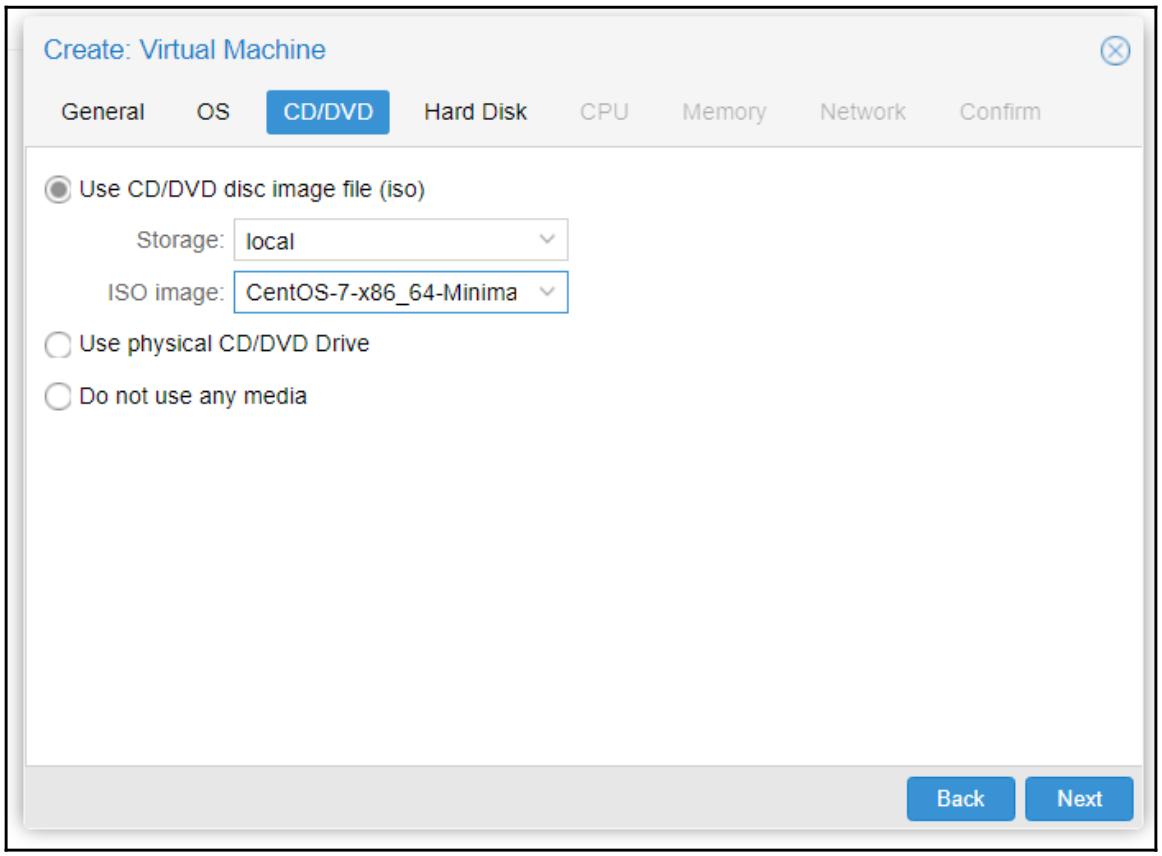


|  |  |
| --- | --- |
| [Совет] | **Совет** |
| Для достижения максимальных производительности и надёжности настоятельно рекомендуется выбирать соответствующий тип ОС. |

**Закладка CD/DVD**

Так как ВМ KVM полностью инкапсулирована и эмулирует некую физическую машину, мы можем загружать эту ВМ или операционную систему только с помощью установленных в виртуальное устройство CD/ DVD образов ISO или через физический диск, подключённый к данному узлу хоста Proxmox. В этой закладке мы можем выбрать будем ли мы применять виртуальный или физический диск CD/ DVD, либо выбрать некий образ ISO. Приводимый ниже снимок экрана отображает блок диалога для закладки **CD/DVD**, в которой мы выбрали ISO CentOS:

**Рисунок 6-4**

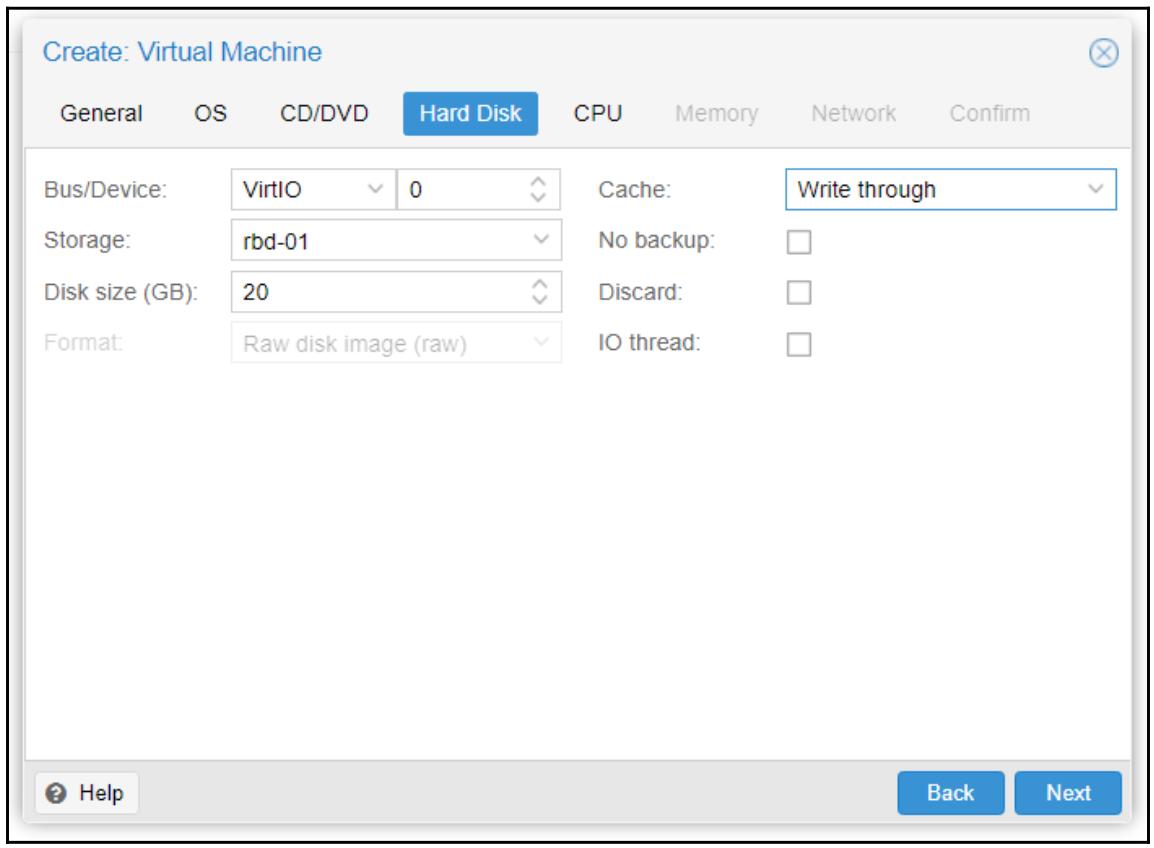


Если мы желаем создать свою ВМ без определения какого- либо образа диска, нам необходимо остановиться на опции **Do not use any media**.

**Закладка Hard Disk**

В этой закладке мы определяем все настройки для своего самого первого образа диска ВМ. Снимок экрана ниже показывает блок диалога с настройкой для нашего примера ВМ.

**Рисунок 6-5**



***Bus/Device***

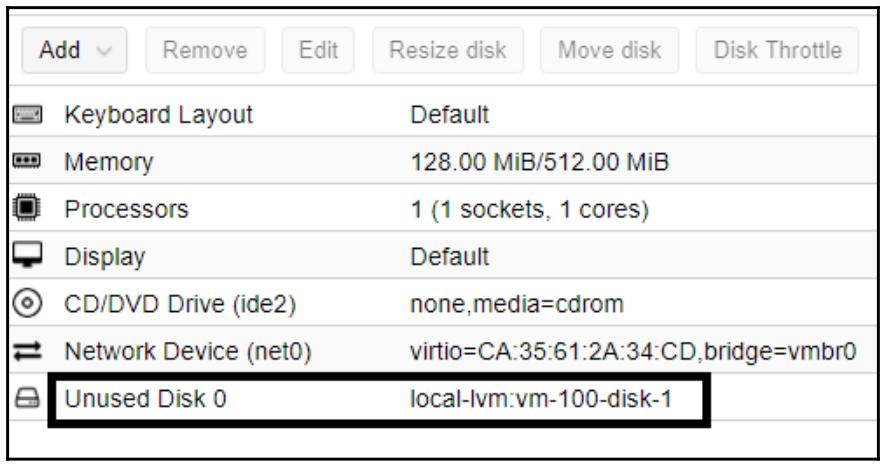
Для данной опции доступно два ниспадающих меню. Одно для выбора *disk image bus type* (типа шины образа диска) и другое для *device ID* (идентификатора устройства).

|  |  |
| --- | --- |
| [Совет] | **Совет** |
| Для максимальной производительности рекомендуется шина VirtIO. |

Для ВМ Windows необходимо выбирать IDE, поскольку Windows не имеет встроенного драйвера для VirtIO. В подобных случаях нам мы можем воспользоваться следующими шагами для добавления возможностей VirtIO в ВМ Windows:

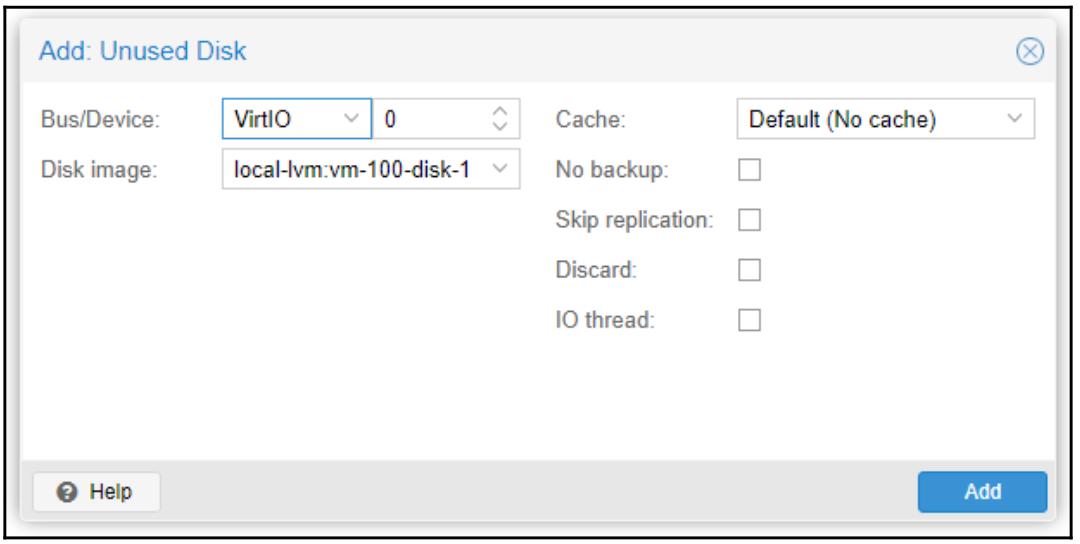
1. Создайте необходимую ВМ с IDE и установите Windows обычным образом.
2. Добавьте второй дисковый образ с требуемой шиной VirtIO и перезагрузитесь в Windows.
3. Выгрузите самый последний драйвер VirtIO для Windows со следующих ссылок и затем загрузите его через виртуальный диск CD:
   * [https:/​/​fedorapeople.​org/​groups/​virt/​virtio-​win/​direct-downloads/​stable-​virtio/​virtio-​win.​iso](javascript:if(confirm(%27https://onreader.mdl.ru:80/​/​fedorapeople.​org/​groups/​virt/​virtio-​win/​direct-downloads/​stable-​virtio/​virtio-​win.​iso%20%20\n\nThis%20file%20was%20not%20retrieved%20by%20Teleport%20Pro,%20because%20it%20is%20addressed%20using%20an%20unsupported%20protocol%20(e.g.,%20gopher).%20%20\n\nDo%20you%20want%20to%20open%20it%20from%20the%20server?%27))window.location=%27https://onreader.mdl.ru:80/​/​fedorapeople.​org/​groups/​virt/​virtio-​win/​direct-downloads/​stable-​virtio/​virtio-​win.​iso%27)
   * [http:/​/​www.​linux-​kvm.​org/​page/​WindowsGuestDrivers/​Download\_Drivers](javascript:if(confirm(%27http://onreader.mdl.ru/​/​www.​linux-​kvm.​org/​page/​WindowsGuestDrivers/​Download_Drivers%20%20\n\nThis%20file%20was%20not%20retrieved%20by%20Teleport%20Pro,%20because%20it%20is%20addressed%20on%20a%20domain%20or%20path%20outside%20the%20boundaries%20set%20for%20its%20Starting%20Address.%20%20\n\nDo%20you%20want%20to%20open%20it%20from%20the%20server?%27))window.location=%27http://onreader.mdl.ru/​/​www.​linux-​kvm.​org/​page/​WindowsGuestDrivers/​Download_Drivers%27)
4. Обновите соответствующий драйвер для вновь обнаруженного оборудования под устанавливаемый образ диска VirtIO.
5. Погасите свою ВМ Windows и зарегистрируйтесь в своей инструментальной панели Proxmox.
6. В закладке **Hardware** Данной ВМ выберите установленный образ IDE и кликните **Remove**. Заметим, что это не удаляет окончательно указанный образ диска. Такой образ диска теперь будет отображаться как **Unused Disk 0**:

**Рисунок 6-6**



1. Выберите **Unused disk 0** и кликните **Edit**. Это откроет блок диалога с вариантами для выбора необходимого типа **Bus/Device** и прочих вариантов настройки:

**Рисунок 6-7**



1. В этом блоке диалога мы можем выбрать желаемый тип шины и прочие варианты настроек, если они требуются.
2. Кликните по кнопке **Add** для добавления соответствующего образа диска обратно в эту ВМ.

Перечисленные шаги нужны для включения в вашей ВМ Windows использования образа диска VirtIO. Раз соответствующий драйвер уже загружен, нт необходимости загружать его для дополнительных образов дисков VirtIO.

***Storage***

Это ниспадающее меню для выбора соответствующего хранилища, в котором должен сохраняться необходимый образ диска. Одновременно с названием такого хранилища данное ниспадающее меню также отображает общую ёмкость и доступное пространство хранения для подключаемых устройств хранения.

***Disk Size (GB)***

Это текстовый блок для определения необходимого размера данного устройтва хранения в ГБ. Его значение может бвть только численным. Мы также можем применять стрелки данного текстового блока вверх и вниз для определения требуемого размера диска.

***Format***

Это ниспадающее меню для выбора типа образа диска. Если мы выбираем хранилище, которое поддерживает только определённый тип диска, тогда варианты этого меню будут отображаться серыми. Например, для нашего примера кластера у нас является выбранным хранилище Ceph RBD, которое может хранить для Proxmox 5.0 только образы **.raw**. Поэтому варианты формата показаны серыми.

Если мы выбрали необходимый формат образа диска неверно или позднее наши требования изменились на другой формат, мы можем просто воспользоваться вариантом **Move disk** в соответствующей закладке **Hardware** для изменения соответствующего формата. Это также можно сделать применив CLI воспользовавшись следующим форматом:

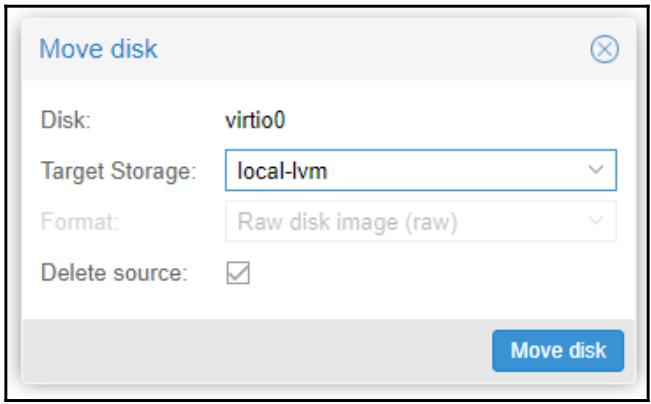
**#qemu-img convert -O <type> <source\_image> <destination\_image>**

Если мы желаем преобразовать образ диска **.qcow2** в **.raw**, команда должна быть следующей:

**#qemu-img convert -O raw vm-101-disk-1.qcow2 vm-101-disk-1.raw**

Данная команда великолепно работает для локального хранилища и хранилищ NFS, ZFS и Gluster, однако не применима для RBD. Для изменения соответствующего формата образа диска, хранимого в RBD, воспользуйтесь опцией **Move disk** в своей инструментальной панели Proxmox. Помимо хранимого в RBD образа диска данный вариант **Move disk** может быть применён для перемещения любого образа диска хранимого в любом хранилище через имеющийся GUI вообще без применения какого- либо CLI. Этот вариант также полезен для перемещения образа диска с одного хранилища на другое без отключения соответствующей ВМ. Перемещение можно выполнять с локального хранилища на совместно используемое и наоборот. Для перемещения образа диска или изменения его формата выберите необходимый образ диска в соответствующей закладке **Hardware** и кликните по **Move disk** для открытия блока диалога:

**Рисунок 6-8**



Как показано в предыдущем снимке экрана, для своего примера мы перемещаем некий образ диска **.raw** с RBD в локальное хранилище. Если мы выберем вариант **Delete source**, он удалит соответствующий файл источника автоматически после окончания преобразования или перемещения. Если эта опция не выбрана, тогда нам придётся удалять такой файл источника вручную. В этом случае файл источника будет отображаться для нас как не используемый образ диска в соответствующей закладке **Hardware** нашей ВМ.

***Cache***

Данное ниспадающее меню позволяет нам выбирать требующийся метод кэширования, применяемый для данного образа диска. Мы уже изучили различные варианты кэширования в разделе [Кэширование образа виртуального диска](file:///C:\Users\Adm\Documents\PVE\Ch04.html#030304) [Главы 4, Системы хранения](file:///C:\Users\Adm\Documents\PVE\Ch04.html). Мы можем изменять необходимый вариант кэширования в любое время, даже после того, как наша ВМ полностью создана и находится в рабочем состоянии. После каждого изменения варианта кэширования нам потребуется цикл отключения- включения данной ВМ для разрешения нового варианта кэширования.

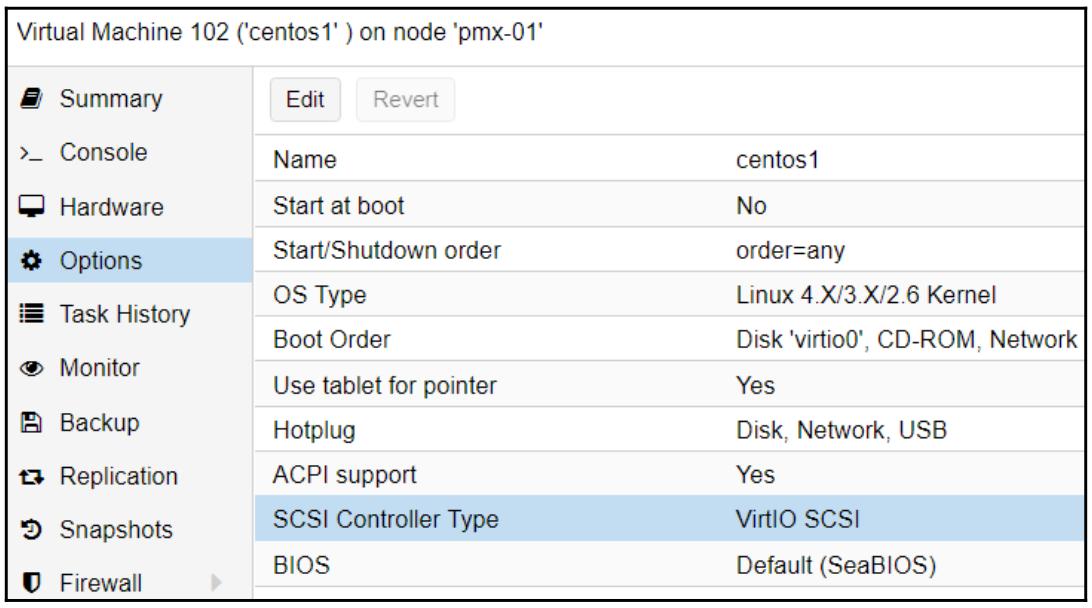
***No backup***

Если включена данная опция, соответствующий образ виртуального диска никогда не будет включаться в резервное копирование. По умолчанию данная опция отключена.

***Discard***

Образы диска в Proxmox являются разряженными (sparsed) вне зависимости от типа самого образа, что означает, что такой образ диска медленно растёт по мере того, как данные сохраняются в нём. С течением времени данные создаются и уничтожаются внутри файловой системы данного образа диска. Однако в разряженном образе диска, даже после удаления данных, они никогда не возвращают своё свободное пространство. Сама ВМ может выдавать в отчёте правильное свободное пространство хранения, однако хранилище Proxmox будет показывать более высокое использование пространства. Имеющаяся опция **Discard** делает возможным для определённого узла возвращать обратно всё свободное пространство, которое не содержит никаких данных. Это эквивалентно опции TRIM, которая была введена для устройств SSD. Прежде чем воспользоваться этим вариантом, мы обязаны убедиться что данные ВМ применяют контроллер **VirtIO SCSI**. Мы можем установить **SCSI Controller Type** в закладке **Options** виртуальной машины:

**Рисунок 6-9**



Опция **Discard**может быть не применимой в некоторых средах, решениях хранения и операционных системах. Проведите исчерпывающее тестирование прежде чем реализовывать её в своей среде. В некоторых случаях данная опция **Discard** может вызывать блокирование ВМ, что потребует цикла выключения- включения. Сама ВМ потребует цикла выключения- включения чтобы разрешить эту опцию после повторного включения.

