Режимы кеширования виртуальных дисков KVM в Proxmox VE

Записки IT специалиста

Технический блог специалистов ООО"Интерфейс"

- Главная
- Режимы кеширования виртуальных дисков KVM в Proxmox VE

Производительность дисковой подсистемы очень часто является узким местом многих вычислительных систем, и каждый системный администратор сталкивался и сталкивается с необходимостью оптимизации процессов обмена с накопителями. Одним из эффективных решений, позволяющих ускорить операции ввода - вывода является кеширование. Proxmox предлагает для виртуальных дисков KVM несколько режимов кеширования, в данной статье мы рассмотрим, как они работают, а также их достоинства и недостатки.

Прежде чем говорить о режимах кеширования, сделаем небольшое отступление и разберем в общих чертах как устроен процесс взаимодействия с накопителями в ОС Linux, так как именно на ее основе работает KVM-виртуализация.

Любые дисковые операции с физическим носителем достаточно дороги, вне зависимости от того, какой именно тип носителя перед нами, при этом операции случайного доступа всегда медленнее линейных, в ряде случаев медленнее на порядок и более. Что для старенького жесткого диска, что для современного NVMe такая тенденция будет сохраняться, отличаться будут только лишь абсолютные цифры.

При этом операции записи всегда дороже операций чтения. На это влияет множество факторов, но именно операции записи часто становятся узким горлышком системы. Почему так? Потому что данные на чтение мы можем эффективно и многократно кешировать на различных уровнях, обращаясь к

физическому носителю только тогда, когда данные в кеше отличаются от данных на диске. А вот при записи не все так просто. Система должна убедиться, что данные действительно записаны на накопитель и только потом процесс сможет работать дальше. Иначе в случае аварийного отключения питания все данные, которые не были физически записаны на накопитель будут потеряны.

Но процесс записи - дорогой процесс, поэтому по мере развития журналируемых файловых систем стало возможно многоуровневое кеширование записи. Система принимает данные в кеш и ставит их в очередь для физической записи на носитель, но процессу, инициировавшему операцию записи, выдается подтверждение, он считает, что данные физически записаны и продолжает работать дальше, не дожидаясь реальной записи данных.

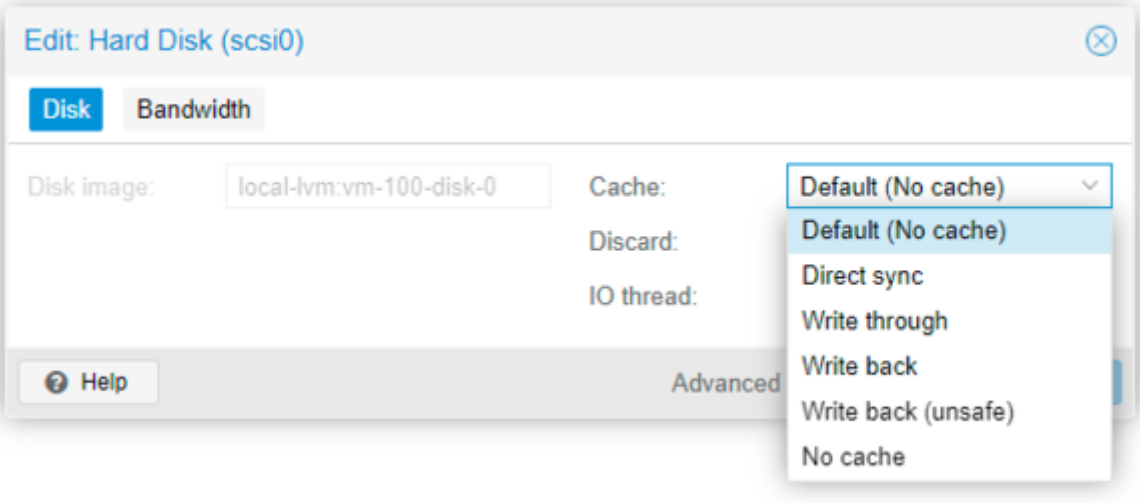
Чем этом может быть чревато? В случае аварийного отключения питания или иных сбоев (kernel panic и т.д.) вы потеряете часть информации, которая физически не была записана на накопитель. Но сама файловая система или база данных останутся в подавляющем большинстве случаев в исправном и непротиворечивом состоянии. Это достигается за счет журналирования и транзакционного контроля. При этом существует ряд критических операций, которые должны быть обязательно зафиксированы на физическом носителе, для этого предусмотрен системный вызов **fsync**, который принудительно сбрасывает **страничный кеш** на диск.

Здесь мы подходим к первому уровню кеширования на уровне операционной системы, страничный кеш является частью виртуальной памяти Linux, обладает высокой скоростью работы и эффективностью, но имеет самую низкую надежность. В случае любого сбоя содержимое страничного кеша будет потеряно.

