Технический справочник. Снимки виртуальных машин

1. Снимки

Снимки - это функция хранилища, которая позволяет администратору создать точку восстановления ОС, приложений и данных виртуальной машины на определенный момент времени. Снимки сохраняют данные, присутствующие в настоящий момент в образе жесткого диска виртуальной машины, в виде тома COW, что позволяет восстановить данные, существовавшие на момент создания снимка. При создании снимка поверх текущего слоя создается новый слой COW. Все операции записи, выполняемые после создания снимка, записываются в новый слой COW.

Важно понимать, что образ жесткого диска виртуальной машины представляет собой цепочку из одного или нескольких томов. С точки зрения виртуальной машины эти тома выглядят как один образ диска. Виртуальная машина не обращает внимания на то, что ее диск состоит из нескольких томов.

Термины "том COW" и "слой COW" используются взаимозаменяемо, однако "слой" точнее отражает временный характер снимков. Каждый снимок создается для того, чтобы администратор смог отменить неудовлетворительные изменения, внесенные в данные после его создания. Снимки обеспечивают функциональность, аналогичную функции Отменить (Undo), присутствующей во многих текстовых процессорах.

Снимки жестких дисков виртуальных машин, помеченных как общие (shareable), а также тех, которые основаны на подключении Прямых LUN (Direct LUN), нельзя сделать ни "на лету", ни как-либо иначе.

Три основные операции с снимками:

- Создание, которое включает в себя первый снимок, созданный для виртуальной машины.
- Предварительный просмотр, который включает в себя собственно предварительный просмотр снимка с целью определить, следует ли восстанавливать системные данные на момент времени, когда был сделан этот снимок.
- Удаление, которое включает в себя удаление точки восстановления, которая больше не требуется.

Сведения об операциях с снимками применительно к конкретным задачам см. в разделе Снимки Руководства по управлению виртуальными машинами.

2. Снимки "на лету" в zVirt

Снимки жестких дисков виртуальных машин, помеченных как **общие** (**shareable**), а также тех, которые основаны на подключении **Прямых LUN** (**Direct LUN**), нельзя сделать ни "на лету", ни как-либо иначе.

Снимок любой другой виртуальной машины, которая не находится в процессе клонирования или миграции, можно сделать во время ее работы, приостановки или остановки.

Когда инициируется создание снимка виртуальной машины "на лету", Менеджер управления отправляет хосту SPM запрос на создание нового тома, который будет использоваться виртуальной машиной. Когда новый том готов, Менеджер управления использует VDSM для связи с libvirt и qemu на хосте, где запущена виртуальная машина, чтобы он начал использовать новый том для операций записи виртуальной машины. Если виртуальная машина может писать в новый том, то операция создания снимка считается успешной и виртуальная машина прекращает писать в предыдущий том. Если виртуальная машина не может писать в новый том, то операция создания снимка считается неуспешной и новый том удаляется.

С момента начала создания снимка "на лету" до момента готовности нового тома виртуальной машине требуется доступ как к своему текущему, так и к новому тому, поэтому оба тома открыты как на чтение, так и на запись.

Виртуальные машины с установленным гостевым агентом, поддерживающим "заморозку", могут обеспечить согласованность файловой системы по всем снимкам. Зарегистрированные гостевые машины Linux могут устанавливать qemu-guest-agent, чтобы включить "заморозку" перед созданием снимков.

Если во время создания снимка на виртуальной машине присутствует гостевой агент, совместимый с "заморозкой", то VDSM использует libvirt для связи с агентом, чтобы подготовиться к созданию снимка. Незавершенные операции записи завершаются, после чего файловые системы "замораживаются" перед созданием снимка. Когда создание снимка завершено и libvirt переключает виртуальную машину на новый том для выполнения операций записи на диск, файловая система "размораживается" и запись на диск возобновляется.

Все попытки создания снимков "на лету" делаются с включенной "заморозкой". Если команда создания снимка завершается неудачно из-за отсутствия совместимого гостевого агента, то создание снимка "на лету" инициируется повторно без использования флага "заморозки использования". Когда виртуальная машина возвращается в состояние до создания снимка с "замороженными" файловыми системами, выполняется ее чистая

загрузка и проверка файловой системы не требуется. Если же виртуальная машина восстанавливается с предыдущего снимка без "заморозки" файловой системы, то требуется проверка файловой системы при загрузке.

3. Создание снимка

В zVirt первоначальный снимок виртуальной машины отличается от последующих снимков тем, что первоначальный снимок сохраняет свой формат: либо QCOW2, либо RAW. Первый снимок виртуальной машины использует существующие тома в качестве базового образа. Дополнительные снимки - это дополнительные слои COW, отслеживающие изменения, которые после создания предыдущего снимка были внесены в данные, хранящиеся в образе.

Как показано на рисунке, при создании снимка - тома, составляющие виртуальный диск, служат базовым образом для всех последующих снимков.

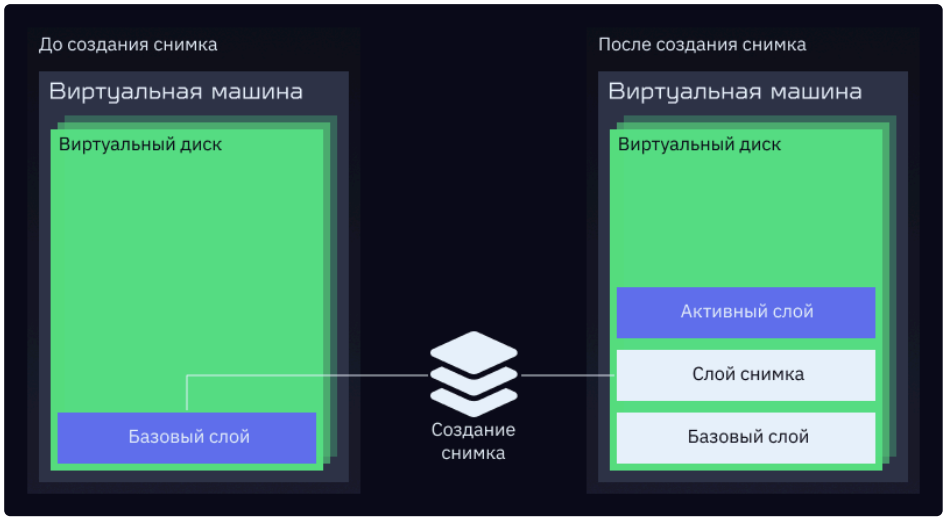

Рисунок 1. Создание первоначального снимка

При создании последующих (после первоначального) снимков создаются новые тома COW, в которых будут храниться данные, созданные или измененные после создания снимка. Каждый вновь создаваемый слой COW содержит только метаданные COW. Данные, созданные в процессе использования и работы виртуальной машины после создания снимка, пишутся в этот новый слой COW. Когда виртуальная машина используется для изменения данных, существующих в предыдущем слое COW, данные читаются с предыдущего слоя и пишутся на самый новый слой. Виртуальные машины находят данные, проверяя каждый слой COW от самого нового до самого старого, причем это происходит прозрачно для виртуальной машины.

