Некоторое время назад озаботился решением вопроса бэкапа виртуальных машин kvm. Когда стал разбираться в теме, с удивлением обнаружил, что каких-то удобных, устоявшихся решений для горячего бэкапа виртуальных машин без остановки работы в kvm нет. Ни коммерческих решений не увидел, ни бесплатных. Пришлось самому вникать в суть и разбираться в теме.

Содержание:

Введение

С гипервизором kvm лично мне приходилось мало работать. Еще в первое мое знакомство с гипервизорами, когда я выбирал, какой использовать в своей работе, kvm мне не понравился вообще по следующим причинам:

1. Нет удобного инструмента для управления гипервизором под Windows. Для меня это критично, так как моя основная рабочая система ОС Windows.
2. Не увидел готового образа, который бы позволил быстро и без лишних движений развернуть гипервизор на новом железе.

В итоге я остановился на Xen там, где нужно поставить систему на программный рейд mdadm, и Hyper-V в тех случаях, где рейд либо не нужен совсем, либо он аппаратный. В своей работе так или иначе приходится сталкиваться с различными системами, поэтому разбираться приходится во всем, в том числе и в kvm.

Есть 2 различных способа сделать бэкап виртуальной машины kvm:

1. С остановкой или заморозкой системы на короткий промежуток времени.
2. Без остановки системы вообще.

Для первого случая все относительно просто, каких-то нюансов нет. Во втором случае возникают нюансы и вопросы. Просто взять и сделать горячий бэкап виртуальной машины в kvm не получится. Вопрос рассмотрения архивных копий виртуальных машин нужно начинать с выбора типа жесткого диска, который будет использоваться. В зависимости от этого будет разная стратегия бэкапа. С этого мы и начнем.

> Сразу хочу пояснить, что не имею большого опыта в работе с kvm и возможно что-то делаю неправильно. Данная статья мой опыт самостоятельного поиска решения и закрепление его для себя и всех, кому это будет интересным.

Какой диск выбрать в kvm — lvm, raw (img) или qcow2

В kvm есть несколько типов дисков, которые вы можете использовать в своей работе. Они имеют принципиальные отличия как по своей работе, так и по способам бэкапа. Выбирать тот или иной тип нужно в зависимости от ваших задач. Рассмотрим эти типы подробнее.

LVM. Тут все понятно. Использование lvm томов в виде дисков виртуальных машин. Такой диск будет блочным устройством и должен работать быстрее всех остальных типов, так как нет лишней прослойки в виде файловой системы.

Плюсы:

- Снэпшоты средствами самого lvm, с них легко снять бэкап без остановки виртуальной машины.
- Максимальное быстродействие.
- Легко изменить размер диска.

Минусы:

- Более сложное управление по сравнению с дисками в виде отдельных файлов.
- Менее гибкое управление пространством диска. Если весь диск разбить на lvm разделы для виртуалок, то места на нем не останется, даже если виртуалки будут пустые.

- Более сложный перенос на другой сервер.

RAW. Это обычный формат сырых данных. Среди дисков в виде файлов это будет самый простой и быстрый вариант. Все данные пишутся как есть без дополнительных обработок и служебной информации. Формат является универсальным для большинства популярных гипервизоров.

Плюсы:

- Максимальная простота и производительность среди образов в виде файла.
- Универсальный формат с поддержкой в большинстве гипервизоров.
- Легкость переноса виртуальной машины на другой сервер. Достаточно просто скопировать файл.

Минусы:

- Не поддерживает снепшоты ни в каком виде.
- Занимает сразу все выделенное пространство на диске, даже если внутри виртуальной машины место будет свободно. Из-за отсутствия фрагментации в некоторых случаях это может оказаться плюсом.

QCOW2. Родной формат для гипервизора QEMU. Расшифровывается как QEMU Copy-on-write. Этот формат позволяет создавать динамические диски для виртуальных машин, а так же поддерживает снепшоты. Теоретически, скорость работы должна хоть и не сильно, но уступать RAW. Как обстоит дело на практике, я не знаю, не замерял и подробно эту информацию не проверял.

Плюсы:

- Поддержка снепшотов и динамических дисков. Как следствие — более удобное управление дисковым пространством.
- Легкость переноса виртуальной машины на другой сервер. Достаточно просто скопировать файл.