Следующий уровень - это кеш системы хранения, он может иметь различное исполнение, начиная от кеша самого накопителя и заканчивая кешами RAID-контроллеров или программных реализаций (например, ZFS). Но в рассматриваемом нами контексте это не имеет значения, следует только понимать, что этим кешем управляет уже само хранилище и в нормальном режиме записи оно практически всегда пишет в кеш и подтверждает запись. А уже потом, самостоятельно сбрасывает содержимое кеша на физические носители. При использовании вызова **fsync** кеш хранилища также будет сброшен на диск.

Если коротко подвести итог, то следует понимать, что запись с **fsync** (синхронная) будет подтверждена тогда и только тогда, когда будет **физически записана** на устройство хранения. В остальных случаях данные попадут в тот или иной кеш. Но при этом синхронная запись самая медленная и к ней следует прибегать только для действительно важных данных.

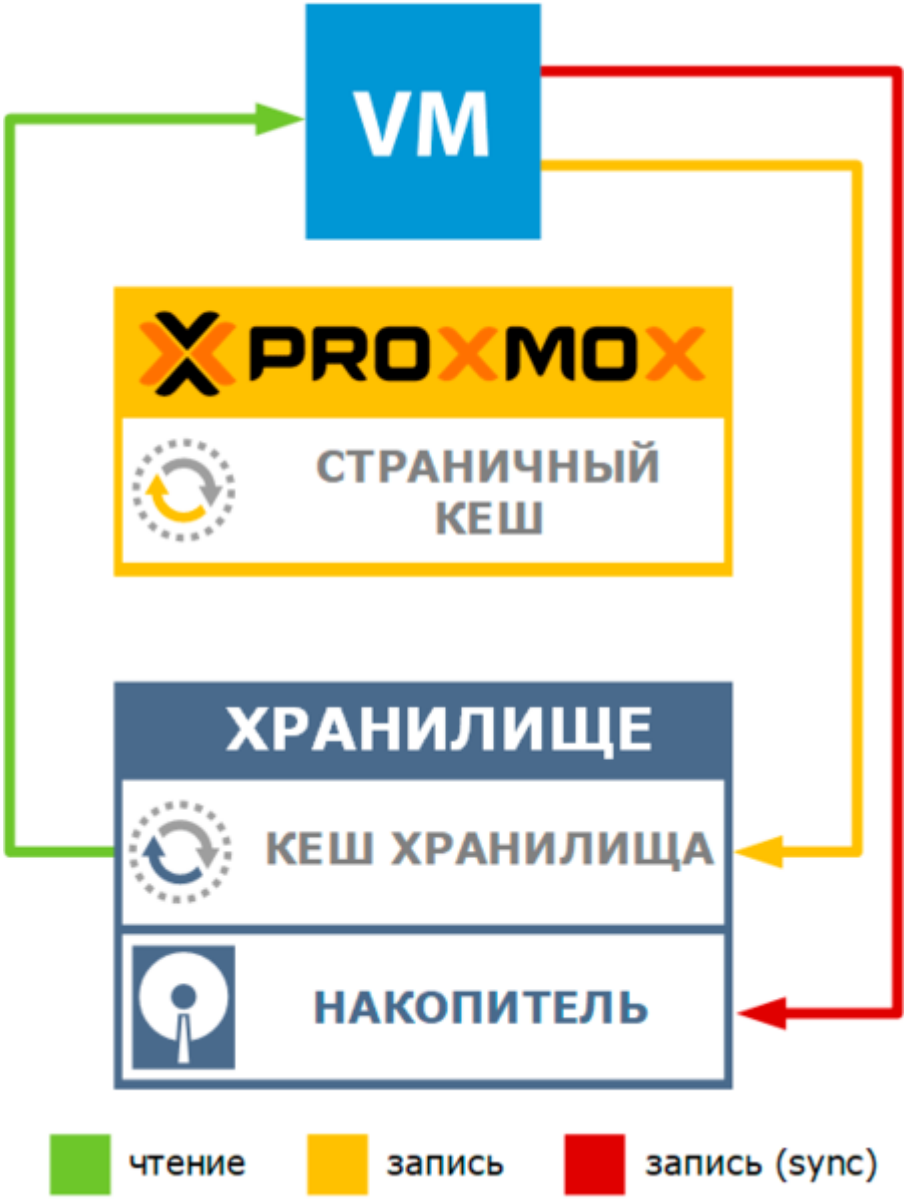
Теперь рассмотрим варианты кеширования для виртуальных дисков KVM, которые доступны в среде Proxmox VE. Ниже показан скриншот из английской версии интерфейса:



В русской версии выполнен перевод отдельных строк, но общее содержание меню практически не отличается от приведенного. Еще раз обращаем внимание, что данные настройки доступны только для виртуальных машин и неприменимы к контейнерам.

No cache

Используется по умолчанию и подразумевает то, что KVM никак не кеширует операции ввода - вывода и вообще никак не вмешивается в них.