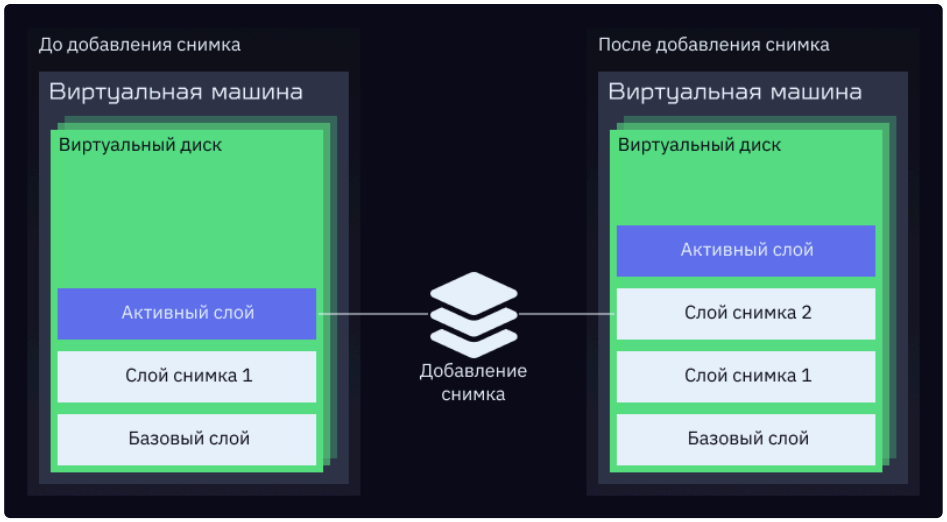

Рисунок 2. Создание дополнительных снимков

4. Предварительный просмотр снимков

Чтобы выбрать снимок для восстановления виртуального диска с него, администратор может предварительно просмотреть все ранее созданные снимки.

Из доступных каждой гостевой машине снимков администратор может выбрать том снимка для предварительного просмотра его содержимого. Как показано на рисунке, каждый снимок сохраняется как том COW, и при предварительном просмотре новый слой предварительного просмотра копируется из просматриваемого снимка. Гостевая машина взаимодействует с предварительным просмотром, а не с фактическим томом снимка.

После предварительного просмотра выбранного снимка администратор может "закрепить" слой предварительного просмотра и восстановить с него данные гостевой машины до состояния, зафиксированного на снимке. В таком случае гостевая машина подключается к слою предварительного просмотра.

Завершив предварительный просмотр снимка, администратор может выбрать **Отменить** (**Undo**), чтобы удалить слой предварительного просмотра снимка. Слой, содержащий сам снимок, сохраняется, несмотря на удаление слоя предварительного просмотра.

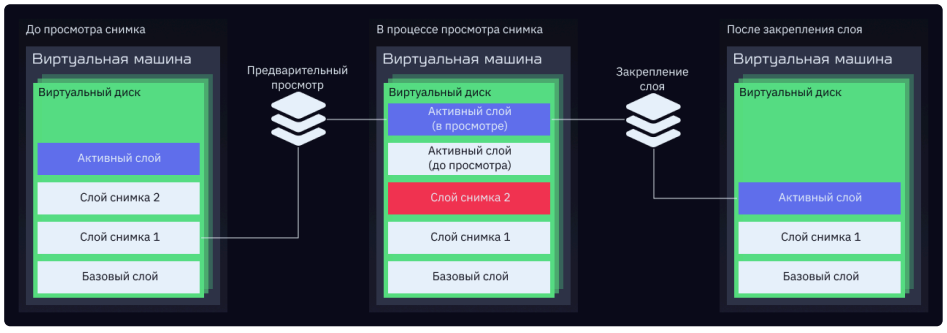

Рисунок 3. Предварительный просмотр снимков

5. Удаление снимка

Отдельные снимки, которые больше не нужны, можно удалить. После удаления снимка нельзя уже будет восстановить виртуальный диск до соответствующей точки восстановления. При этом не обязательно освобождается место на диске, которое занимал снимок, и не удаляются данные. Дисковое пространство будет освобождено только в том случае, если последующий снимок перезапишет данные удаленного. Если, например, удаляется третий снимок из пяти, то неизмененные данные третьего снимка должны быть сохранены на диске, чтобы можно было использовать четвертый и пятый снимки. Однако если четвертый или пятый снимок перезаписал данные третьего, то третий снимок становится избыточным и дисковое пространство можно освободить. Помимо возможного освобождения дискового пространства, удаление снимка также может повысить производительность виртуальной машины.

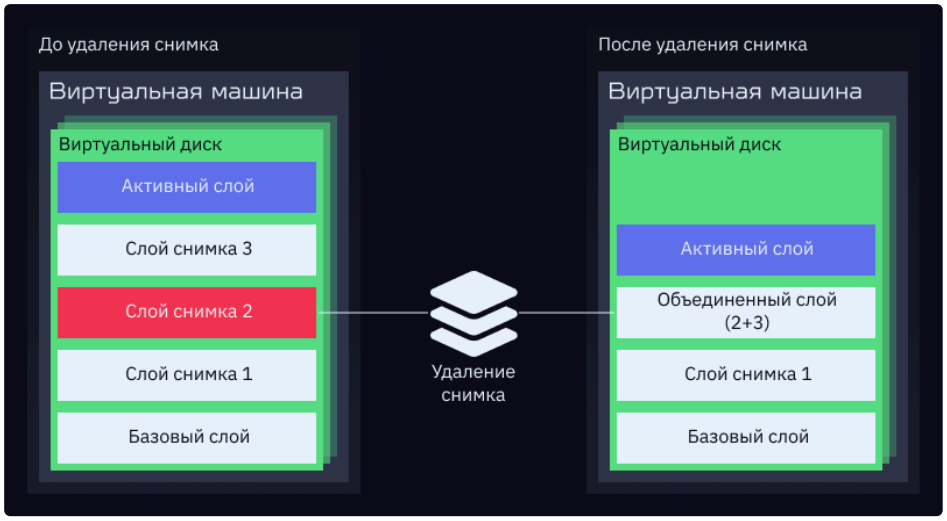

Рисунок 4. Удаление снимка

Удаление снимка обрабатывается как асинхронное блочное задание, в котором VDSM протоколирует операцию в файле восстановления для виртуальной машины, чтобы задание можно было отследить, даже если VDSM перезапустится или виртуальная машина выключится во время операции. После начала операции удаляемый снимок нельзя просмотреть или использовать в качестве точки восстановления, даже если операция завершится сбоем или будет прервана. В операциях, в которых активный слой должен быть объединен со своим родительским слоем, операция разбивается на два этапа, во время которых данные копируются с активного слоя на родительский, а запись на диск зеркалируется как на активном, так и на родительском слое. Наконец, задание считается выполненным после того, как данные в удаляемом снимке будут объединены с его родительским снимком, а VDSM синхронизирует изменения по всей цепочке образов.

Если попытка удаления неудачна, устраните основную проблему (например, отказ хоста, недоступность устройства хранения или даже временную проблему с сетью) и повторите попытку.

Технический справочник. Хранилище

1. Обзор доменов хранения

Домен хранения - это набор образов, имеющих общий интерфейс хранения. Домен хранения содержит полные образы шаблонов и виртуальных машин (включая снимки), файлы ISO и их метаданные. Домен хранения может состоять либо из блочных устройств (SAN - iSCSI или FCP), либо из файловой системы (NAS - NFS, GlusterFS или иных POSIX-совместимых файловых систем).

В NAS все виртуальные диски, шаблоны и снимки являются файлами.

В SAN (iSCSI/FCP) каждый виртуальный диск, шаблон или снимок представляет собой логический том. Блочные устройства объединяются в логическую сущность, называемую группой томов, а затем с помощью Менеджера логических томов (Logical Volume Manager, LVM) делятся на логические тома для использования в качестве виртуальных жестких дисков.

Виртуальные диски могут иметь один из двух форматов: **QCOW2** или **raw**. Возможные типы хранилищ: **динамически расширяемые** (**sparse**) или **предварительно размеченные** (**preallocated**). Снимки всегда имеют формат **sparse** , но могут создаваться для дисков любого из этих форматов.

Для виртуальных машин, совместно использующих один домен хранения, возможна миграция между хостами, относящимися к одному кластеру.

