Kernel-based Virtual Machine

Установка Windows 2003 x86-64

Virtio-HDD

Драйверы виртуальных устройств (virtio) значительно повышают производительность подсистем виртуальной машины, но требуют отдельной установки. Последние версии можно скачать тут, на 23.11.2013 последняя версия 0.1-74: раз или два. upd. новый адрес virtio драйверов, FedoraProject.

Если в системе не будет virtio-драйвера контроллёра жёстких дисков, то система либо не стартует, показывая синий экран, либо не видит жёсткий диск на этапе установки.

Соответсвенно, решить этот вопрос можно:

1. либо на этапе установки Windows;
2. либо после установки системы

В первом случае, необходимо на самом первом этапе установки подключить образ дискеты раз или два с virtio-драйвером жёсткого диска (viostor) (для этого нажать F6 и следовать инструкциям).

Во втором случае, необходимо подключить на рабочей машине временный virtio-диск, установить на него драйвера, перезагрузить компьютер, поменять тип системного диска на virtio. После этого система должна благополучно загрузиться на новых драйверах.

Ни смотря ни на какие шаманства, второй способо у меня не заработал, поэтому стараюсь установить virtio на этапу установки windows

Baloon drivers

Baloon drivers реализует динамическое выделение памяти для виртуальных машин т.е., излишки ОЗУ виртуальной машины передаются в распоряжение хостовой системы, до востребования.

Установка: Подключить образ компакт-диска с virtio-драйверами к виртуальной машине. Далее, Windows (Guest OS) → «Диспетчер устройств» → Другие устройства → PCI Device → Обновить драйвер. Выбрать каталог с драйвером **CD-ROM:\WNET\AMD64**.

После установки в разделе «Системные устройства» диспетчера устройств появится «VirtIO Ballon Driver».

Ethernet-контроллер

Использование VirtIO-сетевого адаптера значительно повышает скорость обмена данными между хостом и гостевой системой.

Установка: Подключить образ компакт-диска с virtio-драйверами к виртуальной машине. Далее, Windows (Guest OS) → «Диспетчер устройств» → Другие устройства → Ethernet-контроллер → Обновить драйвер. Выбрать каталог с драйвером **CD-ROM:\XP\AMD64**.

После установки в разделе «Сетевые платы» диспетчера устройств появится «Red Hat VirtIO Ethernet Adapter».

Display Driver

VMware SVGA II

Утверждается, что эмуляция «cirrus» видеокарты, на 20 % медленее, чем «std». В свою очередь «std» в 5 раз медленее, чем эмуляция «vmware». Очевидно, что если нет иных причин, то надо устанавливать эмуляцию дисплея «vmware».

Установка:

1. скачать vmware display driver: x86,x86-64 или x86, x86-64;
2. выключить виртуальную машину, с помощью virt-manager'а подключить соответствующий образ компакт-диска с драйверами, установиить vmvga видеоподсистему;
3. загрузить ОС, установить драйверы дисплея.

После установки в разделе «Видеоадаптеры» диспетчера устройств появится «VMware SVGA II». Субъективно, графика работает шустрее. Однако доступные режимы экрана предполагают наличие монитора с соотношением сторон 4×3. Здесь описаны действия по добавлению пользовательского разрешения в Windows, но у меня не взлетело.

Spice qxl

Второй способ ускорить графику - использовать Qxl для эмуляции видео. Для этого надо:

1. скачать для поддержки Qxl Windows guest tools и/или драйвера: тут или тут, поместить в доступный из гостевой ОС каталог;
2. выключить виртуальную машину, включить через virt-manager режим дисплея qxl, там же установить параметры Spice, а именно, порт и ip-адрес для удалённого подключения, включить машину, установить драйвера. В debian'е столкнулся с тем, что Spice не стартует после настройки в virt-manager'е т.к., последний записывает в конфигурацию порт для защищённого режима (tls), а в настройках qemu по-умолчанию tls выключено. Решение - отредактировать через *virsh edit <имя_ВМ>* настройки ВМ, точнее, удалить запись tlsPort=«xxxx».

Скорость работы удивляет, разница с реальным железом заметна только на видео, которое сжимается с потерями, а потому немного портится.

Просмотр с помощью Spicy, который необходимо подключать к указанному в настройках Spice порту на ip хоста виртуальной машины. Выход из полноэкранного режима Spicy - L_Shifs + F12.

Работа со snapshot'ами

Если верить интернету, то virt-manager в августе 2013-ого года обзавёлся поддержкой internal-снапшотов, но до debian'а эта долгожданная новинка ещё не добралась. Ждём с нетерпением, а пока используем для этих целей консольную утилиту virsh:

Internal Snapshots

Внутренний снимок: снимок у которого сохранённое состояние (saved state) и его последующие изменения (delta), физически хранятся в одном QCOW2-файле. Внутренние снимки удобны тем, что обеспечивают доступность к предыдущим состояниям виртуальной машины, даже в случае её переноса на другую платформу.