***IO thread***

Для образов диска в KVM имеется два варианта:

* **IO thread**
* **io=native**

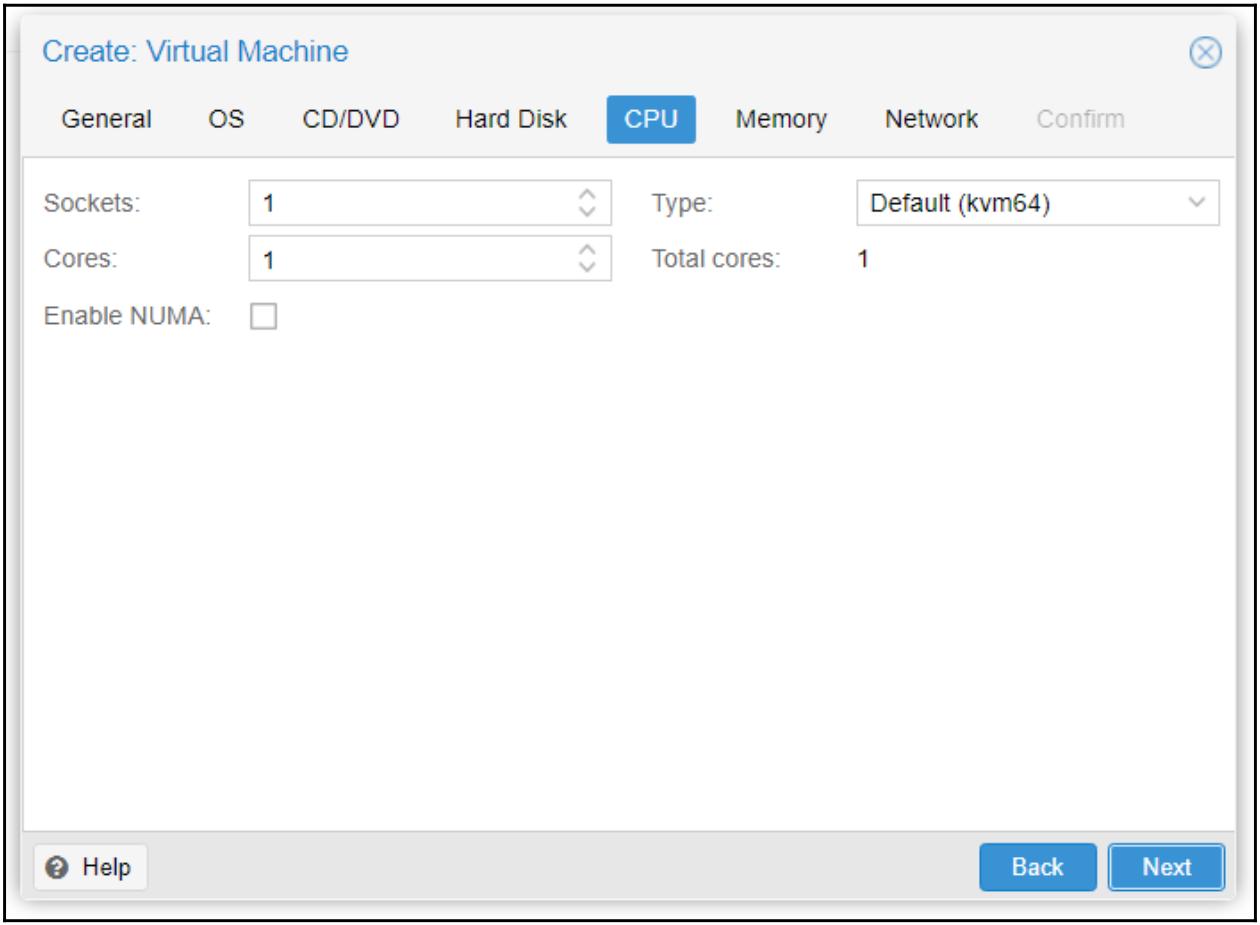
По умолчанию Proxmox применяет **io=native** для всех образов дисков пока имеющаяся опция **IO thread** специально не отмечается для определённого диска.

Такая опция **IO thread** позволяет каждому образу диска иметь свои собственные потоки вместо ожидания в очереди с кем- нибудь ещё. Так как дисковый ввод/ вывод больше не осуществляет ожидания, поскольку имеет свои собственные потоки, он не задерживает прочие задачи или очереди, относящиеся к этой ВМ, что в свою очередь ускоряет общую производительность этой ВМ помимо всего прочего, предоставляя увеличенную производительность диска. Эта опция **IO thread** является относительно новой в Proxmox. Имеется несколько экземпляров сообщений, в которых из- за данной опции происходила блокировка ВМ. Таким образом, проводите тщательное тестирование прежде чем реализуете эту функцию в некоторой промышленной среде.

**Закладка CPU**

данная закладка позволяет настраивать виртуальные ЦПУ для виртуальных машин. Приводимый далее экранный снимок отображает блок диалога с доступными опциями ЦПУ:

**Рисунок 6-10**



***Sockets***

Эта опция служит для определения общего числа сокетов, которое может применяться в данной ВМ. Для определённой ВМ мы можем использовать более одного сокета даже если её физический узел не имеет достаточного числа сокетов. Это может быть полезным только в том случае, если некое приложение в этой ВМ требует от нас более одного сокета. Но это вовсе не играет в пользу увеличения производительности ВМ в машине с единственным сокетом.

***Cores***

Данная опция служит для определения общего числа ядер, которое может применять данная ВМ. Хорошим практическим приёмом является начало применения какой- то ВМ с меньшим числом ядер и затем последовательно увеличивать их по мере необходимости в зависимости от нагрузки. Выделение большого числа ядер некоторой ВМ вызовет ненужную нагрузку на все доступные ресурсы в данном узле. Обычно некая ВМ может предоставлять хорошую производительность с двумя или четырьмя ядрами если только она не является ВМ с высокими запросами, такими как сервер удалённых рабочих мест или сервер SQL/ Exchange.

***Enabling NUMA***

Доступ к неоднородной памяти (**NUMA**, **non-uniform memory access**) не является каким- то новым подходом для работы с памятью в среде со множеством ЦПУ, однако это новое добавление в Proxmox VE. С применением NUMA оперативная память может равномерно распространяться между ЦПУ, что увеличивает производительность, поскольку более нет узкого места в том, что все ЦПУ пытаются осуществлять доступ к одним и тем же банкам памяти. В Proxmox опция NUMA также разрешает подключение в горячем режиме памяти и ЦПУ. Без этой опции подключение оперативной памяти и ЦПУ не работает вовсе.

Любой узел с более чем одним сокетом обычно осведомлён об NUMA. Следовательно, включение NUMA для ВМ в таком узле привнесёт преимущества в производительность ВМ. NUMA всегда пытается удерживать определённую ВМ в одном и том же пакете ЦПУ. Мы можем проверять текущее состояние NUMA в своём кластере Proxmox с применением такой команды:

**# numastat**

Эта команда отобразит все те узлы в вашем кластере, которые осведомлены о NUMA, а также их состояния производительности.

***Type***

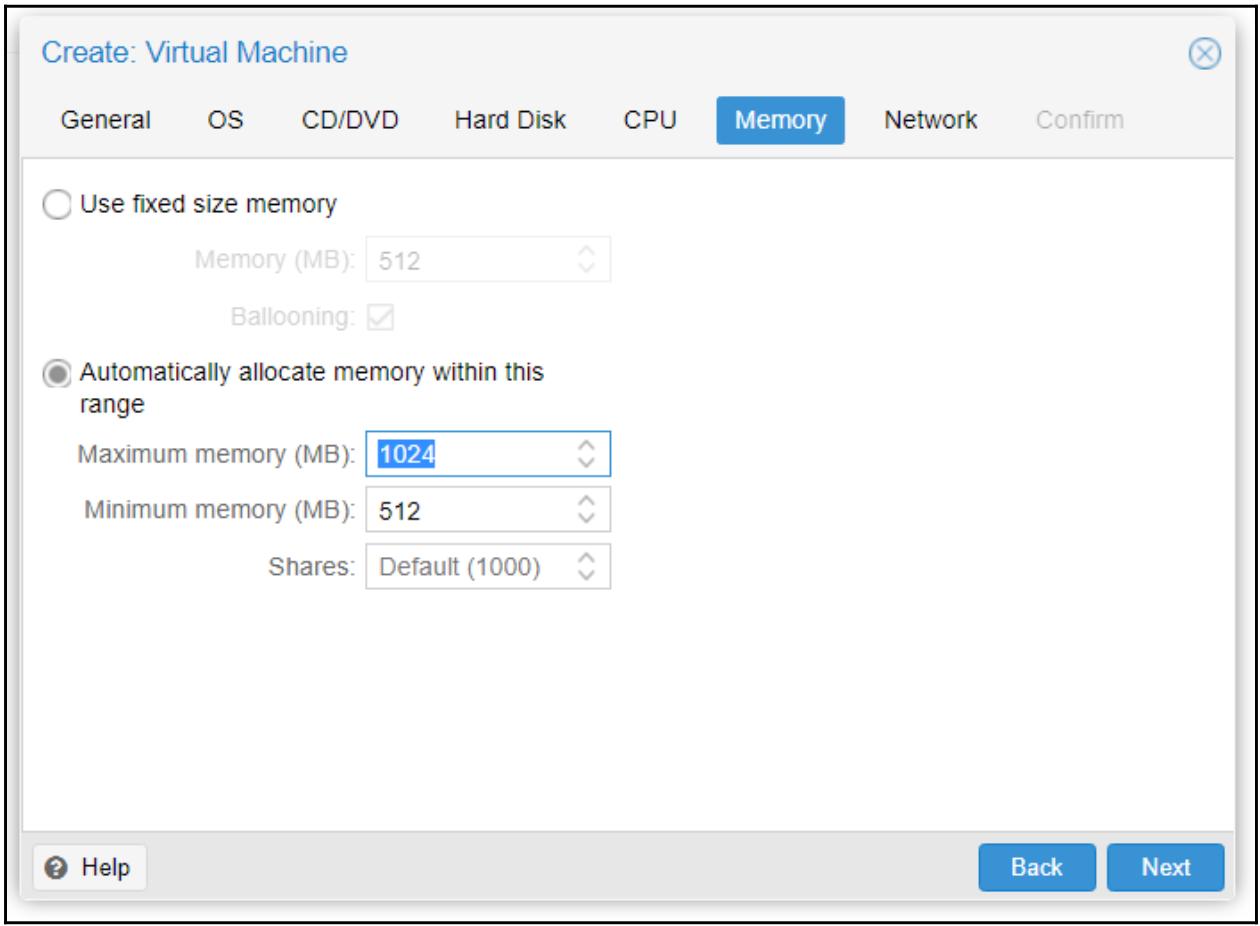
Это наспадающее меню для выбора типа пакета ЦПУ. По умолчанию для всех ВМ выбирается тип ЦПУ **Default (kvm64)**. Обычным вариантом применения является случай, когда определённое приложение требует инструкции SSE или AVX. Выбирая тип ЦПУ определённого хоста мы даём ВМ прямой доступ к физическому ЦПУ.