2. Типы хранилищ, лежащие в основе доменов хранения

Домены хранения могут базироваться на блочном и файловом хранилищах.

Файловое хранилище

zVirt поддерживает файловые хранилища следующих типов: NFS, GlusterFS, другие POSIX-совместимые файловые системы и локальные хранилища хостов.

Управление файловым хранилищем производится извне среды zVirt. Хранилище NFS управляется NFS-сервером zVirt или другим сторонним сервером хранения, подключенным к сети.

Хосты могут управлять файловыми системами своих собственных локальных хранилищ.

Блочное хранилище

В блочном хранилище используются неформатированные блочные устройства. Блочные устройства объединяются в группы томов с помощью Менеджера логических томов (LVM). Экземпляр LVM работает на всех хостах и не знает об экземплярах, работающих на других хостах. VDSM добавляет логику кластеризации поверх LVM, сканируя группы томов на наличие изменений. Обнаружив изменения, VDSM обновляет отдельные хосты, давая им команду обновить их информацию о группе томов. Хосты делят группу томов на логические тома, записывая метаданные логических томов на диск. Если к существующему домену хранения добавляется дополнительная емкость, Менеджер управления дает команду VDSM на каждом хосте обновить информацию о группе томов.

LUN - это отдельное блочное устройство. Для подключения к LUN используется один из поддерживаемых протоколов блочных хранилищ: iSCSI или Fibre Channel. Менеджер управления управляет программными iSCSI-подключениями к LUN. Управление всеми остальными подключениями к блочному хранилищу производится извне среды zVirt. Любые изменения в блочной среде хранения (такие как создание, расширение или удаление логических томов и добавление нового LUN) обрабатываются Менеджером логических томов (LVM) на специально выбранном хосте, который называется Менеджером пула хранения (Storage Pool Manager, SPM). Затем изменения синхронизируются с помощью VDSM, метаданные хранилища которого обновляются на всех хостах в кластере.

3. Типы доменов хранения

zVirt поддерживает следующие типы доменов хранения, а также типы хранилищ, поддерживаемые каждым доменом хранения:

- **Домен данных** хранит образы жестких дисков всех виртуальных машин в среде zVirt. Образы дисков могут содержать установленную операционную систему или данные, хранящиеся на виртуальной машине или сгенерированные ею. Домены хранения данных поддерживают хранилища NFS, iSCSI, FCP, GlusterFS и POSIX-совместимые хранилища. Домен данных не может использоваться несколькими центрами данных.
- **Домен экспорта** служит для временного хранения образов жестких дисков и шаблонов виртуальных машин, передаваемых между центрами данных. Кроме того, в доменах экспорта хранятся резервные копии виртуальных машин. Домены экспорта поддерживают NFS-хранилища. К одному домену экспорта могут обращаться несколько центров данных, но в конкретный момент времени использовать его может только один центр данных.

Сущность **домен экспорта** считается устаревшей. Домен экспорта можно отключить от центра данных и импортировать в другой центр данных в той же или другой среде. Затем виртуальные машины, "плавающие" виртуальные диски и шаблоны можно выгрузить из импортированного домена хранения в подключенный центр данных.

- **Домен ISO** ранее использовался для хранения файлов ISO. Файлы ISO - это представления физических CD или DVD. В среде zVirt используются в основном такие типы файлов ISO, как установочные диски операционных систем, приложений и гостевых агентов. Эти образы можно подключать к виртуальным машинам и загружать так же, как происходила бы загрузка при установке физического диска в дисковод.

Домен хранения ISO является устаревшим и более не поддерживается.

4. Форматы хранения для виртуальных дисков

Хранилище виртуальной машины в формате QCOW2

QCOW2 - это формат хранения для виртуальных дисков. QCOW расшифровывается как QEMU с копированием при записи. Формат QCOW2 отделяет физический уровень хранения от виртуального уровня, добавляя сопоставление между логическими и физическими блоками. Каждому логическому блоку сопоставляется его физическое смещение, что делает возможным избыточное выделение пространства и создание снимков виртуальных машин, в которых каждый том QCOW представляет только изменения, внесенные в базовый виртуальный диск.

Первоначальным сопоставлением для всех логических блоков задаются смещения в базовом файле или томе. Когда виртуальная машина записывает данные в том QCOW2 после снимка, соответствующий блок считывается из базового тома, изменяется с учетом новой информации и пишется в новый том снимка QCOW2. Затем сопоставление обновляется так, чтобы указывать на новое место.

Raw

Формат RAW имеет более высокую производительность, чем QCOW2, поскольку к виртуальным дискам, хранящимся в формате RAW, не применяется форматирование. Выполнение операций с данными виртуальной машины на виртуальных дисках в формате RAW не требует дополнительных действий от хостов. Когда виртуальная машина пишет данные по заданному смещению на своем виртуальном диске, вводимые/выводимые данные пишутся по тому же смещению в базовый файл или логический том.

Формат RAW требует, чтобы все пространство заданного образа было выделено заранее, если только не используются управляемые извне LUN с динамическим выделением из массива хранения.

5. Политики выделения пространства для виртуальных дисков

Предварительно размеченное пространство

Все пространство, необходимое для виртуального диска, выделяется до создания виртуальной машины. Если для виртуальной машины создается образ диска размером 20 ГБ, то он использует 20 ГБ емкости домена хранения. Предварительно размеченный диск, как правило, обеспечивает более высокую скорость записи, поскольку не требует выполнения операции по выделению пространства, хотя это в свою очередь оборачивается потерей гибкости. Такая политика затрудняет для Менеджера управления избыточное выделение пространства. Предварительно размеченный диск рекомендуется для виртуальных машин, работающих с интенсивным вводом/выводом и чувствительных к задержкам при операциях чтения/записи. Обычно в эту категорию попадают виртуальные серверы.

Если используется функция динамического выделения пространства, которую обеспечивает серверная часть хранилища, то при выделении пространства для виртуальных машин на Портале администрирования по-прежнему следует выбирать предварительно размеченное пространство.

Динамически расширяемое пространство

При создании виртуальной машины задаётся верхний предел размера виртуального диска. Изначально образ диска не использует всю выделенную ёмкость домена хранения. Используемое пространство растет по мере того, как виртуальная машина пишет данные на диск. Это происходит до тех пор, пока не будет достигнут верхний предел. При удалении данных из образа диска емкость не возвращается домену хранения. Данная политика рекомендуется для виртуальных машин, работающих с вводом/выводом средней или низкой интенсивности и не очень чувствительных к задержкам чтения/записи. Обычно в эту категорию попадают виртуальные рабочие станции.

Если серверная часть хранилища обеспечивает функцию динамического выделения пространства, то ее следует использовать как предпочтительный метод динамического выделения. В графическом пользовательском интерфейсе необходимо выбирать политику с предварительно размеченным пространством, а динамическое выделение должно возлагаться на серверную часть.

6. Версии метаданных хранилища в zVirt

zVirt хранит информацию о доменах хранения в виде метаданных в самих доменах хранения. В каждый основной релиз zVirt входят улучшения реализации метаданных хранилища.

Метаданные V1

- Каждый домен хранения содержит метаданные, описывающие его собственную структуру, и все имена физических томов, на которых базируются виртуальные диски.
- Мастер-домены дополнительно содержат метаданные для всех доменов и имен физических томов в пуле хранения. Общий размер этих метаданных не может превышать 2 КБ, что ограничивает количество доменов хранения в пуле.
- Базовые образы шаблонов и виртуальных машин доступны только для чтения.
- Метаданные V1 применимы к доменам хранения NFS, iSCSI и FC.