Минусы:

- Более низкая производительность, по сравнению с другими типами образов.

У каждого типа есть свои преимущества и недостатки. Lvm проще всего бэкапить, но им сложнее всего управлять. Для того, кто хорошо знаком с lvm это не проблема, если сталкиваешься первый раз, то возникает много вопросов. У raw нет снепшотов, лично для меня это большой минус, я этот формат вообще не рассматриваю. Если нет максимальной нагрузки на дисковую подсистему, то QCOW2 мне кажется наиболее приемлемым вариантом. Собственно, ниже и пойдет рассказ, как совместить удобство управления QCOW2 и простоту бэкапов и снэпшотов lvm. Я покажу, как сделать живой бэкап

виртуальной машины kvm в формате qcow2 без остановки виртуальной машины.

Бэкап виртуальной машины kvm без остановки

Перед практической реализацией живого бэкапа виртуальной машины, снова немного поговорим о теории. Для бэкапа виртуальной машины kvm достаточно просто скопировать ее диски в виде файлов. Это можно сделать даже при работающей машине. Но если в этот момент на виртуалке идет какая-то дисковая активность, вы скорее всего получите неработающий образ. И это логично. Значит нам, чтобы гарантированно скопировать диск, виртуальную машину нужно выключить, либо придумать что-то еще.

Это «что-то еще» является снепшот. Мы можем сделать во время бэкапа снепшот виртуальной машины, зафиксировав ее состояние и сделать копию с этого снепшота. После завершения копирования снепшот объединяется с основным файлом и машина продолжает работать в штатном режиме. Завершать работу или замораживать виртуальную машину в данном случае нет необходимости.

С самими снепшотами тоже есть нюансы. Есть 2 разных типа снепшота. Например, если вы сделаете снепшот qcow2 средствами virt-manager, то удивитесь, не увидев новый файл снепшота. Оказывается, снепшот будет создан внутри qcow2, а файл как был один, так и останется. Это очень неудобно, так как если у вас будет занят весь доступный для файла объем, вы гарантированно получите ошибку при старте вашей виртуалки. А контролировать объем диска, в котором находятся снепшоты трудно и неудобно. И бэкапы в таком случае делать тоже не получится. Хотя получится, но все равно не удобно. Нужно будет отдельными командами конвертировать состояние виртуальной машины до снепшота в отдельный файл и его уже забирать для бэкапа. Плюс ко всему машину в этом случае нужно будет заморозить на момент создания снимка. Заморозка будет кратковременной, но все равно без нее не обойтись. Я приведу для примера команды, которыми это можно делать, но сам я не стал использовать такой способ, потому что, во-первых, не смог найти команды на объединения снепшота и основного файла, во-вторых, меня не устраивает даже кратковременная заморозка системы.

Замораживаем виртуалку:

```
# virsh domfsfreeze vm-name
```

Делаем снепшот:

```
# qemu-img create -f qcow2 -b vm-name.qcow2 vm-name-snapshot.qcow2
```

Размораживаем

```
# virsh domfsthaw vm-name
```

Конвертируем снепшот в отдельный образ:

```
# qemu-img convert -O raw vm-name-snapshot.qcow2 /backup-vm/vm-name-snapshot.img
```

Преобразовываем raw обратно в qcow2:

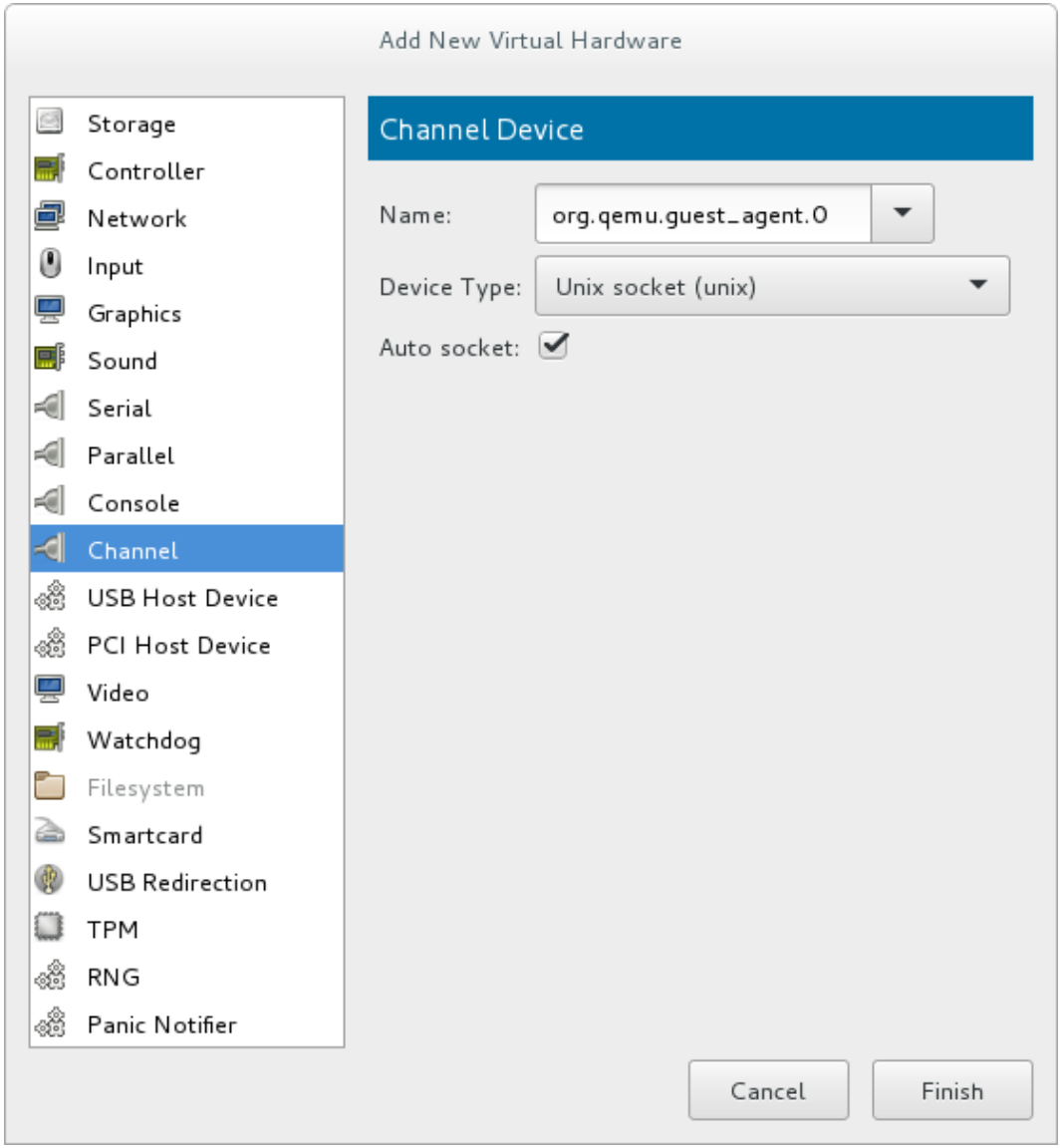
```
# qemu-img convert -O qcow2 /backup-vm/vm-name-snapshot.img /backup-vm/vm-name-snapshot.qcow2
```

После этого надо объединить снимок с остальным файлом. Я не стал искать, как это сделать, так как тут используется заморозка виртуальной машины, хоть и кратковременная, но я хочу обойтись без нее. Поэтому я стал дальше искать варианты.

В рунете я вообще не нашел информации, как все сделать красиво, удобно и быстро. Нашел инструкцию на официальном сайте libvirt — http://wiki.libvirt.org/page/Live-disk-backup-with-active-blockcommit Дальше фактически будет перевод с моими пояснениями.

Установка qemu-guest-agent

Мы будем делать снепшот средствами **virsh**. При этом остановки или заморозки системы не будет вообще. Чтобы этот вариант корректно работал, вам необходимо установить на виртуальную машину **qemu-guest-agent**. Для этого вам сначала нужно добавить Channel Device по имени **org.qemu.guest_agent.0**. Проще всего это сделать через virt-manager.