|  |  |
| --- | --- |
| [Совет] | **Совет** |
| Для получения наилучшей производительности следует применять тип хоста. Тем самым ваши ВМ будут способны осуществлять прямой доступ к имеющимся ЦПУ без какого бы то ни было уровня эмуляции. Именно это является оптимальным видом для некоторой среды, в которой все узлы являются идентичными. Для получения максимальной переносимости какой- то ВМ следует выбирать типы ЦПУ KVM или Qemu. |

**Закладка Memory**

Эта закладка позволяет настраивать выделение оперативной памяти определённой ВМ. Приводимый ниже моментальный снимок экрана отображает блок диалога для нашего примера ВМ:

**Рисунок 6-11**

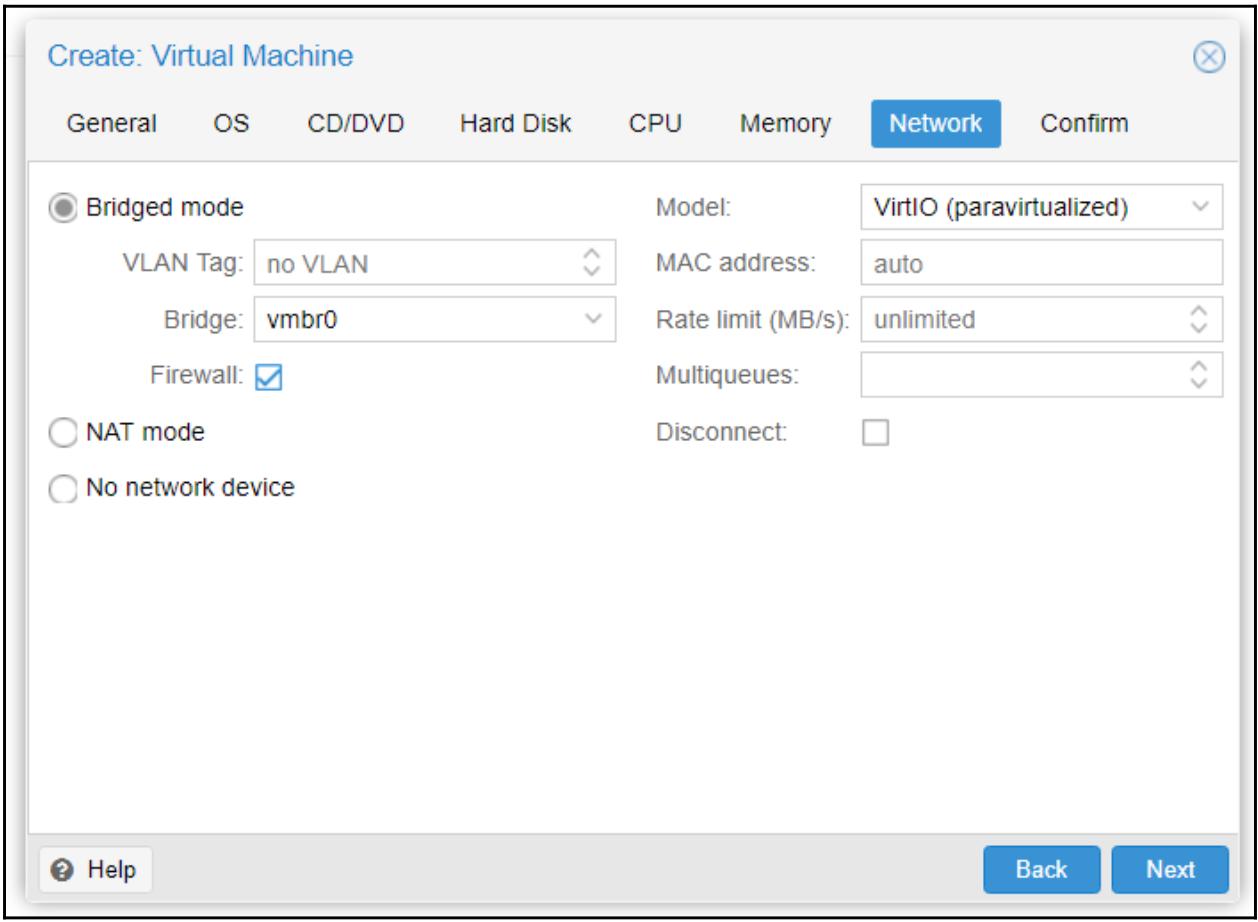


В Proxmox мы можем устанавливать для ВМ фиксированную или динамическую оперативную память. Автоматический диапазон также именуется как раздувание (ballooning) памяти. При установленном варианте с фиксированной памятью вся оперативная память выделяется одномоментно. При установке динамической памяти оперативная память выделяется на основе требований ВМ, но в рамках установленного диапазона. Автоматическое выделение памяти великолепно работает вВМ Linux, однако для ВМ Windows раздувание памяти потребляет боле высокие объёмы ресурсов ЦПУ, что вызывает замедление такой ВМ. Таким образом, для ВМ Windows рекомендуется везде где это возможно применять фиксированную память.

**Закладка Network**

Данная закладка делает возможной настройку текущего виртуального сетевого интерфейса для определённой ВМ. Приводимый далее снимок экрана отображает блок диалога настройки сети для нашего примера ВМ:

**Рисунок 6-12**



***Bridged mode***

Этот режим позволяет ВМ подключаться к сетевой среде с применением моста. Такая ВМ не получает прямого доступа к имеющейся внешней сетевой среде. Мы можем установить VLAN ID на уровне самого узла, что сделает невозможной его настройку внутри такой ВМ. Данный Режим моста (**Bridged mode**) также предоставляет варианты межсетевого экрана для своей ВМ. Для нашего примера мы выбрали имеющийся по умолчанию мост **vmbr0** и включили его опцию **Firewall**.

***Firewall***

Чтобы разрешить межсетевой экран для сетевых интерфейсов следует пометить данную опцию. Без данной опции никаие правила межсетевого экрана не будут применяться к данному интерфейсу ВМ. Более подробно мы рассмотрим имеющийся межсетевой экран Proxmox в [Главе 9, Межсетевой экран Proxmox VE](file:///C:\Users\Adm\Documents\PVE\Ch09.html).

***NAT mode***

Это режим предоставляет ВМ прямой доступ ко внешним сетевым средам. Сентевой обмен не пропускается ни через какие мосты. Если в данной сетевой среде применяется VLAN, он должен быть настроен внутри такой ВМ чтобы иметь все пакеты данных помеченными или не помеченными. Опция межсетевого экрана Proxmox не доступна при использовании режима NAT.

***No network device***

Данный вариант создаст определённую ВМ без каког бы то ни было настроенного сетевого интерфейса.

***Model***

Это ниспадающее меню для выбора типа интерфейса виртуальной сетевой среды. Для максимальной сетевой производительности настоятельно рекомендуется применение VirtIO. Windows не поставляется с каким бы то ни было драйвером VirtIO. Следовательно, если он применяется к ВМ Windows, нам следует вручную загрузить этот драйвер из ISO, который мы выгрузили в разделе [Закладка Hard Disk](file:///C:\Users\Adm\Documents\PVE\Ch06.html#020104) этой главы. Для ВМ Windows мы также можем воспользоваться Intel E1000. Начиная с Windows 7 и выше этот драйвер для Intel включён в поставку.

***MAC address***

По умолчанию все MAC адреса для виртуальных сетевых интерфейсов назначаются автоматически. Набрав некий MAC адрес в данном текстовом блоке мы можем определить конкретный MAC адрес для данного интерфейса. Это может быть полезным в случае, когда определённому приложению в данной гостевой ВМ требуется какой- то специфический MAC адрес.

***Rate limit (MB/s)***

Именно этот текстовый блок служит для определения максимально допустимой скорости сетевого интерфейса, определяемой в Мегабайтах в секунду. Это очень полезная опция для ограничения сетевых ресурсов в ВМ. Без установки какого бы то ни было значения данная ВМ будет пытаться использовать максимальную доступную полосу пропускания.

***Multiqueues***

Обычно ВМ KVM имеют единственную очередь в которой отправляются и принимаются все пакеты появляющиеся со временем, причём не применяется никакое распараллеливание. Множественные очереди (**Multiqueues**) удаляют такое узкое место, делая возможной одновременные отправку и приём, применяя для параллельных очередей виртуальные ядра ЦПУ. **Множественные очереди** особенно полезны для ВМ, которые активны с большим числом клиентов, например, для некоторого веб сервера. В закладке Proxmox **Network** в блоке диалога создания определённой ВМ мы можем ввести численное значение для определения того сколько данная ВМ должна использовать очередей одновременно. Данное значение не должно превосходить общее число выделенных для этой ВМ vCPU. Например, если данная ВМ имеет число ядер равное четырём, мы можем установить значение **Multiqueues** равное **4**. **Множественные очереди** существенно повышают производительность сетевой среды ВМ поскольку и отправка и приём могут осуществляться одновременно.

|  |  |
| --- | --- |
| [Совет] | **Совет** |
| Имейте в виду, что **Множественные очереди** также увеличивают использование ЦПУ данной ВМ, так как каждая очередь находится в зависимости от каждого vCPU. |

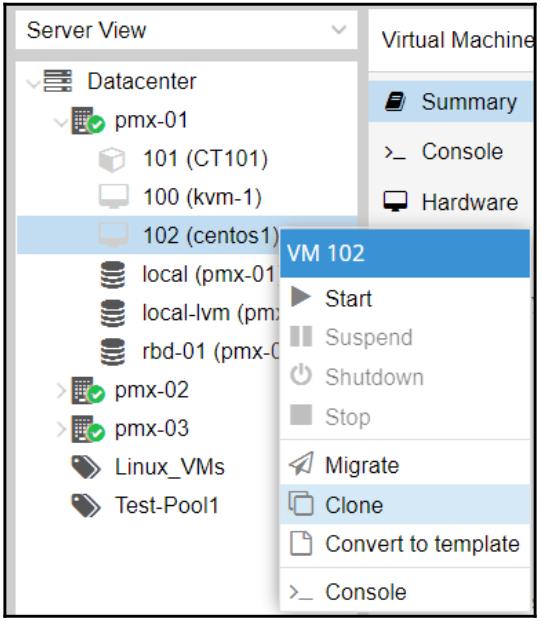
***Disconnect***

Если включена данная опция, определяемый виртуальный сетевой интерфейс будет создан совместно с данной ВМ, однако не будет активирован.