Метаданные V2

- Все метаданные домена хранения и пула хранятся в виде тегов логического тома, а не записываются на логический том. Метаданные о томах виртуальных дисков по-прежнему хранятся в логических томах доменов.
- Имена физических томов больше не включаются в метаданные.
- Базовые образы шаблонов и виртуальных машин доступны только для чтения.
- Метаданные V2 применимы к доменам хранения iSCSI и FC.

Метаданные V3

- Все метаданные домена хранения и пула хранятся в виде тегов логического тома, а не записываются на логический том. Метаданные о томах виртуальных дисков по-прежнему хранятся в логических томах доменов.
- Базовые образы виртуальных машин и шаблонов теперь доступны не только для чтения. Это изменение делает возможным создание снимков на лету, миграцию хранилища на лету и клонирование из снимка.
- Добавлена поддержка метаданных Unicode для имен томов на языках, отличных от английского.
- Метаданные V3 применимы к доменам хранения NFS, GlusterFS, POSIX, iSCSI и FC.

Метаданные V4

- Поддержка уровней совместимости QCOW2 - формат образа QCOW включает в себя номер версии, что позволяет вводить новые функции, которые изменяют формат образа, делая его несовместимым с более ранними версиями. Более новые версии QEMU (1.7 и выше) поддерживают QCOW2 версии 3, у которой нет обратной совместимости, но есть улучшения, такие как нулевые кластеры и обеспечена повышенная производительность.

- Новый том xleases для поддержки аренды ВМ - эта функция позволяет получать аренду для каждой виртуальной машины в общем хранилище без привязки аренды к диску виртуальной машины.

Аренда ВМ позволяет:

- Избежать ситуации "split-brain".
- Запустить ВМ на другом хосте, если исходный хост перестанет отвечать на запросы, тем самым повышается доступность ВМ с признаком высокой доступности.

Метаданные V5

- Поддержка переменного выравнивания SANLOCK.
- Поддержка новых свойств:
  - `BLOCK_SIZE` - хранит размер блока домена хранения в байтах.
  - `ALIGNMENT` - задает форматирование и размер тома xlease. (1-8 МБ). Определяется максимальным числом поддерживаемых хостов (значение указывается пользователем) и размером блока диска.

    Например, для блока в 512 байт и 2 000 хостов том xlease имеет размер 1 МБ. Для 4 КБ и 2 000 хостов - 8 МБ.

    По умолчанию максимальное количество хостов равно 250, поэтому том xlease имеет размер 1 МБ для дисков с 4-килобайтным блоком.
- Устаревшие свойства:
  - Поля `LOGBLKSIZE`, `PHYBLKSIZE`, `MTIME` и `POOL_UUID` были удалены из метаданных домена хранения.
  - Поле `SIZE` (размер в блоках) было заменено полем `CAP` (размер в байтах).

- Невозможно загрузиться с диска в формате 4КБ, так как загрузочный диск всегда использует эмуляцию 512 байт.
- Формат NFS всегда использует 512 байт.

7. Автоматическое восстановление домена хранения в zVirt

Хосты в среде zVirt ведут мониторинг доменов хранения в своих центрах данных, считывая метаданные из каждого домена. Домен хранения становится неактивным, когда все хосты в центре данных сообщают, что не могут получить доступ к домену хранения.

Вместо того, чтобы отключить неактивный домен хранения, Менеджер управления предполагает, что он стал неактивным на время (например, из-за временного отключения

сети). Каждые 5 минут Менеджер управления пытается повторно активировать все неактивные домены хранения.

Для устранения причины перебоя в подключении к хранилищу может потребоваться вмешательство администратора, но после восстановления подключения Менеджер управления сам позаботится о повторной активации доменов хранения.

8. Storage Pool Manager (SPM)

Для описания внутренней структуры доменов хранения zVirt использует метаданные. Структурные метаданные пишутся в сегмент каждого домена хранения. Хосты работают с метаданными домена хранения по схеме "один пишет, многие читают". Посредством структурных метаданных домена хранения отслеживается создание и удаление образов и снимков, а также расширение томов и доменов.

Хост, который может вносить изменения в структуру домена данных, называется Менеджером пула хранения (Storage Pool Manager, SPM). SPM координирует все изменения метаданных в центре данных, такие как создание и удаление образов дисков, создание и объединение снимков, копирование образов между доменами хранения, создание шаблонов и выделение пространства для блочных устройств. На каждый центр данных имеется один SPM. Все остальные хосты могут только читать структурные метаданные домена хранения.

Хост можно задать в качестве SPM вручную или назначить с помощью Менеджера управления. Менеджер управления назначает роль SPM, инициируя попытку потенциального хоста SPM взять на себя аренду конкретного хранилища. Аренда позволяет хосту SPM писать метаданные хранилища. Словосочетание "конкретного хранилища" означает, что факт аренды записывается в домен хранения, а не отслеживается Менеджером управления или хостами. Факт аренды конкретного хранилища записывается в специальный логический том с именем **leases** в домене хранения `master`. Метаданные о структуре домена данных записываются в специальный логический том с именем **metadata**. Логический том **leases** защищает логический том **metadata** от изменений.

Менеджер управления использует VDSM для выдачи команды **spmStart** хосту, инициируя попытку VDSM на этом хосте взять на себя аренду конкретного хранилища. В случае успеха хост получает роль SPM и сохраняет аренду конкретного хранилища до тех пор, пока Менеджер управления не потребует, чтобы роль SPM взял на себя новый хост.

Менеджер управления переносит роль SPM на другой хост в следующих случаях:

- Хост SPM не может получить доступ ко всем доменам хранения, но может получить доступ к домену хранения **master**
- Хост SPM не может продлить аренду из-за потери подключения к хранилищу или из-за того, что том аренды заполнен и операцию записи не удается выполнить

- Хост SPM аварийно завершает работу

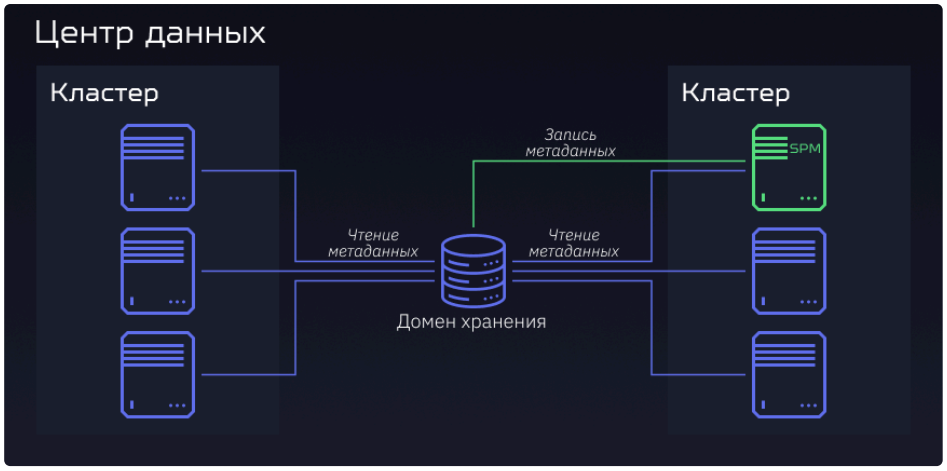

Рисунок 1. Эксклюзивная запись метаданных хостом SPM

9. Процесс выбора Менеджера пула хранения

Если хосту не была назначена роль Менеджера пула хранения (SPM) вручную, то процесс выбора SPM инициируется и управляется Менеджером управления.

Первым делом Менеджер управления запрашивает у VDSM подтверждение того, какой хост взял на себя аренду конкретного хранилища.