Внутренние снимки можно делать, как при работающей виртуальной машине (online), так и при выключенной виртуальной машине (offline). В первом случае в снимок виртуальной машины запишется и её текущее состояние: ram + cpu. Сохранить текущее состояние, без создания снимка можно, либо через virt-manager, либо через virsh:

** Текущее состояние ВМ **

```
# сохранение состояние
virsh # save vmname foo_state_save
# восстановление состояния
virsh # restore foo_state_save
```

Просмотр всех виртуальных машин:

```
virsh # list --all
```

Создать снимок:

```
virsh # snapshot-create-as vmname snapshotname
```

Просмотреть снимки:

```
virsh # snapshot-list vmname
```

Восстановить из снимка:

```
virsh # snapshot-revert vmname snapshotname
```

Удалить снимок:

```
virsh # snapshot-delete vmname snapshotname
```

Внутренние снапшоты в настоящее время использовать не рекомендуется (источник раз и два).

External Snapshots

Внешние снимки: здесь оригинальный qcow2 файл переключается в режим «только чтение» - это будет сохранённое состояние (save state), а все дальнейшие изменения (delta) пишутся в новый qcow2-файл. Внешние снимки используются для создания бекапов т.к., точка сохранения будучи в режиме только чтение, может быть прочитана параллельно с работой виртуальной машины.

Подробнее

Backing files как снапшоты

При включенной виртуальной машине на базе существующего диска создаётся новый диск, который использует существующий в качестве неизменяемого хранилища (backing store). Все изменения будут вноситься в созданный файл

```
qemu-img create -F qcow2 -f qcow2 -b vm.backing_store.qcow2 vm.work_store.qcow2
```

Для удобства использования можно делать символическую ссылку vm.store.qcow2 на vm.work_store.qcow2 и уже через неё подключать хранилище к виртуальной машине.

Для записи изменений из текущего файла данных в родительский служит команда

```
qemu-img commit vm.work_store.qcow2
```

Откат изменений происходит через

- создание нового рабочего хранилища на базе backing store;
- возврат к использованию backing store в качестве основного хранилища ВМ.

Источник http://dustymabe.com/2015/01/11/qemu-img-backing-files-a-poor-mans-snapshotrollback/

Миграция с VBox на KVM

Миграция с VirtualBox на KVM с сохранением снапшотов. Алгоритм следующий.

1. Определить иерархию снапшотов в vbox'e, анализируя файл описания виртуальной машины *.vbox; Один из снапшотов будет идентичен коревому файлу ВМ, который находится вне каталога Snapshots. Остальная иерархия определяется по названиям снапшотов.

1. Сконвертировать в vdi файлы в raw формат. Увеличение размера в 5 раз не должно смущать.

   ```
   VBoxManage clonehd \{331640c6-4610-4490-b619-d300f3965c9e\}.vdi
   \{331640c6\}.raw -format raw
   ```

1. Сконвертировать полученные raw файлы в qcow2 формат;

   ```
   qemu-img convert -f raw \{331640c6\}.raw -O qcow2 \{331640c6\}.qcow2
   ```

1. Установить связи между qcow2 файлами (по смыслу понятно, что ниже приведены файлы корневой фс и первого снапшота) через:

   ```
   qemu-img convert -O qcow2 -o backing_file="Root.snap.qcow2" Sn01.temp.qcow2
   Sn01.snap.qcow2
   ```

1. Чтобы Sn01.snap.qcow2 не испортить при включении ВМ (данный файл может использоваться в дальнейшей цепочке снапшотов), можно создать от него ответвление:

   ```
   qemu-img create -f qcow2 -b Sn01.snap.qcow2 Sn01.work.qcow2
   ```

Примерно так. Вопрос поднимался на ЛОР'е.
P.S. по неизвестным причинам включение сжатия qcow2 файлов ключом «-c» при конвертации приводит к ошибке загрузки Windows.

Увеличение размера диска и ФС гостевой виртуальной машины

Список пулов хранения

```
virsh # pool-list --all
Name                 State      Autostart
-----------------------------------------
default              active     yes
kvm_storage          active     yes
```

Список образов в пуле

```
virsh # vol-list kvm_storage
Name                 Path
-----------------------------------------
disk.storage      /storage/kvm/disk.storage
```

Увеличение размера образа

```
virsh # vol-resize disk.storage 12G --pool kvm_storage
Size of volume 'disk.storage' successfully changed to 12G
```

Увеличение размер раздела и его файловой системы

Изменения размера раздела производятся в отмонтированном состоянии, следовательно, если расширяется системный раздел, то без загрузки с Live-CD (например, SystemRescueCd) не обойтись. При этом завершить все операции над диском можно с помощью графической утилиты Gparted.