**Создание ВМ клонированием**

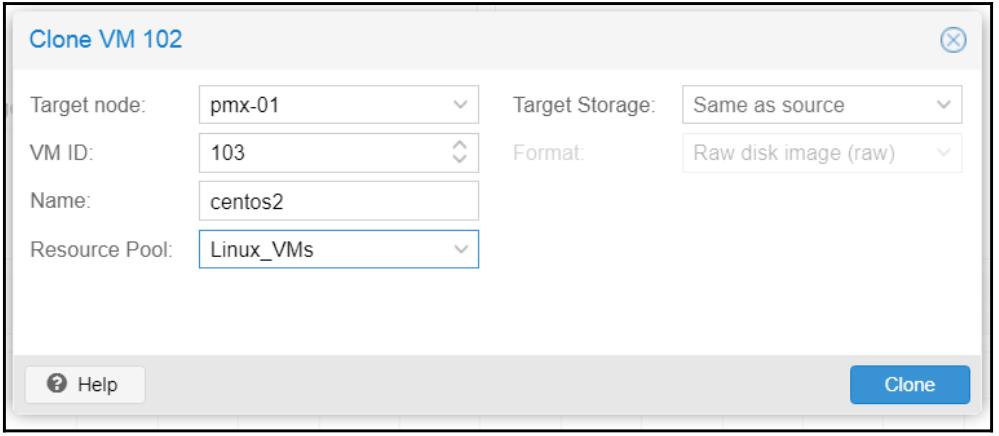
При развёртывании множества ВМ с идентичными настройками их индивидуальное создание может стать временеёмким процессом. В данном случае мы можем клонировать некую имеющуюся ВМ или какой- то шаблон. Клонирование создаёт полностью независимую ВМ с идентичной конфигурацией. Такая клонированная ВМ ни коим образом не связана с той ВМ из которой она клонирована. К такой опции клонирования можно получить доступ из меню контекста кликнув правой кнопкой по той ВМ, которая подлежит клонированию:

**Рисунок 6-13**



Одним из вариантов применения клонирования является некая стратегия резервного копирования. Какая- то ВМ может клонироваться в отдельные узлы, и даже в отдельную систему хранения, на постоянной основе. В случае возникновения события, при котором её основной узел становится недоступным, такая клонированная ВМ может быть поднята и запущена за считанные минуты без какой либо необходимости осуществления процесса восстановления. Следующий моментальный снимок отображает блок диалога подобного клонирования после клика по **Clone** в меню имеющегося контекста:

**Рисунок 6-14**



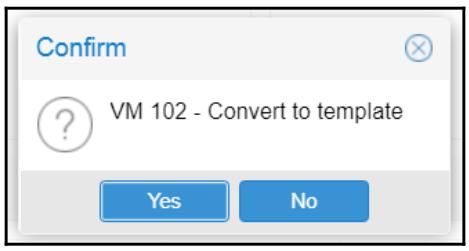
или для добавления новых приложений и тому подобного, так как первоначальная ВМ всё ещё остаётся полностью работающей ВМ.

**Создание ВМ из шаблона**

Аналогично клонированию некий шаблон также является быстрым способом развёртывания полностью настроенных ВМ без прохождения через процесс полного создания ВМ и установки вручную ОС и приложений. Мы можем создать некую новую ВМ и установить необходимые ОС и прочие необходимые программы перед их преобразованием в какой- то шаблон. Таким образом, все вновь создаваемые из данного шаблона ВМ будут с полностью настроенными ОС и программами. Что устанавливает отличие шаблона от простого клонирования, так это то, что после преобразования какой- то ВМ в шаблон она не может быть вновь включена. Если для данного шаблона ВМ требуются какие- либо изменения, необходимо создать новую ВМ, настроить её и затем преобразовать в шаблон. Однако, мы можем изменять имеющиеся аппаратные ресурсы конкретного шаблона. Основным преимуществом применения шаблона в сравнении с клонированными ВМ состоит в том, что шаблон допускает для нас как создание полностью клонированных ВМ, так и ВМ с присоединённым клоном.

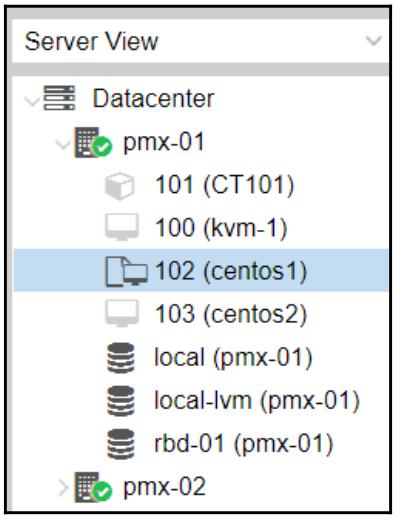
для создания ВМ из какого- то клона нам потребуется вначале создать такой шаблон. Мы можем выполнить это преобразовав некую настроенную ВМ в шаблон. Для данного варианта мы можем пойти путём выбора необходимой ВМ с последующим кликом по ней правой кнопкой и выбором **Convert to template** для открытия блока диалога:

**Рисунок 6-15**



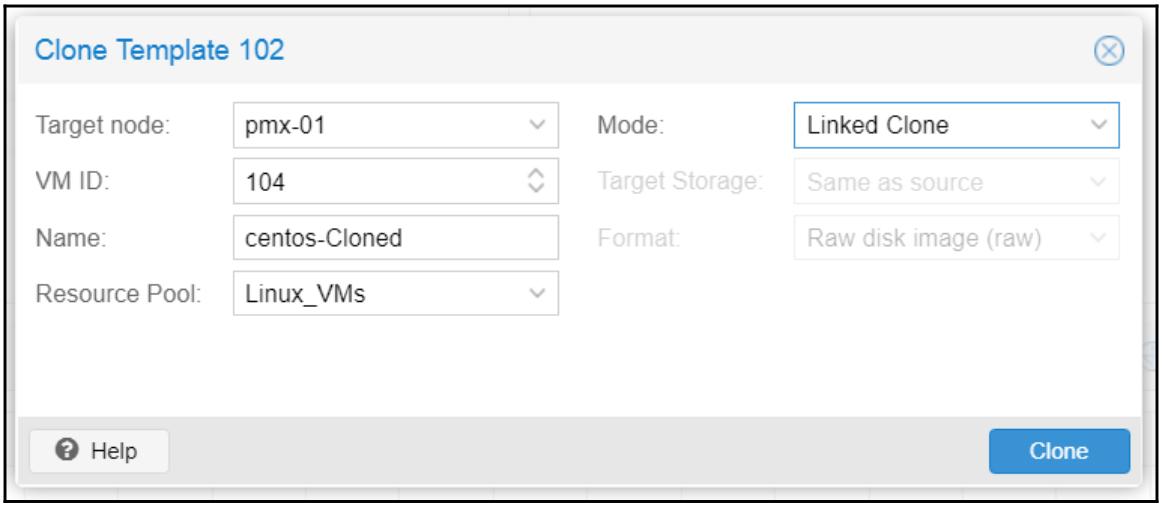
В этом примере мы собираемся преобразовать одну из имеющихся у нас ВМ, **102 (centos1)**, в некий шаблон. Кликните по **Yes** для преобразования этой ВМ в шаблон. Как уже упоминалось ранее, после того, как данная ВМ преобразована в шаблон, эта ВМ сама по себе более не может применяться. Другой требующей указания разницей является то, что её иконка в инструментальной панели Proxmox является уникальной для шаблонов KVM, что отображено в следующем экранном снимке:

**Рисунок 6-16**



С помощью данного шаблона мы можем теперь клонировать ВМ, которые будут идентичны этому шаблону. Сама процедура клонирования одна и та же, будь то ВМ или шаблон KVM. Чтобы создать некую новую ВМ или развернуть из данного шаблона множество ВМ кликните правой кнопкой по выбранному шаблону чтобы открыть его контекстное меню и затем кликните по опции **Clone**. Это откроет требующийся блок диалога клонирования, как это показано на снимке экрана внизу:

**Рисунок 6-17**



**Узел назначения**

Это ниспадающее меню для выбора того, в каком узле мы желаем создать клонированную ВМ. Это может быть тот же самый узел или любой другой узел в данном кластере с достаточными ресурсами.

**Режим**

В Proxmox 5.0 имеются два режима клонирования:

* **Full Clone** (Полное клонирование)
* **Linked Clone** (Присоединённое клонирование)

**Полное клонирование** создаёт идентичную копию данной ВМ, включая её образ виртуального диска. Это действительно изолированная ВМ, так как она никоим образом не зависит от своего шаблона или ВМ источника. Даже если мы удалим сам шаблон или ВМ источника, наша вновь созданная ВМ всё ещё будет работать без каких- то проблем. **Full Clone** потребляет столько же пространства хранения, сколько и его первоначальная ВМ, так как его виртуальный диск также клонируется. Полные клоны полезны когда выделяемые ресурсы идентичны для всех оснащаемых ВМ, однако гостевые операционные системы могут отличаться, а могут и быть теми же самыми.

**Присоединённый клон** создаёт некий дубликат первоначальной ВМ за вычетом самого первоначального образа виртуального диска. Это создаёт некий дополнительный пустой образ диска, который ссылается на свой первоначальный виртуальный диск, причём в такой присоединённый образ клонированного диска помещаются только новые данные. Все запросы на чтение, за исключением касающихся новых данных, автоматически перенаправляются в его первоначальный образ диска. **Linked Clone** существенным образом зависит от своего шаблона или ВМ источника. Такой режим клонирования полезен в том случае, когда все клонированные ВМ будут иметь в точности те же самые конфигурации оборудования и программного обеспечения, в том числе и гостевую операционную систему. **Linked Clone** потребляет намного меньше пространства хранения, так как первоначальный или базовый образ никогда не дублируется, а всего лишь применяется в качестве ссылки в получаемом новом **Присоединённом клоне** ВМ.

|  |  |
| --- | --- |
| [Совет] | **Совет** |
| Хотя мы и не можем включить имеющийся шаблон, мы всё ещё можем производить изменения его ресурсов, такие как ЦПУ или Оперативная память. Однако это не рекомендуется при практическом применении, так как любое изменение оборудования может создавать проблемы при включении какой- то клонированной ВМ. Также очень важно гарантировать, что получаемый шаблон источника никоим образом не разрушается. Любое нарушение шаблона вызывает отказ всех присоединённых клонов. |

**Расширенные параметры настройки для ВМ**

Теперь мы рассмотрим некоторые дополнительные варианты настройки, которые мы можем применять для расширения имеющихся возможностей виртуальных машин KVM.

**Настройка звуковых устройств**

В этом разделе мы собираемся рассмотреть как добавлять поддержку звуковых устройств в ВМ. По умолчанию Proxmox не добавляет аудио оборудование в ВМ. Для того чтобы операционная система данной ВМ запустила службу звуковых устройств в файл настроек этой ВМ следует добавить некоторые аргументы с применением CLI. Как и в Proxmox VE 4.1 невозможно добавлять звуковое взаимодействие с помощью GUI. приводимые ниже шаги добавляют некое звуковое устройство в ВМ:

1. Зарегистрируйтесь в узле Proxmox через SSH или напрямую в консоли.
2. Перейдите в каталог настроек Proxmox
3. **/etc/pve/nodes/<node\_name>/qemu-server/<vm\_id>.conf**

1. Откройте файл настроек ВМ предпочитаемым вами редактором и добавьте следующие аргументы:

Для ВМ Windows 10 и последующих версий:

args: -device intel-had,id=sound5,bus=pci.0,

addr=0x18 -device hda-micro,id=sound5-codec0,

bus=sound5.0,cad=0 -device had-duplex,

id=sound5-codec1,bus=sound5.0,cad=1

Для ВМ Windows XP:

args: -device AC97,addr=0x18

1. Сохраните этот файл настроек и выйдите из редактора.
2. Выполните цикл выключения- включения для активации имеющегося звукового устройства.