Менеджер управления отслеживает историю назначений SPM, начиная с первоначального создания домена хранения. Доступность роли SPM подтверждается тремя способами:

- Команда **getSPMstatus**: Менеджер управления использует VDSM для проверки хоста, который последним имел статус SPM, и получает один из вариантов ответа: "SPM", "Состязающийся (Contending)" или "Свободный (Free)".
- В томе метаданных для домена хранения содержится последний хост со статусом SPM.
- Том метаданных для домена хранения содержит версию последнего хоста со статусом SPM.

Если работоспособный и реагирующий на запросы хост сохраняет аренду конкретного хранилища, то Менеджер управления помечает этот хост как SPM на Портале администрирования. Никаких дополнительных действий не выполняется.

Если хост SPM не отвечает, то он считается недоступным. Если для хоста настроено управление питанием, то он автоматически изолируется. Если же нет, то требуется его ручная изоляция. Роль SPM не может быть назначена новому хосту до тех пор, пока предыдущий SPM не будет изолирован.

Когда роль SPM и аренда конкретного хранилища не будут кем-то заняты, Менеджер управления назначит их случайно выбранному работающему хосту в центре данных.

Если назначить роль SPM на новом хосте не удастся, то Менеджер управления добавит хост в список хостов, на которых эта операция не удалась, отмечая эти хосты как неподходящие на роль SPM. Этот список очищается в начале следующего процесса выбора SPM, чтобы все хосты снова подходили на роль SPM.

Менеджер управления продолжает отправлять запрос на то, чтобы роль SPM и аренду конкретного хранилища принял на себя случайно выбираемый хост, которого нет в списке отказавших хостов, и продолжает так делать до тех пор, пока SPM не будет успешно выбран.

Каждый раз, когда текущий SPM перестает реагировать на запросы или оказывается неспособен выполнять свои обязанности, Менеджер управления инициирует процесс выбора SPM.

10. Эксклюзивные ресурсы и Sanlock в zVirt

Доступ к некоторым ресурсам в среде zVirt должен предоставляться на эксклюзивной основе.

Один из таких ресурсов - роль SPM. Если бы несколько хостов работали как SPM, то возник бы риск повреждения данных, поскольку одни и те же данные можно было бы изменить из двух мест одновременно.

Ранее эксклюзивность SPM поддерживалась и отслеживалась с помощью функции VDSM, называемой **safelease**. Факт аренды записывался в специальную область во всех доменах хранения в центре данных. Все хосты в среде могли отслеживать статус SPM независимо от сети. Безопасная аренда VDSM поддерживала эксклюзивность только одного ресурса: роли SPM.

Sanlock имеет ту же функциональность, но рассматривает роль SPM как один из ресурсов, которые можно заблокировать. **Sanlock** - более гибкий инструмент, поскольку позволяет блокировать дополнительные ресурсы.

Приложения, требующие блокировки ресурсов, могут регистрироваться в Sanlock. Зарегистрированные приложения могут затем запросить у Sanlock блокировку ресурса от их имени, чтобы больше никакое приложение не смогло получить к нему доступ. Например, вместо того, чтобы блокировать статус SPM самостоятельно, VDSM теперь требует, чтобы это сделал Sanlock.

Блокировки отслеживаются в пространстве блокировки **lockspace** диска. Для каждого домена хранения есть одно пространство блокировки. В случае блокировки ресурса SPM работоспособность каждого хоста отслеживается в пространстве блокировки. Индикатором работоспособности хоста служит его способность обновлять свой **hostid** , полученный от Менеджера управления при подключении к хранилищу, и записывать через регулярные

промежутки времени временную метку в пространство блокировки. Логический том **ids** отслеживает уникальные идентификаторы каждого хоста и обновляет свое содержимое всякий раз, когда хост обновляет свой hostid. Ресурс SPM может удерживаться только работоспособным хостом.

Ресурсы отслеживаются в логическом томе **leases** диска. Ресурс считается занятым, когда в его представление на диске добавлен уникальный идентификатор процесса, который его занял. В случае роли SPM ресурс SPM обновляется идентификатором хоста (hostid), который его занял.

Процесс Sanlock на каждом хосте должен лишь единожды проверить ресурсы, чтобы убедиться, что они заняты. После первоначальной проверки Sanlock может вести мониторинг пространств блокировки до тех пор, пока временная метка хоста с заблокированным ресурсом не устареет.

Sanlock ведет мониторинг приложений, которые используют ресурсы: например, мониторинг VDSM в части статуса SPM и hostid. Если хост не может обновить свой hostid из Менеджера управления, то он теряет эксклюзивность по отношению ко всем ресурсам в пространстве блокировки. Sanlock обновляет состояние ресурса, чтобы показать, что он более не занят.

Если хост SPM не может записать временную метку в пространство блокировки в домене хранения в течение заданного периода времени, то экземпляр Sanlock хоста требует, чтобы процесс VDSM освободил свои ресурсы. Если процесс VDSM отвечает, то его ресурсы освобождены и другой хост может занять ресурс SPM в пространстве блокировки.

Если VDSM на хосте SPM не отвечает на запросы на освобождение ресурсов, то Sanlock на хосте останавливает процесс VDSM. Если остановить процесс командой `kill` не удается, Sanlock прибегает к эскалации, пытаясь остановить VDSM командой `sigkill`. Если команда `sigkill` не дает результата, Sanlock ждет, пока демон **watchdog** перезагрузит хост.

Всякий раз, когда VDSM на хосте обновляет свой hostid и записывает временную метку в пространство блокировки, демон **watchdog** получает **pet**-метку. Когда VDSM не может этого сделать, демон **watchdog** перестаёт получать **pet**-метку. Если в течение заданного периода времени демон **watchdog** не получил **pet**-метку, он перезагружает хост. Применение этого последнего уровня эскалации гарантирует, что ресурс SPM освобождается и может быть занят другим хостом.

11. Динамическое выделение и избыточное выделение пространства

Менеджер управления применяет политики выделения ресурсов для оптимизации использования хранилища в среде виртуализации. Политика динамического выделения

позволяет выделять пространство с избытком, исходя из его фактического использования в среде виртуализации.

Избыточное выделение пространства - это выделение виртуальным машинам большего объема хранилища, чем физически доступно в пуле хранения. Как правило, виртуальные машины используют меньше пространства, чем им было выделено. Динамическое выделение позволяет виртуальной машине работать так, как если бы заданное для нее хранилище было выделено полностью, хотя на самом деле выделена лишь его часть.

Несмотря на то, что в Менеджере управления есть своя функция динамического выделения пространства, следует использовать ту функциональность динамического выделения, которую предоставляет (если предоставляет) серверная часть хранилища.

Для поддержки избыточного выделения пространства VDSM определяет пороговое значение, используемое при сравнении логически выделенного пространства хранилища с фактически используемым. Это пороговое значение гарантирует, что объем данных, записываемых в образ диска, будет меньше размера логического тома, на котором базируется образ диска. QEMU определяет наибольшее смещение, записанное в логическом томе и указывающее на самый дальний сектор диска, который можно использовать. VDSM отслеживает наибольшее смещение, отмеченное QEMU, чтобы гарантировать, что использование пространства не превысит указанного порогового значения. Все время, пока VDSM продолжает сообщать, что наибольшее смещение остается ниже порогового значения, Менеджер управления знает, что на рассматриваемом логическом томе есть достаточно места для продолжения операций.

Когда QEMU указывает, что использование превысило пороговое значение, VDSM сообщает Менеджеру управления, что размер образа диска скоро достигнет размера своего логического тома. Менеджер управления запрашивает у хоста SPM, чтобы тот расширил логический том. Этот процесс можно повторять, пока в домене хранения данных для центра данных есть свободное пространство. Когда свободное пространство в домене хранения данных заканчивается, необходимо вручную добавить емкость в хранилище, чтобы расширить его.