Уменьшение размера диска ВМ в хост системе

Подробности о команде virt-sparsify из пакета libguestfs-tools по ссылке

```
virt-sparsify qcow_original qcow_less_size
```

Возможно, будет необходимо указать путь к временному каталогу, имеющему достаточно свободного места, - команда **–tmp <path-to-tmp>**

Virsh

- **Проблема**. Список виртуальных машин пуст:

  ```
  virsh list --all
  ```

- **Решение**. Указать использование системного подключения:

```
virsh --connect qemu:///system list --all
```

- Для использования системного подключения по умолчанию необходимо установить переменную окружения, например, в ~/.bashrc

```
export LIBVIRT_DEFAULT_URI="qemu:///system"
```

Virt-manager: disk+network i/o disabled

Проблема. Не отображаются графики использования сети и дискового ввода/вывода.
Решение. Данные опции по умолчанию выключены т.к., существуют проблемы с масштабированием графиков при значительном количестве виртуальных машин. Для включения в Менеджере виртуальных машин выбрать Правка→Параметры и отметить требуемые индикаторы.

Qcow2 и базовый образ

Qcow2 файл позволяет хранить только изменения относительно другого файла, представляющего базовый образ (backing store). Используя данную особенность, можно запустить несколько виртуальных машин на основе одного образа. После подготовки базового образа к нему подключается параллельно несколько qcow2 файлов и на их основе запускаются vm, каждая из которых «идет своей дорогой».

```
qemu-img create -b /storage/kvm/Base.Storage.00.qcow2 -f qcow2 Storage.01.qcow
```

Mount qcow2

```
# Загружаем модуль ядра Network Block Device
modprobe nbd

# "Подключаем" образ к устройству /dev/nbd0.
qemu-nbd --connect /dev/nbd0 --read-only /путь/к/образу.qcow2

# Ищем разделы на устройстве.
kpartx -arvs /dev/nbd0

# Здесь можно делать с разделами /dev/mapper/ndb0p* что угодно:
# монтировать, форматировать и т.п.

# Убираем девайсы разделов.
kpartx -dvs /dev/nbd0

# Выключаем qemu-nbd.
qemu-nbd --disconnect /dev/nbd0
```

P.S. Аналогичным образом можно монтировать qcow2 к работающей VM и уже в ней производить требуемые манипуляции.

Перемещение VM внутри ФС

Перед непосредственным перемещением VM необходимо отредактировать расположение в ФС. Настройки всех ВМ лежат в каталоге /etc/libvirt/. Интересует каталог storage и qemu.

1. Редактирование хранилища kvm_storage:

```
virsh pool-edit kvm_storage
```

2. Редактирование VM:

```
virsh edit VM
```

P.S. могут возникнуть проблемы с составными qcow2 образами.

Проброс видеокарты

Современное состояние дел (см. ссылку).

- Запрет на загрузку модулей ядра видео и аудио

```
blacklist amdgpu
# случай, когда встроенная карта
# и  hdmi  карта используют единый драйвер.
options snd_hda_intel enable=1,0
```

- Добавление в праметры ядра /etc/default/grub «GRUB_CMDLINE_LINUX_DEFAULT=«amd_iommu=on» или «intel_iommu=on».
- Добавление через virt-manager'е двух устройств видеокарты (видео и hdmi аудио).
- Загрузка ОС (Windows), установка драйверов видео карты.

Проблема 1: подключенный внешний монитор находится в выключенном состоянии. На втором виртуальном, не обращая внимания на сию оказию необходимо установить драйвера и произвести обнаружение подключённых дисплеев. После процедуры на физическом мониторе отобразится второй экран виртуальной машины, подключённый через реальную видео карту.

Проблема 2: при старте VM вылетает ошибка проброса оборудования. В данном случае решилось обновлением биоса материнской платы ASUS Sabertooth 990FX R2.0 с версии 1903 до версии 2901.

Permission denied

- https://superuser.com/questions/298426/kvm-image-failed-to-start-with-virsh-permission-denied
- disable security_driver link
- user groups: kvm libvirt libvirt-qemu

Cockpit over Apache proxy

- проблема «invalid handshake» https://github.com/cockpit-project/cockpit/issues/2053

- решение по ссылке
- другая web-panel https://github.com/retspen/webvirtcloud

Конвертация VM

Инструмент:

- virt-v2v;

Инструкции:

- virt-v2v Windows VM migration pre-requisite

From:
https://www.jurik-phys.net.ru/ - **Jurik-Phys.Net.Ru**

Permanent link:
https://www.jurik-phys.net.ru/itechnology:kvm

Last update: **2025/08/27 16:41**