**Настройка проброса PCI**

В Proxmox имеется возможность проброса (passthrough) устройства PCI напрямую в некую ВМ. В данном разделе мы собираемся рассмотреть как настроить и проверитриводимые далее шаги предназначены для включению и настройки проброса устройства PCI в Proxmox:

1. Зарегистрируйтесь в узле Proxmox через SSH или напрямую в консоли.
2. Откройте настройку файла **grub** с помощью редактора:
3. **# nano /etc/default/grub**

1. Измените **GRUB\_CMDLINE\_LINUX\_DEFAULT="quiet"** на следующее:

Для ЦПУ Intel:

GRUB\_CMDLINE\_LINUX\_DEFAULT="quiet intel\_iommu=on"

Для ЦПУ AMD:

GRUB\_CMDLINE\_LINUX\_DEFAULT="quiet amd\_iommu=on"

1. Сохраните эти изменения и покиньте свой редактор.
2. Для обновления **grub** выполните следующую команду:
3. **# update-grub**

1. Только если вы используете ЦПУ AMD добавьте следующую строку в своём файле настроек **/etc/modprobe.d/kvm\_iommu\_map\_guest.conf**:
2. options kvm allow\_unsafe\_assigned\_interrupt=1

1. Убедитесь что следующие модули загружаются в **/etc/modules**:
   * **vfio\_iommu\_type1**
   * **vfio\_virqfd**
   * **vfio\_pci**
   * **vfio**
2. Выполните повторную загрузку этого узлаProxmox.
3. Определите адрес местоположения требуемого устройства PCI в виде **xx:xx.x**, воспользовавшись следующей командой:
4. **# lspci**

1. Введите в свой файл настроек ВМ следующую строку с соответствующим идентификатором устройства PCI:
2. machine: q35
3. hostpsi0: 01:00.0,pcie=1

1. Выполните цикл выключения- включения данной ВМ.
2. Установите требующиеся драйверы для данного устройства PCI в соответствующей операционной системе ВМ.

**Настройка проброса GPU**

В данном разделе мы собираемся рассмотреть как настроить некий видео адаптер на его прямое применение в ВМ. Тот GPU, который подлежит добавлению в определённую ВМ не должен привязываться к своему узлу хоста. Чтобы убедиться, что данное устройство не используется данным хостом осуществите следующие шаги:

1. Мы должны определить имя производителя и идентификатор устройства, того устройства GPU, которое предполагается пробрасывать. Чтобы прихватить это устройство мы можем выполнить команду **#lspci**. Данное устройство должно для нас отображаться как некий VGA совместимый контроллер. Приводимый далее снимок экрана показывает нам устройство VGA с идентификатором **00:02.0**:

**Рисунок 6-18**



1. Чтобы определить идентификаторы данного устройства и его производителя выполните данную команду снова воспользовавшись следующим форматом:
2. **#lspci -n -s 00:02**

1. Данная команда воспроизведёт ряд чисел. Идентификаторы устройства и производителя являются самыми последними двумя наборами чисел. Ниже приводится соответствующий набор числовых значений для нашего примера узла:
2. **# df00:02.0 0300: 1013:00b8**

1. Примите во внимание этот идентификатор устройства, затем создайте некий файл **/etc/modprobe.d/vfio.conf** для определения явным образом того, чтобы данное GPU было пробрасываемым устройством **vfio** и предотвратить конфликт с назначаемым по умолчанию устройства VGA. Введите следующую строку в свой файл **vfio.conf**:
2. options vfio-pci ids=1013:00b8 disable\_vga

1. Теперь нам следует занести в чёрный список соответствующие устройства VGA по усолчанию с тем, чтобы они не загружались в процессе начального запуска следующим образом:
2. **# echo "blacklist nvidia" >> /etc/modprobe.d/blacklist.conf**
3. **# echo "blacklist radeon" >> /etc/modprobe.d/blacklist.conf**
4. **# echo "blacklist nouveau" >> /etc/modprobe.d/blacklist.conf**

|  |  |
| --- | --- |
| [Совет] | **Совет** |
| При попытке добавления проброса GPU для некоторой ВМ, важно иметь в виду, что не все устройства GPU имеют возможность проброса. пробуйте различные конфигурации. |

На текущий момент мы теперь готовы настроить саму свою ВМ для применения проброса GPU. Рекомендуемым способом настройки является проброс PCI **Open Virtual Machine Firmware** (**OVMF**, Открытого встроенного ПО ВМ). OVMF является проектом для разрешения ВМ применения BIOS **UEFI** (**Unified Extensible Firmware Interface**). Для поддержки свойств OVMF используемая гостевая операционная система обязана поддерживать UEFI. Последующие шагни помогут нам определить совместимость данного устройства GPU с UEFI:

1. Зарегистрируйтесь в своём узле с устройством GPU воспользовавшись SSH.
2. Выполните следующие команды для выгрузки и компиляции некоего инструмента, который в свою очередь выгрузит содержимое ROM устройства вашего GPU:
3. **# git clone https://github.com/awilliam/rom-parser**
4. **# cd rom-parser**
5. **# make**

1. Исполните следующий набор команд для выгрузки содержимого ROM вашего устройства GPU во временный каталог:
2. **# cd /sys/bus/pci/devices/0000:01:00.0/**
3. **# echo 1 > rom**
4. **# cat rom > /home/rom-parser/image.rom**
5. **# echo 0 > rom**

1. Выполните приводимую далее команду чтобы проверить свой выгруженный ROM с помощью синтаксического анализатора ROM, который мы загрузили ранее и выполнить компиляцию на предмет определения того, является ли наше GPU UEFI совместимым:
2. **# cd /home/rom-parser**
3. **# ./rom-parser /tmp/image.rom**

1. Эта команда отобразит информацию, похожую на следующее:
2. Valid ROM signature found @0h, PCIR offset 60h
3. PCIR: type 3, vendor 102b, device: 0532, class: 030000
4. PCIR: revision 0, vendor revision: 2139
5. EFI: Signature Valid
6. Last image

Если PCIR имеет **type 3**, тогда наше устройство GPU является UEFI/ OVMF совместимым.

Соответствующая настройка ВМ должна выглядеть подобно приводимой ниже после выбора BIOS OVMF и добавить необходимую строку **hostpci**. Осуществите необходимые изменения и затем выполните цикл отключения- включения чтобы активировать эти новые настройки.

bios: ovmf

scsihw: virtio-scsi-pci

machine: q35

hostpci0: 02:00,pcie=1,x-vga=on

....................

....................

При использовании GPU устройств nVidia, таких как GeForce Experience, могут возникать ситуации, приводящие к краху виртуальной машины. В подобных случаях добавьте в **/etc/modprobe.d/kvm.conf** приводимую ниже строку. Данная проблема может возникать при применении такого программного обеспечения как PassMark PerformanceTest и SiSoftware Sandra:

options kvm ignore\_msrs=1

**Подготовка к подключению во время работы**

В данном разделе мы намереваемся рассмотреть как настраивать опцию горячего подключения в виртуальных машинах Proxmox. применяя функциональность подключения в горячем режиме мы можем добавлять или удалять устройства или ресурсы в некоторой ВМ без выполнения в ней цикла отключения- включения. Для Proxmox 5.0 мы можем применять варианты подключения в горячем режиме для следующих устройств:

* Дисков
* Сетевых интерфейсов
* ЦПУ
* Оперативной памяти
* USB

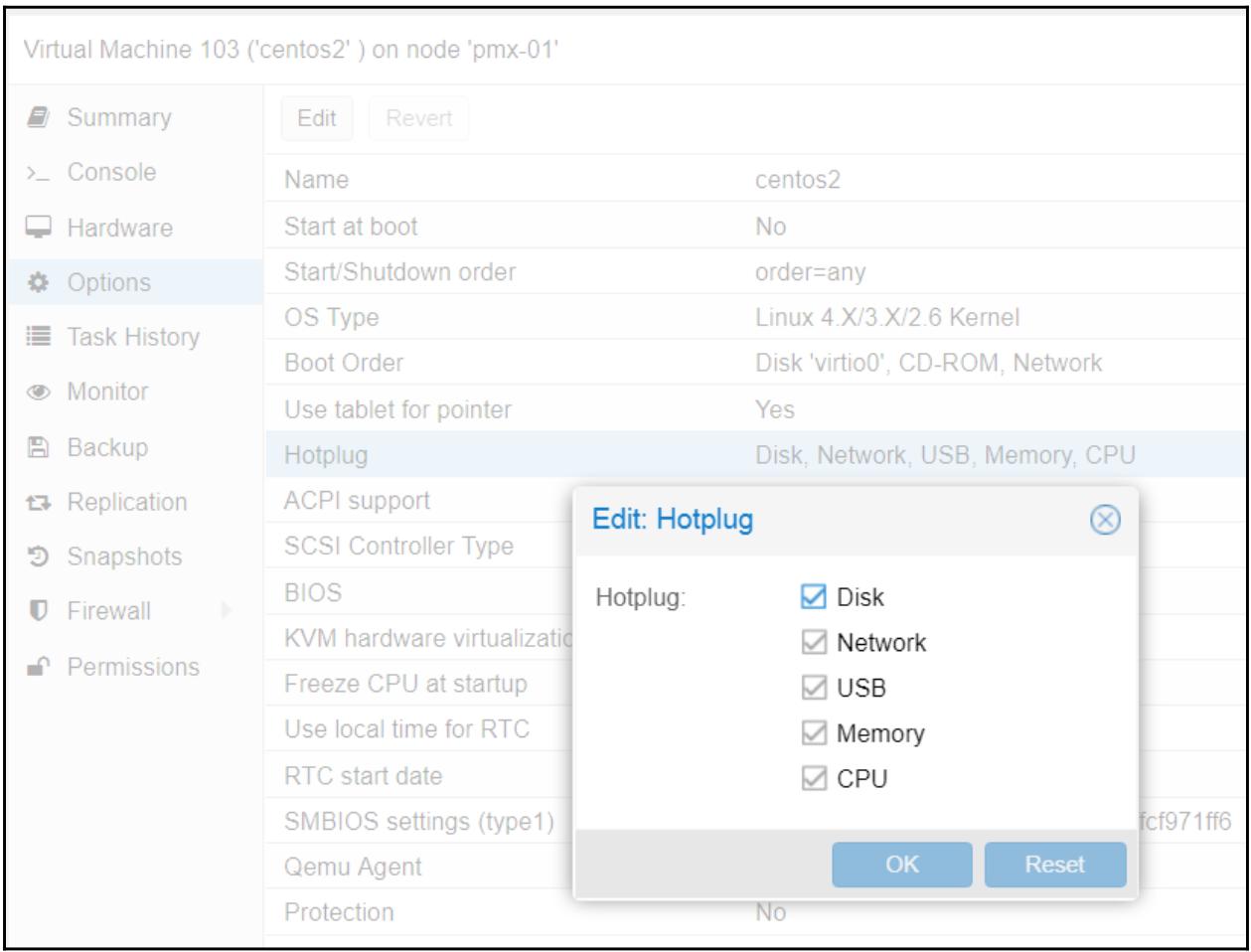
В версии Proxmox 5.0 мы можем увеличивать ЦПУ и Оперативную память, но не можем уменьшать их. относительно дисков и сетевых интерфейсов мы можем одинаково их и подключать в горячем режиме и отключать по- горячему. Следующая таблица показывает какие типы устройств поддерживаются в различных операционных системах:

[Таблице 1-1](file:///C:\\Users\\Adm\\Documents\\PVE\\Ch01.html" \l "table0101" \t "_top)

| **Таблица 6-1. Права доступа** | | | |
| --- | --- | --- | --- |
| **Устройство** | **Ядро** | **Подключение/ отключение в горячем режиме** | **ОС** |
| Диск | Все | Оба | Все версии Linux/ Windows |
| NIC | Все | Оба | Все версии Linux/ Windows |
| ЦПУ | Выше 3.1 | Для Windows только подключение,  в Linux и то и другое | Все версии Linux,  Windows Server 2008 и старше |
| Оперативная память | Выше 3.1 | Для Windows только подключение,  в Linux и то и другое | Все версии Linux,  Windows Server 2008 и старше |

Хотя все основные настройки для включения подключения в горячем режиме для Proxmox следует проводить в CLI, мы можем разрешать и запрещать подключаемые в горячем реиме устройства через меню закладок **Datacenter | Node | VM | Options**, как это отображено на следующем снимке экрана:

**Рисунок 6-19**



Прежде чем горячее подключение будет применено, нам необходимо подготовить ВМ на основе Linux. Для подключения в горячем режиме в гостевой операционной системе Linux необходимо выполнить загрузку двух модулей. Мы можем загрузить эти модули воспользовавшись следующими командами:

**# modprobe acpiphp**

**# modprobe pci\_hotplug**

Для их автоматической загрузки в процессе начального запуска мы можем добавить их в **/etc/modules**.