12. Расширение логического тома

Менеджер управления использует динамическое выделение для избыточного выделения пространства, доступного в пуле хранения, и выделяет больше пространства, чем доступно физически. По мере работы виртуальные машины пишут данные. Виртуальная машина с образом диска, использующим динамическое выделение, в конечном счете попытается записать больше данных, чем может вместить логический том, на котором базируется этот образ диска. Когда это происходит, для предоставления виртуальной машине дополнительного пространства и обеспечения ее непрерывной работы применяется расширение логического тома.

В zVirt используется механизм динамического выделения через LVM. В случае хранилища в формате QCOW2 zVirt использует системный процесс хоста **qemu-kvm** для последовательного сопоставления блоков хранилища на диске с логическими блоками. Это позволяет, например, задать логический диск объемом 100 ГБ, базирующийся на логическом томе объемом 1 ГБ. Когда qemu-kvm превышает порог использования, установленный VDSM, локальный экземпляр VDSM отправляет Менеджеру пула хранения запрос на расширение логического тома еще на один гигабайт. VDSM на хосте, где запущена виртуальная машина, нуждающаяся в расширении тома, уведомляет VDSM Менеджера пула хранения о том, что ей требуется больше места. SPM расширяет логический том, а его экземпляр VDSM заставляет VDSM хоста обновлять информацию о группе томов и подтвердить, что операция расширения завершена. Хост может продолжать работу.

Для расширения логического тома не нужно, чтобы хост знал, какой другой хост выполняет роль SPM; это может быть даже сам SPM. Для обмена сообщениями о необходимости расширения хранилища используются логические тома **inbox** и **outbox**. **Inbox** и **outbox** - это выделенные логические тома в домене хранения данных. Хост, которому нужно, чтобы SPM расширил логический том, пишет сообщение в пространство **inbox**, выделенное для этого конкретного хоста. SPM периодически читает **inbox**, выполняет запрошенные расширения логических томов и пишет ответ в **outbox**. После отправки запроса хост проверяет входящие сообщения на наличие ответов каждые две секунды. Получив ответ об успешном выполнении его запроса на расширение логического тома, хост обновляет карту логического тома в инструменте сопоставления устройств, чтобы увидеть вновь выделенное пространство.

Когда объем физического хранилища, доступный пулу хранения, почти исчерпан, несколько образов могут исчерпать используемое ими пространство без возможности его расширения. Когда пул хранения исчерпывает свое пространство, QEMU возвращает ошибку **enospc**, которая указывает на то, что на устройстве больше нет доступного пространства. В этот момент работающие виртуальные машины автоматически приостанавливаются, и требуется ручное вмешательство, чтобы добавить новый LUN в группу томов.

Когда в группу томов добавляется новый LUN, Менеджер пула хранения автоматически распределяет дополнительное пространство между логическими томами, которые в нем нуждаются. Автоматическое выделение дополнительных ресурсов позволяет соответствующим виртуальным машинам автоматически продолжать работу без перерыва либо возобновить работу после остановки.

13. Влияние действий в домене хранения на емкость хранилища

Включение, выключение и перезагрузка виртуальной машины без запоминания состояния

Эти три процесса влияют на слой копирования при записи (COW) в виртуальной машине без запоминания состояния. Дополнительные сведения см. в строке **Без запоминания состояния (Stateless)** таблицы Виртуальная машина: общие настройки в описании настроек виртуальных машин.

Создание домена хранения

Создание домена блочного хранения сопровождается созданием файлов с теми же именами, что и у семи логических томов, показанных ниже, и изначально должно требовать меньше места.

```
ids             64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-ao---- 128.00m
inbox           64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-a----- 128.00m
leases          64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-a-----   2.00g
master          64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-ao----   1.00g
metadata        64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-a----- 512.00m
outbox          64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-a----- 128.00m
xleases         64f87b0f-88d6-49e9-b797-60d36c9df497 -wi-a-----   1.00g
```

Удаление домена хранения

Удаление домена хранения высвобождает столько дискового пространства, сколько занимал удаленный процесс.

Миграция домена хранения

При миграции домена хранения дополнительная емкость хранилища не используется. Дополнительную информацию о миграции доменов хранения см. в разделе Миграция доменов хранения между центрами данных в одной среде в Руководстве по администрированию.

Перемещение виртуального диска в другой домен хранения

Для миграции виртуального диска требуется достаточно свободного места в целевом домене хранения. Приблизительный размер свободного места в целевом домене можно посмотреть на Портале администрирования.

Видимая емкость зависит от типа перемещаемого хранилища. Например, если переносить диск с предварительно выделенным пространством из блочного хранилища в файловое, то получившееся в результате свободное пространство может быть значительно меньше первоначального.

При миграции виртуального диска в другой домен хранения на лету также создается снимок, который автоматически объединяется после завершения миграции. Дополнительную информацию о перемещении виртуальных дисков см. в разделе Перемещение виртуального диска в Руководстве по администрированию.

Приостановка работы домена хранения

При приостановке работы домена хранения дополнительная емкость хранилища не используется.

Создание снимка виртуальной машины

Создание снимка виртуальной машины может повлиять на емкость домена хранения.

- При создании снимка на лету по умолчанию используются снимки памяти и создаются два дополнительных тома для каждой виртуальной машины. Первый том - это сумма памяти, видеопамяти и 200-мегабайтного буфера. Второй том содержит конфигурацию виртуальной машины размером в несколько мегабайт. При использовании блочного хранилища округление происходит до ближайшей единицы хранения (unit), которую может обеспечить zVirt.
- При создании снимка оффлайн он изначально занимает 1 ГБ блочного хранилища и динамически расширяется вплоть до размера диска.
- Клонирование снимка создает новый диск того же размера, что и исходный.
- Фиксация снимка удаляет все дочерние тома в зависимости от того, где в цепочке происходит фиксация.
- Удаление снимка в конечном итоге приводит к удалению дочернего тома для каждого диска и поддерживается только при работающей виртуальной машине.
- При предварительном просмотре снимка создается временный том для каждого диска, чтобы обеспечить достаточно места для предварительного просмотра.
- Отмена предварительного просмотра снимка удаляет временный том, созданный инструментом предварительного просмотра.

Подключение и удаление прямых LUN

Подключение и удаление прямых LUN не влияет на домен хранения, поскольку они не являются компонентом домена хранения. Дополнительные сведения см. в разделе Общие сведения о миграции между хранилищами "на лету" в Руководстве по администрированию.

Технический справочник. Системные учетные записи

1. Учетные записи пользователей Менеджера управления

При установке пакетов **ovirt-engine**, **ovirt-vmconsole** создается ряд учетных записей системных пользователей для поддержки zVirt. Каждый системный пользователь имеет идентификатор пользователя, присваиваемый по умолчанию (`UID`). Создаются следующие учетные записи системных пользователей:

- Пользователь **vdsm** (`UID 36`). Требуется для инструментов поддержки, которые монтируют домены хранения NFS и обращаются к ним.
- Пользователь **ovirt** (`UID 108`). Владелец экземпляра Wildfly ovirt-engine.
- Пользователь **ovirt-vmconsole** (`UID 992`). Требуется для гостевой консоли, подключаемой по последовательному интерфейсу

2. Группы Менеджера управления

При установке пакетов **ovirt-engine**, **ovirt-vmconsole** создается ряд групп системных пользователей для поддержки zVirt. Каждая группа системных пользователей имеет идентификатор группы, присваиваемый по умолчанию (`GID`). Создаются следующие группы системных пользователей:

- Группа **kvm** (`GID 36`). Члены группы:
  - Пользователь **vdsm**.
- Группа **ovirt** (`GID 108`). Члены группы:
  - Пользователь **ovirt**.
- Группа **ovirt-vmconsole** (`GID 998`). Члены группы:
  - Пользователь **ovirt-vmconsole**.