Add New Virtual Hardware
Storage
Controller
Network
Input
Graphics
Sound
Serial
Parallel
Console
Channel
USB Host Device
PCI Host Device
Video
Watchdog
Filesystem
Smartcard
USB Redirection
TPM
RNG
Panic Notifier
Channel Device
Name:
org.qemu.guest_agent.0
Device Type:
Unix socket (unix)
Auto socket:
Cancel
Finish

Затем в систему надо установить пакет qemu-guest-agent. Для debian и centos все просто:

```
# apt-get install qemu-guest-agent
```

```
# yum install qemu-guest-agent
```

Для windows надо с диска virtio-win установить пакет qemu-ga из папки guest-agent, которая находится в корне диска.

Подробнее об установке qemu-guest-agent можно прочитать в документации redhat. Теперь у нас все готово для выполнения живого бэкапа виртуальной машины в kvm без остановки этой виртуалки.

Выполняем снепшот виртуальной машины:

```
# virsh snapshot-create-as --domain vm-name backup-snapshot -diskspec vda,file=/snapshot/vm-name-backup.qcow2 --disk-only --atomic --quiesce --no-metadata
```

vm-name	имя виртуальной машины
backup-snapshot	название снэпшота, может быть любым
vda	имя диска, для которого указываем адрес снепшота
/snapshot/vm-name-backup.qcow2	путь и имя файла для снепшота

Для того, чтобы узнать имена дисков виртуальной машины, воспользуйтесь командой:

```
# virsh domblklist vm-name
```

> Обращаю внимание на важный момент, который я сначала упустил и долго не мог разобраться в логике создания снепшота. Параметр **diskspec** не указывает для какого диска создается снепшот, а просто обозначает путь до снепшота. Если у вас несколько дисков, а в diskspec вы указали только один диск, то снепшоты все равно будут созданы для всех дисков. При этом для тех дисков, где вы указали путь, будет использоваться указанный, а для остальных будет по-умолчанию в той же папке, где лежит исходный файл ВМ. Этот нюанс

серьезно меняет логику работы скриптов для автоматизации бэкапа в том случае, если у машин несколько дисков. Написав скрипт для бэкапа машин с одним диском, у меня не получилось его быстро оптимизировать для нескольких.

После того, как сделали снимок, скопируйте куда вам нужно основной файл виртуальной машины. Я предпочитаю его сразу сжимать всеми ядрами процессора с помощью архиватора **pigz**:

```
# pigz -c /virt/vm-name.qcow2 > /backup-vm/vm-name.qcow2.gz
```

Когда выполнение бэкапа завершено, объединим снапшот с основным файлом:

```
# virsh blockcommit vm-name vda --active --verbose --pivot
```

После этого файл снэпшота можно удалить:

```
# rm /snapshot/vm-name-backup.qcow2
```

Для полноты картины забэкапим еще и настройки виртуалки:

```
# virsh dumpxml vm-name > /backup-vm/vm-name.xml
```

И на этом все. Мы сделали полный бэкап работающей виртуальной машины kvm без ее остановки. Сама она даже не поняла, что с ней что-то сделали. Работающие БД на ней не испытывают никаких проблем. Проверял на postgresql. Терминальные сессии в windows server тоже чувствуют себя отлично.

Скрипт для автоматического бэкапа виртуальных машин KVM

По мотивам всего написанного выше я набросал простенький скрипт для автоматического живого бэкапа всех дисков всех работающих в момент выполнения скрипта виртуальных машин. Обращаю внимание, что бэкапятся все диски. Скрипт привожу просто для примера, не используйте его, если вам не нужны полные бэкапы всех дисков. Лично я бэкаплю только небольшие системные диски. Все, что много весит, предпочитаю архивировать другими способами уже изнутри виртуальной машины. Если это базы данных, то делаю бэкапы нужных баз postgresql, если это файловый сервер, то использую