В тех гостевых ОС Linux, которые основываются на Ядре с версией ниже 4.7 нам потребуется создать новый файл правил **udev** в файле **/lib/udev/rules.d/80-hotplug-cpu-mem.rules** и добавить в него следующие строки:

SUBSYSTEM=="cpu",ACTION=="add",TEST=="online",ATTR{online}=="0",

ATTR=={online}="1"

SUBSYSTEM=="memory",ACTION=="add",TEST=="state",

ATTR{state}=="offline",ATTR=={state}="online"

Для госетвых операционных систем основанных на Ядре 4.7 или более новых, нам не нужно добавлять такие правила **udev**, для подключения оперативной памяти в горячем режиме,однако нам всё ещё потребуются параметры для ЦПУ. Нам понадобится добавить следующий параметр ядра в процессе начального запуска:

memhp\_default\_state=online

Для разрешения подключения памяти в горячем режиме нам следует выполнить следующие шаги для добавления параметров ядра начальной загрузки:

1. В SSH гостевой ОС Linux выполните следующую команду:
2. **#gksudo gedit /etc/default/grub**

1. Определите местоположение строки, начинающейся с **GRUB\_CMDLINE\_LINUX\_DEFAULT** и наберите необходимый параметр ядра в самом конце этой строки. Данная строка теперь должна выглядеть следующим образом:
2. GRUB\_CMDLINE\_LINUX\_DEFAULT="quiet splash memhp\_default\_state=online"

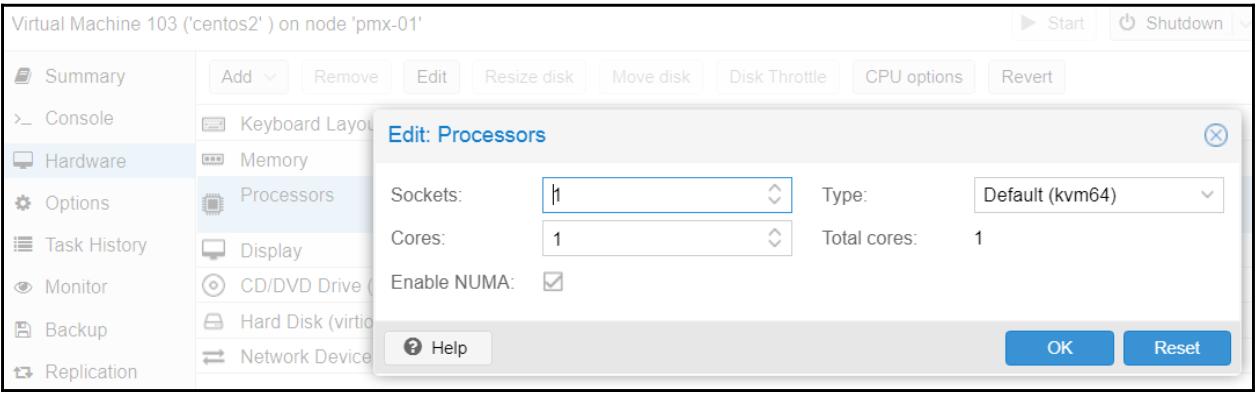
1. Сохраните этот файл и покиньте свой редактор.
2. Для обновления начального загрузчика **grub** выполните следующую команду:
3. **# sudo update-grub**

1. Для активации всех модулей, правил и параметра ядра Proxmox потребуется цикл отключения- включения.
2. После перезагрузки выполните следующую команду чтобы проверить какие параметры ядра загружены при начальном запуске.
3. **# cat /proc/cmdline**

**Настройка подключения во время работы**

Для подключения в горячем режиме ЦПУ и Оперативной памяти нам также необходимо проверить, что для данной ВМ включена опция NUMA. Опцию NUMA можно отыскать в меню **Datacenter | Node | VM | Hardware | Processors**. Для открытия блока диалога ЦПУ кликните **Edit**:

**Рисунок 6-20**



Для настройки горячего подключения диска или сетевого интерфейса не требуются никакие дополнительные настройки.

**Подключение vCPU в горячем режиме**

Для добавления виртуального ЦПУ или vCPU проследуйте в меню **Datacenter | Node | VM | Hardware**, затем выберите **Processors** и кликните по **Edit** для открытия необходимого блока диалога. просто наберите требуемое число ядер или воспользуйтесь стрелками опций вверх и вниз в данном тексте для выбора требующегося числа ядер или vCPU. Кликните по **OK** для принятия изменений. В этом блоке диалога мы также можем добавить новый ЦПУ. Также мы можем добавлять vCPU исполняя в CLI своего узла Proxmox следующую команду:

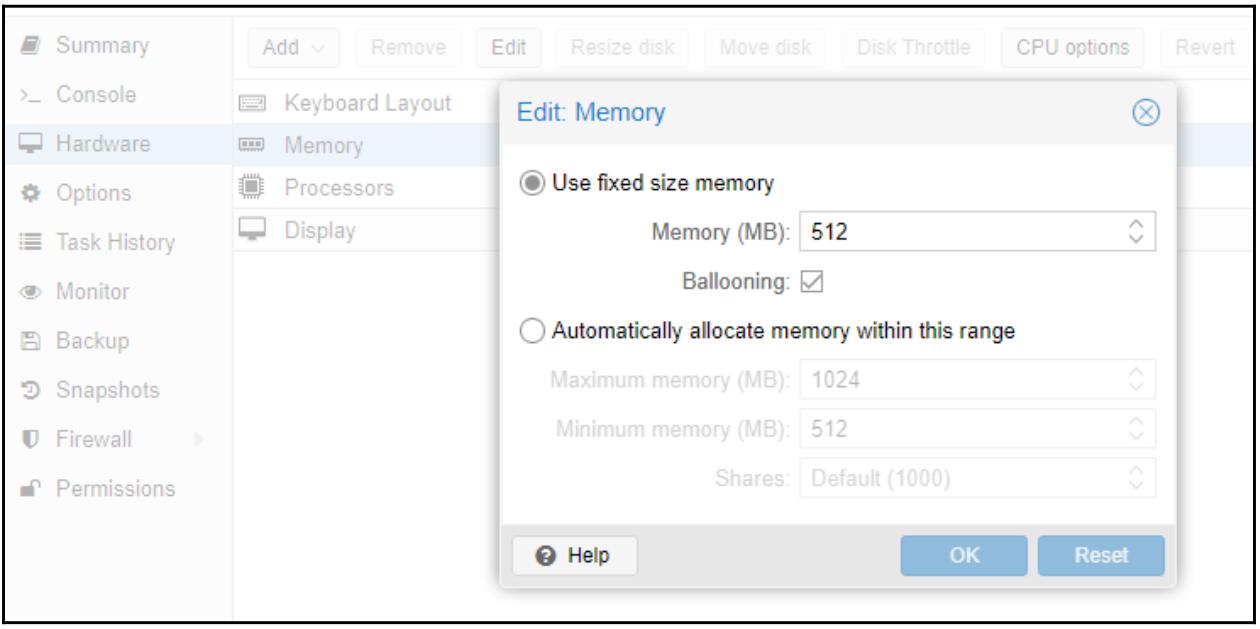
**# qm set <vm\_id> -vcpus 2**

Поскольку в нашем примере ВМ уже имеется один ЦПУ, наша предыдущая команда добавит дополнительный ЦПУ, доведя общее число ЦПУ в этой ВМ до двух.

**Подключение оперативной памяти в горячем режиме**

Чтобы открыть соответствующий блок диалога для изменения выделяемой оперативной памяти проследуйте в меню **Datacenter | Node | VM | Hardware**. Выберите **Memory** и затем кликните по **Edit** для открытия необходимого блока диалога памяти.

**Рисунок 6-21**

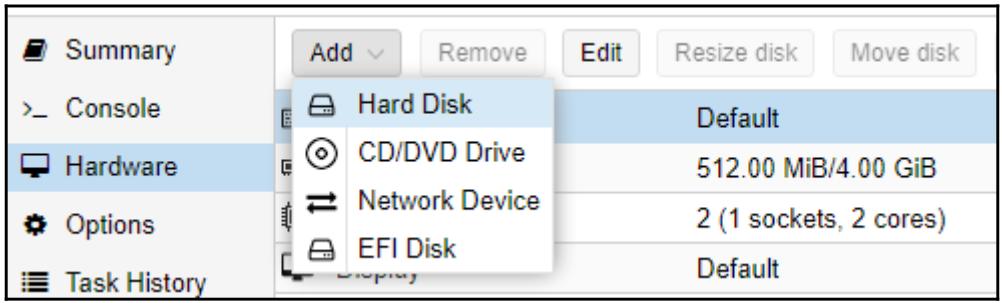


Измените подлежащий выделению объём памяти, а затем кликните **OK** для принятия этих изменений. Проверьте что опция NUMA разрешена в вашем блоке диалога **Processors**, что уже обсуждалось в данном разделе.