3. Учетные записи хоста виртуализации

При установке пакетов **vdsm**, **sanlock**, **ovirt-vmconsole** и **qemu-kvm** на хосте виртуализации создается ряд учетных записей системных пользователей. Каждый системный пользователь

имеет идентификатор пользователя, присваиваемый по умолчанию (`UID`). Создаются следующие учетные записи системных пользователей:

- Пользователь **vdsm** (`UID 36`).
- Пользователь **qemu** (`UID 107`).
- Пользователь **sanlock** (`UID 179`).
- Пользователь **ovirt-vmconsole** (`UID 998`).

Присваиваемые идентификаторы пользователей (UID) и идентификаторы групп (GID) могут варьироваться от системы к системе. За пользователем vdsm зарезервирован UID 36, а за группой kvm зарезервирован GID 36.

Если UID 36 или GID 36 уже используется другой учетной записью в системе, то при установке пакетов vdsm и qemu-kvm возникнет конфликт.

4. Группы хостов виртуализации

При установке пакетов **ovirt-vmconsole** и **qemu-kvm** на хосте виртуализации создается ряд групп системных пользователей. Каждая группа системных пользователей имеет идентификатор группы, присваиваемый по умолчанию (GID). Создаются следующие группы системных пользователей:

- Группа **kvm** (`GID 36`). Члены группы:
  - Пользователь **qemu**.
  - Пользователь **sanlock**.
- Группа **qemu** (`GID 107`). Члены группы:
  - Пользователь **vdsm**.
  - Пользователь sanlock.
- Группа **ovirt-vmconsole** (`GID 998`). Члены группы:
  - Пользователь **ovirt-vmconsole**.

Присваиваемые идентификаторы пользователей (UID) и идентификаторы групп (GID) могут варьироваться от системы к системе. За пользователем vdsm зарезервирован UID 36, а за группой kvm зарезервирован GID 36.

Если UID 36 или GID 36 уже используется другой учетной записью в системе, то при установке пакетов vdsm и qemu-kvm-rhev возникнет конфликт.

5. Специальные пользователи ovirt-administrator и everyone

5.1. ovirt-administrator

Пользователь *ovirt-administrator* в виртуализации zVirt является специальным высокопривилегированным пользователем с обширными административными правами.

Пользователь *ovirt-administrator* задается при установке zVirt ииспользуется менеджером для предоставления расширенных прав и полномочий администраторам системы, необходимых для управления и настройки системных разрешений zVirt. Для пользователя *ovirt-administrator* системой устанавливаются права роли **SuperUser**.

Пользователь *ovirt-administrator* имеет следующие права и роли:

1. Право на чтение и запись информации о виртуальных машинах,включая их статус, имя, описание и т.д.
2. Право на чтение и запись информации о хранилищах данных,включая их имя, тип, состояние и т.д.
3. Право на чтение и запись информации о сетях, включая их имя,описание, состояние и т.д.
4. Право на чтение и запись информации о профиляхвиртуальных машин, включая их имя, описание и т.д.
5. Право на чтение и запись информации о пользователях,включая их имя, описание и т.д.

5.2. everyone

Пользователь *everyone* - специальный идентификатор, который означает всех аутентифицированных пользователей.

Пользователь *everyone* задается по умолчанию при установке zVirt и используется для предоставления минимальных прав и полномочий всем пользователям системы, необходимых для управления и настройки системных разрешений zVirt. Пользователь *everyone* в zVirt представляет собой системную роль, которая по умолчанию предоставляется всем пользователям.

Полномочия пользователя *everyone* ограничены и не позволяютему выполнять какие-либо действия, которые могут повлиять наработу системы или других пользователей. Для пользователя *everyone* системой устанавливаются права роли **UserProfileEditor**.

Пользователь *everyone* имеет минимальные права и полномочия:

1. Право на чтение информации о виртуальных машинах,включая их статус, имя, описание и т.д.
2. Право на чтение информации о хранилищах данных, включаяих имя, тип, состояние и т.д.
3. Право на чтение информации о сетях, включая их имя,описание, состояние и т.д.
4. Право на чтение информации о профилях виртуальных машин,включая их имя, описание и т.д.
5. Право на чтение информации о пользователях, включая ихимя, описание и т.д.

Технический справочник. Шаблоны и пулы

1. Шаблоны и пулы

Среда zVirt предлагает администраторам инструменты, упрощающие предоставление виртуальных машин пользователям, а именно шаблоны и пулы. **Шаблон** - это ярлык, который позволяет администратору быстро создать новую виртуальную машину на основе существующей и уже настроенной виртуальной машины, минуя установку и настройку операционной системы. Это особенно полезно для виртуальных машин, которые будут использоваться как прикладное решение (например, для виртуальных веб-серверов). Если в организации используется много экземпляров конкретного веб-сервера, администратор может создать виртуальную машину для использования в качестве шаблона, установив операционную систему, веб-сервер, любые вспомогательные пакеты и применив уникальные изменения конфигурации. Затем администратор может создать на основе работающей виртуальной машины шаблон, который будет использоваться для создания новых идентичных виртуальных машин по мере необходимости.

Пулы виртуальных машин - это группы виртуальных машин, которые основаны на заданном шаблоне и которые можно быстро предоставить пользователям. Разрешение на использование виртуальных машин в пуле дается на уровне пула: пользователю, которому дано разрешение на использование пула, будет назначена любая виртуальная машина из пула. В силу особенностей работы пула данные после сессии конкретного пользователя не сохраняются в ВМ. Поскольку виртуальные машины назначаются пользователям без учета того, какую виртуальную машину в пуле они использовали в прошлом, пулы не подходят для задач, требующих сохраняемости данных. Пулы виртуальных машин лучше всего подходят для задач, когда пользовательские данные хранятся централизованно, а для доступа к ним и их использования нужна виртуальная машина, или когда сохраняемость данных не важна. Пул создается вместе с виртуальными машинами, которые в него входят и после создания остаются в остановленном состоянии. Затем по запросу пользователя они запускаются.

2. Шаблоны

Чтобы создать шаблон, администратор создает виртуальную машину и выполняет ее индивидуальную настройку. Устанавливаются нужные пакеты, применяются индивидуальные настройки, и виртуальная машина подготавливается к целевому использованию, чтобы минимизировать изменения, которые придется вносить в нее после развертывания. Необязательным (но рекомендуемым) шагом перед созданием шаблона из

виртуальной машины является генерализация. Генерализация призвана удалить такие сведения, как имена пользователей системы, пароли и информацию о часовом поясе, которые изменятся при развертывании. Генерализация не влияет на индивидуальные настройки. Генерализация гостевых машин Windows и Linux в среде zVirt обсуждается в статье Шаблоны в Руководстве пользователя. Для генерализации гостевых машин Linux используется `virt-sysprep`. Для генерализации гостевых машин Windows используется `sys-prep`.

Когда виртуальная машина, являющаяся основой для шаблона, удовлетворительно настроена, генерализирована (если нужно) и остановлена, администратор может создать из нее шаблон. При создании шаблона из виртуальной машины создается доступная только для чтения копия специально настроенного виртуального диска. Образ, доступный только для чтения, выступает в качестве базового для всех создаваемых впоследствии виртуальных машин, основанных на этом шаблоне. Иными словами, шаблон - это по сути индивидуально настроенный виртуальный диск, доступный только для чтения, с соответствующей конфигурацией виртуального оборудования. Оборудование виртуальных машин, созданных из шаблона, можно менять (например, выделить 2 ГБ ОЗУ виртуальной машине, созданной из шаблона с 1 ГБ ОЗУ). Однако виртуальный диск шаблона нельзя изменить, так как это приведет к изменению всех виртуальных машин на основе этого шаблона.