rsync для создания инкрементных бэкапов. Ну и так далее. Каждый выбирает средства на свой вкус.

```
#!/bin/bash

# Дата год-месяц-день
data=`date +%Y-%m-%d`
# Папка для бэкапов
backup_dir=/backup-vm
# Список работающих VM
vm_list=`virsh list | grep running | awk '{print $2}'`
# Список VM, заданных вручную, через пробел
#vm_list=(vm-1 vm-2)
# Лог файл
logfile="/var/log/kvmbackup.log"

# Использовать это условие, если список VM задается вручную
#for activevm in "${vm_list[@]}";
# Использовать это условие, если список работающих VM берется автоматически
for activevm in $vm_list
    do
        mkdir -p $backup_dir/$activevm
        # Записываем информацию в лог с секундами
        echo "`date +"%Y-%m-%d_%H-%M-%S"` Start backup $activevm" >> $logfile
        # Бэкапим конфигурацию
        virsh dumpxml $activevm > $backup_dir/$activevm/$activevm-$data.xml
        echo "`date +"%Y-%m-%d_%H-%M-%S"` Create snapshots $activevm" >> $logfile
        # Список дисков VM
        disk_list=`virsh domblklist $activevm | grep vd | awk '{print $1}'`
        # Адрес дисков VM
        disk_path=`virsh domblklist $activevm | grep vd | awk '{print $2}'`
        # Делаем снепшот дисков
        virsh snapshot-create-as --domain $activevm snapshot --disk-only --atomic --quiesce --no-metadata
        sleep 2
```

```
        for path in $disk_path
            do
                echo "`date +"%Y-%m-%d_%H-%M-%S"` Create backup $activevm $path" >> $logfile
                # Вычленяем имя файла из пути
                filename=`basename $path`
                # Бэкапим диск
                pigz -c $path > $backup_dir/$activevm/$filename.gz
                sleep 2
            done
        for disk in $disk_list
            do
                # Определяем путь до снепшота
                snap_path=`virsh domblklist $activevm | grep $disk | awk '{print $2}'`
                echo "`date +"%Y-%m-%d_%H-%M-%S"` Commit snapshot $activevm $snap_path" >> $logfile
                # Объединяем снепшот
                virsh blockcommit $activevm $disk --active --verbose --pivot
                sleep 2
                echo "`date +"%Y-%m-%d_%H-%M-%S"` Delete snapshot $activevm $snap_path" >> $logfile
                # Удаляем снепшот
                rm $snap_path
            done
        echo "`date +"%Y-%m-%d_%H-%M-%S"` End backup $activevm" >> $logfile
    done
```

Обращаю внимание на несколько моментов:

vm_list может либо автоматически заполняться списком работающих виртуальных машин, либо можно вручную указать список только тех машин, которые нужно бэкапить. Для этого нужно раскомментировать соответствующую строку в начале скрипта при объявлении переменной. Затем выбрать подходящую строку для цикла for. Он будет немного разным, в зависимости от списка.

Снепшоты будут создаваться в той же папке, где хранится основной файл данных. На время отладки советую для начала закомментировать строку с удалением снепшота, на всякий случай. Рекомендую этот крипт прогнать сначала на тестовом сервере. Я его написал на примере одного гипервизора,

больше нигде не проверял. Но работает он однозначно, я проверил несколько раз с разными списками VM. В итоге оставил его у себя на сервере в работе.

Для регулярной очистки бэкапов, если они у вас будут храниться где-то в доступном с этого сервера месте, можно добавить в самый конец условие очистки. Для полного бэкапа VM мне видится нормальной схема хранения бэкапов в несколько месяцев, с созданием архивов раз в неделю. Чаще бэкапить системные диски не вижу смысла, там редко бывают ежедневные изменения.

```
/usr/bin/find /backup-vm/*/ -type f -mtime +90 -exec rm -rf {} \;
```

Данная команда удалит все файлы старше 90 дней в папках с названиями виртуальных машин, расположенных в */backup-vm*.

Заключение

В начале статьи я сказал, что не встретил готового решения по живому бэкапу kvm. (upd. Уже после публикации статьи мне подсказали готовое решение kvmBackup.) На самом деле это не совсем верно. Есть хорошая готовая сборка на основе kvm — **proxmox**. Я подробно рассматривал ее в отдельном материале — Установка и настройка proxmox. Но все же это решение конкретного коллектива разработчиков со своими возможностями и ошибками. В проксмокс реализован живой бэкап виртуальных машин из коробки. К сожалению, я не смог быстро найти техническую реализацию их решения. Возможно, все было бы еще проще. Но так тоже ничего получилось.

Важное дополнение в самом конце. Не забывайте проверять возможность восстановления систем из ваших резервных копий. Ведь

конечная цель не сделать бэкап, а восстановиться из него!!!

Помогла статья? Есть возможность отблагодарить автора