**Подключение дисков/ vNIC в горячем режиме**

Для подключения в горячем режиме нового диска или сетевого интерфейса проследуйте в меню **Datacenter | Node | VM | Hardware** и затем выберите некий элемент из ниспадающего меню **Add**. Получаемые блоки диалога для добавления этих ресурсов аналогичны тем блокам диалога процесса создания ВМ, которые мы уже наблюдали в более ранних разделах данной главы. На следующем снимке экрана показано ниспадающее меню **Add**:

**Рисунок 6-22**



|  |  |
| --- | --- |
| [Совет] | **Совет** |
| Хотя подключение в горячем режиме ЦПУ и Оперативной памяти работают и в Linux, и в Windows, убедитесь в этом проведя надлежащее тестирование прежде чем реализуете их в своей промышленной среде. Добавление в горячем режиме ЦПУ/ Оперативной памяти может вызывать некие нестабильные ситуации для вашей ВМ, вызывая её замораживания и требуя полной перезагрузки.. |

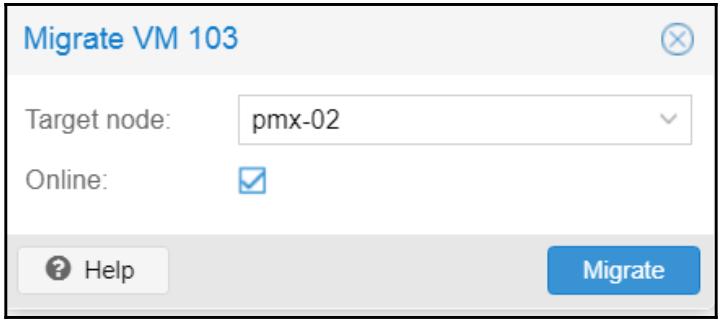
**Миграция виртуальных машин KVM**

Миграция Proxmox делает возможным перемещение виртуальных машин KVM в некий узел Proxmox как в отключённом режиме, так и во включённом состоянии (по- живому). Наиболее распространённым сценарием миграции ВМ является тот случай, когда некоторому узлу Proxmox требуется перезагрузка в результате основополагающего обновления ядра или прочих исправлений. Прочие ситуации могут включать в себя отказы оборудования, замену узла, проблемы с программным обеспечением и тому подобное. Без наличия соответствующего варианта миграции в реальном времени любая перезагрузка была бы очень сложной для администратора, так как все работающие ВМ следовало бы отключить прежде чем произвести такой процесс перезагрузки. Это может вызывать существенное время простоя в критически важных виртуальных средах.

При наличии варианта с миграцией в реальном времени некая работающая ВМ может быть перемещена на другой узел без времени простоя. В процессе миграции в реальном времени данная ВМ не претерпевает никаких основных остановов. После перезагрузки её основного узла просто выполните миграцию этой ВМ обратно на него. Любые отключённые ВМ также запросто могут перемещаться.

Proxmox придерживается очень минималистского подхода к своему процессу миграции. Для доступа к необходимому процессу миграции кликните правой кнопкой по той ВМ, которая подлежит миграции, для открытия её контекстного меню и затем выберите **Migration** или кликните по кнопке **Migratioe** в правом верхнем углу для открытия требуемого блока диалога. Приводимый ниже снимок экрана отображает такой блок диалога миграции:

**Рисунок 6-23**



В этом блоке диалога просто выберите необходимый узел назначения, а после этого, в зависимости от того выполяется ли миграция в отключённом режиме или в реальном времени, кликните на соответствующий флаговый блок. Затем нажмите имеющуюся клавишу **Migrate** для того, чтобы начать процесс миграции. В зависимости от размера виртуального диска выделенной данной ВМ оперативной памяти время процесса миграции может быть различным. Миграция в реальном времени/ во включённом состоянии, помимо всего прочего, также перемещает всё содержимое оперативной памяти данной ВМ. Чем больше выделенная оперативная память, тем длиннее будет процесс миграции. В своём предыдущем примере мы осуществили миграцию в режиме реального времени миграцию ВМ с идентификатором **#103** на узел **pmx-02**.

**Выводы**

В данной главе мы рассмотрели виртуальные машины KVM и то, как создавать, клонировать и мигрировать их при необходимости. Мы рассмотрели также некий расширенные настройки, такие как добавление звукового оборудования и включение проброса PCI/ GPU для ВМ RVM. Вооружаясь такой мощностью клонирования мы можем без усилий масштабировать виртуальный кластер при оснащении идентичных виртуальных машин. Также были объяснены опциональные и не используемые в промышленной реализации установки встраиваемых виртуальных сред.

Наилучшей практикой использования виртуальных машин KVM являются случаи применения всех операционных систем, отличающихся от Linux, а также когда обязательна общая изоляция всех ресурсов между Вм.

В своей следующей главе мы собираемся рассмотреть более подробно контейнеры LXC. Мы изучим почему администраторам Proxmox следует выбирать их вместо виртуальных машин KVM.

[Вопросы? Обсудите их на mdl.ru](javascript:tocall())  
[Нашли ошибку? Сообщите о ней на данной странице](https://bugs.launchpad.net/openstack-manuals/+filebug?field.title=MasteringProxmox3ed&field.comment=%0A%0A-----------------------------------%0ABuilt:%202018-01-06T17:43:47+00:00%0Agit%20SHA:%2013fb30b7c263d715eee84f0ca1a16df79697eb88%0AURL:%20file:///C:/Users/Adm/Documents/PVE/Ch06.html%0Asource%20File:%20file:a35-prades%0Axml:id:%20Mastering-Proxmox-3-ed)

Дополнительные ссылки:

* [Атакующий код запуска оболочки с нуля](http://onreader.mdl.ru/OffensiveShellcodeScratch/content/index.html), Ришалин Пиллэй, май 2022
* [Linux подсистема Windows (WSL) для профессионалов](http://onreader.mdl.ru/ProWindowsSubsystemLinux/content/index.html), Хайден Барнс, Apress, июнь 2021
* [Внутреннее устройство CPython](http://onreader.mdl.ru/CPythonInternals/content/Ch10.html), Энтони Шоу, Real Python, январь 2021
* [Контейнеры Linux и Виртуализация: с точки зрения ядра](http://onreader.mdl.ru/LinuxContainersVirtualizationKernelPerspective/content/Ch06.html), Шашанк Мохан Джейн, Apress, октябрь 2020
* [Изучаем подсистемы Windows для Linux](http://onreader.mdl.ru/LearnWindowsSubsystemLinuxProfessionals/content/Ch05.html), Прэйтик Сингх, Apress, сентябрь 2020
* [Всё что требуется для RabbitMQ](http://onreader.mdl.ru/RabbitMQEssentialsDistributedScalableApplications.2nd/content/index.html), 2е изд., Ловайса Йохансон, Дэйвид Доссо, Packt Publishing, август 2020
* [Практика загрузки. Изучение процесса загрузки Linux, Windows и Unix](http://onreader.mdl.ru/HandsonBooting/content/index.html), Йогеш Бабар, Apress, июль 2020
* [Распределённые системы для практиков](http://onreader.mdl.ru/DistributedSystemsForPractitioners/content/Ch07.html), Даймос Раптис, Learnpub, май 2020
* [Практическая автоматизация предприятия в Linux](http://onreader.mdl.ru/HandsOnEnterpriseAutomationOnLinux/content/index.html), Джеймс Фриман, Packt Publishing, январь 2020, Действенное выполнение крупномасштабной автоматизации инфраструктуры Linux с применением Ansible.
* [Внутреннее устройство баз данных](http://onreader.mdl.ru/DatabaseInternals/content/index.html), Алекс Петров, O`Reilly Media, Inc., октябрь 2019
* [Книга рецептов параллельного программирования Python](http://onreader.mdl.ru/PythonParallelProgrammingCookbook.2nd/content/index.html). 2е изд., Джанкарло Закконе, Packt Publishing, сентябрь 2019
* [Полное руководство Ansible](http://onreader.mdl.ru/MasteringAnsible.3ed/content/index.html). 3е изд., Джеймс Фриман и Джесс Китинг, Packt Publishing, март 2019
* [Книга рецептов NGINX](http://onreader.mdl.ru/NGINXCookbook/content/index.html) Дерек ДеДжонге, O’Reilly Media, Inc, ноябрь 2018
* [Полное руководство Ceph, 2е изд.](http://onreader.mdl.ru/MasteringCeph.2ed/content/index.html) Ник Фиск, Packt Publishing, февраль 2019
* [Docker для разработчиков Rails](http://onreader.mdl.ru/DockerRailsDevelopersApplicationsEverywhere/content/index.html) Роб Айзенберг, The Pragmatic Programmers, LLC., февраль 2019, с дополнениями по настройкам Django и 100Gb IB
* [Глава 11. SQL Server и контейнеры](http://onreader.mdl.ru/ProSQLServerOnLinux/content/Ch11.html) (включая Kubernetes) Боб Вордс, "Профессиональный SQL Server поверх Linux", Apress, октябрь 2018
* [Полное руководство параллельного программирования на Python](http://onreader.mdl.ru/MasteringConcurrencyInPython/content/index.html) Куан Нгуен, Packt Publishing, ноябрь 2018
* [Asyncio в Python 3](http://onreader.mdl.ru/UsingAsyncioPython3/content/index.html) Цалеб Хаттингх, O’Reilly Media, Inc, март 2018
* [RabbitMQ для профессионалов](http://onreader.mdl.ru/RabbitMQInDepth/content/index.html) Гайвин Рой, Manning Publications, сентябрь 2017
* [Proxmox. Полное руководство. 3е изд](file:///C:\Users\Adm\Documents\PVE\Ch06.html) Васим Ахмед, Packt Publishing, ноябрь 2017
* [Книга рецептов Ceph, 2е изд](http://onreader.mdl.ru/CephCookbook.2ed/content/index.html) Викхайят Умрао,Мишель Хаккет,Каран Сингх, Packt Publishing, ноябрь 2017
* [Изучаем Ceph](http://onreader.mdl.ru/LearningCeph2ed/content/index.html), 2е изд., Энтони Д`Атри, Вайбхав Бхембре, Каран Сингх, Packt Publishing, октябрь 2017
* [Книга рецептов виртуализации KVM](http://onreader.mdl.ru/KVMVirtualizationCookbook/content/Ch08.html) Константин Иванов, Packt Publishing, июнь 2017
* [Полное руководство работы с сетями на Python](http://onreader.mdl.ru/MasteringPythonNetworking/content/index.html). Эрик Чоу, Июнь 2017
* [Контейнеризация при помощи LXC](http://onreader.mdl.ru/ContainerizationWithLXC/content/index.html) Константин Иванов, Packt Publishing, март 2017
* [Proxmox. Полное руководство](http://onreader.mdl.ru/MasteringProxmox.2ed/content/index.html). 2е изд., Васим Ахмед, Packt Publishing, май 2016
* [Книга рецептов Ceph](http://onreader.mdl.ru/CephCookbook/content/index.html) Каран Сингх, Packt Publishing, февраль 2016
* [Полная виртуализация. Базовая коммерческая редакция](http://onreader.mdl.ru/VirtualizationComplete/content/index.html): Proxmox-freeNAS-Zentyal-pfSense. Ли Р. Сюрбер, февраль 2016
* [Zabbix. Полное руководство](http://onreader.mdl.ru/MasteringZabbix.2ed/content/Ch04.html). 2е изд., Андреа Далле Ваккье, сентябрь 2015
* [Книга рецептов Proxmox. Главы 1-6, Дополнения: Преобразование OpenVZ в LXC, Организация ограждения](http://onreader.mdl.ru/ProxmoxCookbook/content/index.html) Васим Ахмед, Packt Publishing, август 2015
* [Изучаем Ceph](http://onreader.mdl.ru/LearningCeph/content/index.html) Каран Сингх, Packt Publishing, январь 2015