После создания шаблона его можно использовать как основу для множества виртуальных машин. При создании виртуальной машины из заданного шаблона можно выбрать тип выделяемого диска - **тонкий** (**thin**) или **клонированный** (**clone**). Виртуальные машины, клонируемые из шаблонов, используют полную копию базового образа шаблона, допускающую запись. Хотя они и не экономят место (в отличие от динамического выделения), зато не зависят от наличия шаблона. Виртуальные машины, создаваемые из шаблона способом динамического выделения, используют образ из шаблона, доступный только для чтения, в качестве базового образа. При этом требуется, чтобы шаблон и все созданные на его основе виртуальные машины хранились в одном домене хранения. Изменения данных и вновь сгенерированные данные сохраняются в образе с возможностью копирования при записи. Каждая виртуальная машина на основе шаблона использует один и тот же базовый образ, доступный только для чтения, а также уникальный для виртуальной машины образ с возможностью копирования при записи. Это экономит пространство за счет ограничения числа копий идентичных данных, хранящихся в хранилище. Кроме того, при частом использовании базового образа, доступного только для чтения, данные, к которым производится обращение, могут начать кэшироваться, что выльется в чистый рост производительности.

3. Пулы

Пулы виртуальных машин позволяют быстро предоставлять пользователям множество одинаковых виртуальных машин в качестве рабочих станций. Пользователи, у которых есть разрешение на доступ к виртуальным машинам из пула и их использование, получают доступную виртуальную машину, исходя из их положения в очереди запросов. Виртуальные машины в пуле не допускают сохранения данных. При каждом назначении виртуальной машины из пула она выделяется в своем базовом состоянии. Это идеально подходит для ситуаций, когда пользовательские данные хранятся централизованно.

Пулы виртуальных машин создаются из шаблона. Каждая виртуальная машина в пуле использует один и тот же базовый образ, доступный только для чтения, и временный образ, допускающий копирование при записи, для хранения измененных и вновь сгенерированных данных. Виртуальные машины в пуле отличаются от других виртуальных машин тем, что слой копирования при записи, который содержит сгенерированные пользователем и измененные данные, теряется при выключении виртуальной машины. Из этого следует, что для хранения пула виртуальных машин нужно столько же места, что и для его базового шаблона, плюс некоторое пространство для данных, сгенерированных или измененных за время использования. Пулы виртуальных машин - это эффективный способ предоставления вычислительной мощности пользователям для решения ряда задач без затрат на хранение, связанных с предоставлением каждому пользователю выделенной виртуальной рабочей станции.

Пример 1. Пример использования пула

Компания, оказывающая услуги технической поддержки, держит в штате 10 специалистов. Однако в любой заданный момент времени работают только 5 из них. Вместо того, чтобы создавать 10 виртуальных машин (по одной на каждого сотрудника службы поддержки), можно создать пул из 5 виртуальных машин. Сотрудники службы поддержки выделяют себе виртуальную машину в начале смены и возвращают ее в пул в конце смены.

Начало работы

Terraform-провайдера zVirt используется для взаимодействия с компонентами платформы виртуализации zVirt начиная с версии 4.3.

Подробная информацию о ресурсах провайдера представлена на вкладке Ресурсы TF.

Рекомендуется использовать версию Terraform не ниже 1.5.

1. Особенности

- Опции инициализации не имеют реализованных средств глубокой валидации.
- Параметр `state` отрабатывается всеми ресурсами индивидуально. Обилие изменений, требующих перезагрузки, способно привести к многократному перезапуску виртуальной машины.
- Миграция ВМ осуществляется через перезапуск ВМ с выбором целевого кластера.
- Механизмы инициализации срабатывают только для одного сетевого интерфейса. Интерфейс определяется по имени внутри гостевой ОС и должен быть известен заранее.
- Получение IP-адреса реализовано через источник данных(data source) **ovirt_wait_for_ip** и может занимать значительное время. В случаи использования одного сетевого интерфейса рекомендуется использовать механизм инициализации(cloud-init).

2. Схема

```
terraform {
  required_version = ">= 1.5.0"
  required_providers {
    ovirt = {
      source  = "ovirt/ovirt"
      version = ">= 2.1.5"
    }
  }
}

provider "ovirt" {
  url          = "${var.zvirt_url}/api"
  username     = var.zvirt_username
  password     = var.zvirt_password
```

```
  tls_insecure = true
}
```

Обязательные параметры (или аргументы):

1. `url` (String) - URL-адрес для подключения к API.

   Например: https://example.com/ovirt-engine/api/

2. `username` (String) - имя пользователя и пространство имён (реалм) в zVirt.

   Например: admin@zvirt@internalsso

3. `password` (String,Sensitive) - пароль пользователя в zVirt.
4. `tls_insecure` (Boolean) - отключает проверку сертификата при подключении к zVirt.

Дополнительные параметры (или аргументы):

Параметры `tls_ca_bundle`, `tls_ca_dirs`, `tls_ca_files` взаимозаменяемы.

1. `tls_ca_bundle` (String) - активирует проверку сертификата Engine по предоставленным сертификатам CA. Передаваемая цепочка сертификатов должна быть в формате PEM.

   Например: `tls_ca_bundle = var.ovirt_ca`

2. `tls_ca_dirs` (List of String) - активирует проверку сертификата Engine на соответствие сертификатам CA, предоставленным в указанных каталогах. Каталог должен содержать только файлы с сертификатами в формате PEM.

   Например: `tls_ca_dirs = ["/etc/pki/"]`

3. `tls_ca_files` (List of String) - активирует проверку сертификата Engine на соответствие сертификатам CA, предоставленным в файлах, указанных в этом параметре. Файлы должны содержать сертификаты в формате PEM.

   Например: `tls_ca_files = ["/etc/pki/root_ca.pem", "/etc/pki/intermediate_ca.pem"]`

3. Дополнительная подготовка

Для работы с провайдером необходимо использовать учетную запись **admin@zvirt**.

4. Установка и запуск Linux

Поддерживаются актуальные дистрибутивы семейства RedHat и Ubuntu\ Debian и аналоги.

1. Скачайте дистрибутив Terraform и установите его согласно инструкции для используемой ОС.
2. В домашней директории пользователя, который будет использовать Terraform, создайте следующий каталог и все промежуточные подкаталоги:
   a. **~/.terraform.d/plugins/registry.terraform.io/ovirt/ovirt/2.1.5/linux_amd64/**
3. Скопируйте полученный бинарный файл провайдера в созданный каталог (linux_amd64) под именем **terraform-provider-ovirt_2.1.5**
4. Создайте каталог для проверки провайдера. В нем создайте Terraform-манифест следующего содержания:

```
terraform {
  required_version = ">= 1.5.0"
  required_providers {
    ovirt = {
      source  = "ovirt/ovirt"
      version = ">= 2.1.5"
    }
  }
}
```

5. Выполните команду `terraform init` в тестовом каталоге. Игнорируйте сообщения о неподписанном пакете.

Источники данных Terraform

Источники данных позволяют Terraform использовать информацию, определенную вне Terraform, определенную другой отдельной конфигурацией Terraform или измененную функциями.

▶ ovirt_cluster

▶ ovirt_cluster_hosts

▶ ovirt_vnic_profile

▶ ovirt_storage_domain

▶ ovirt_templates

Ресурсы TF

▶ ovirt_znic_attachment

▶ ovirt_zdisk_attachment

▶ ovirt_zdisk

▶ ovirt_disk_attachments

▶ ovirt_zvm