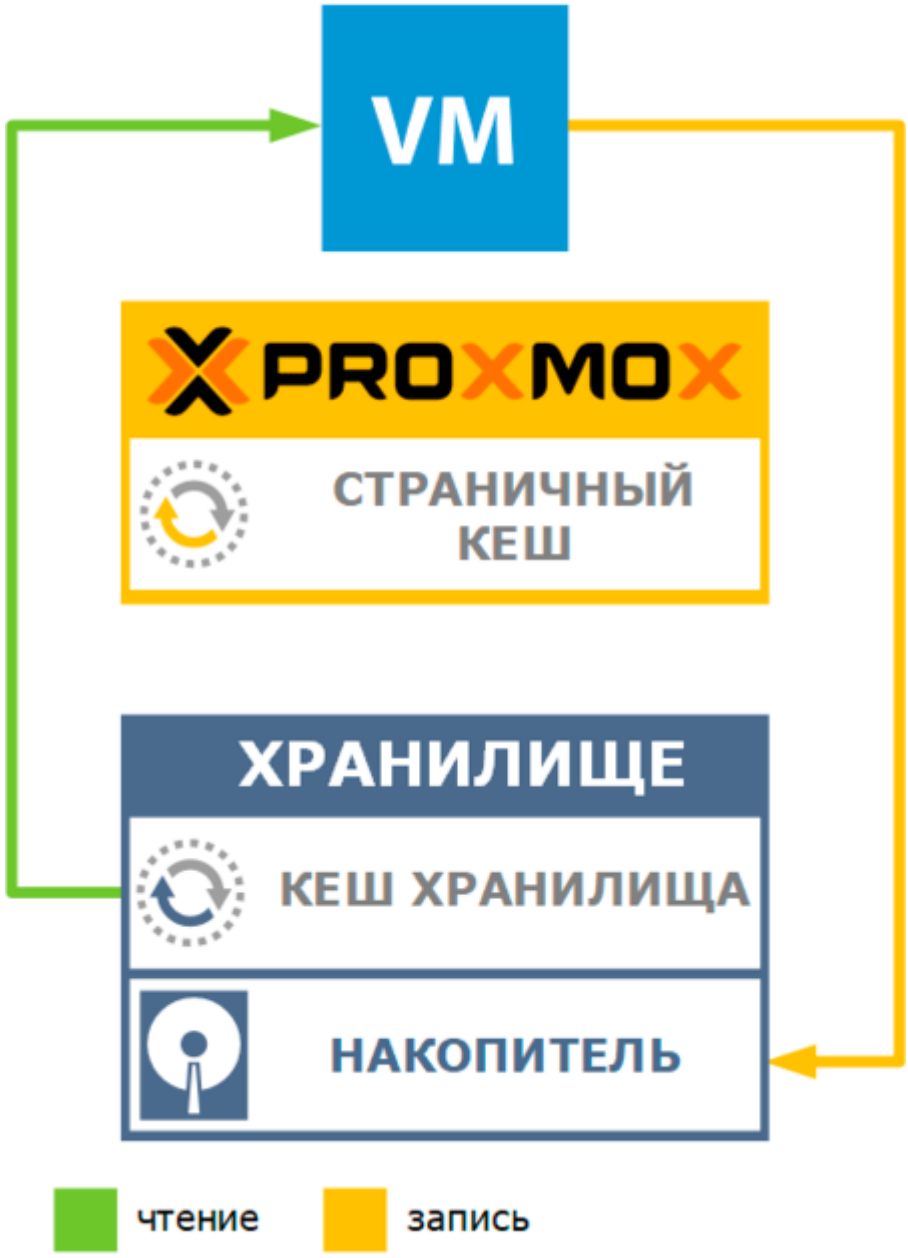
По сути, это эквивалентно прямому доступу виртуальной машины к хранилищу, при этом процесс записи управляется самим хранилищем, т.е. записываемые данные могут быть помещены в кеш. Виртуальный контроллер устройства хранения осведомлен о наличии кеша обратной записи у хранилища, и виртуальная машина будет использовать **fsync** для фиксации важных данных. Чтение также производится непосредственно из хранилища. Производительность ввода - вывода полностью зависит от производительности хранилища.

Кеш гостевого диска (внутри виртуальной машины) настроен на обратную запись (writeback), т.е. внутри гостевой системы будет активно использоваться кеширование и для сброса внутреннего кеша также должен использоваться вызов fsync.

На наш взгляд использование этой настройки по умолчанию полностью оправдано. Гипервизор никак не влияет на дисковые операции и их скорость и надежность полностью зависят от хранилища, что в большинстве случаев и ожидается пользователями.

Direct sync

Прямая синхронизация - это самый медленный, но самый безопасный режим. Каждая операция записи производится в синхронном режиме, что обеспечивает гарантированную запись данных, чтение также производится непосредственно из хранилища.



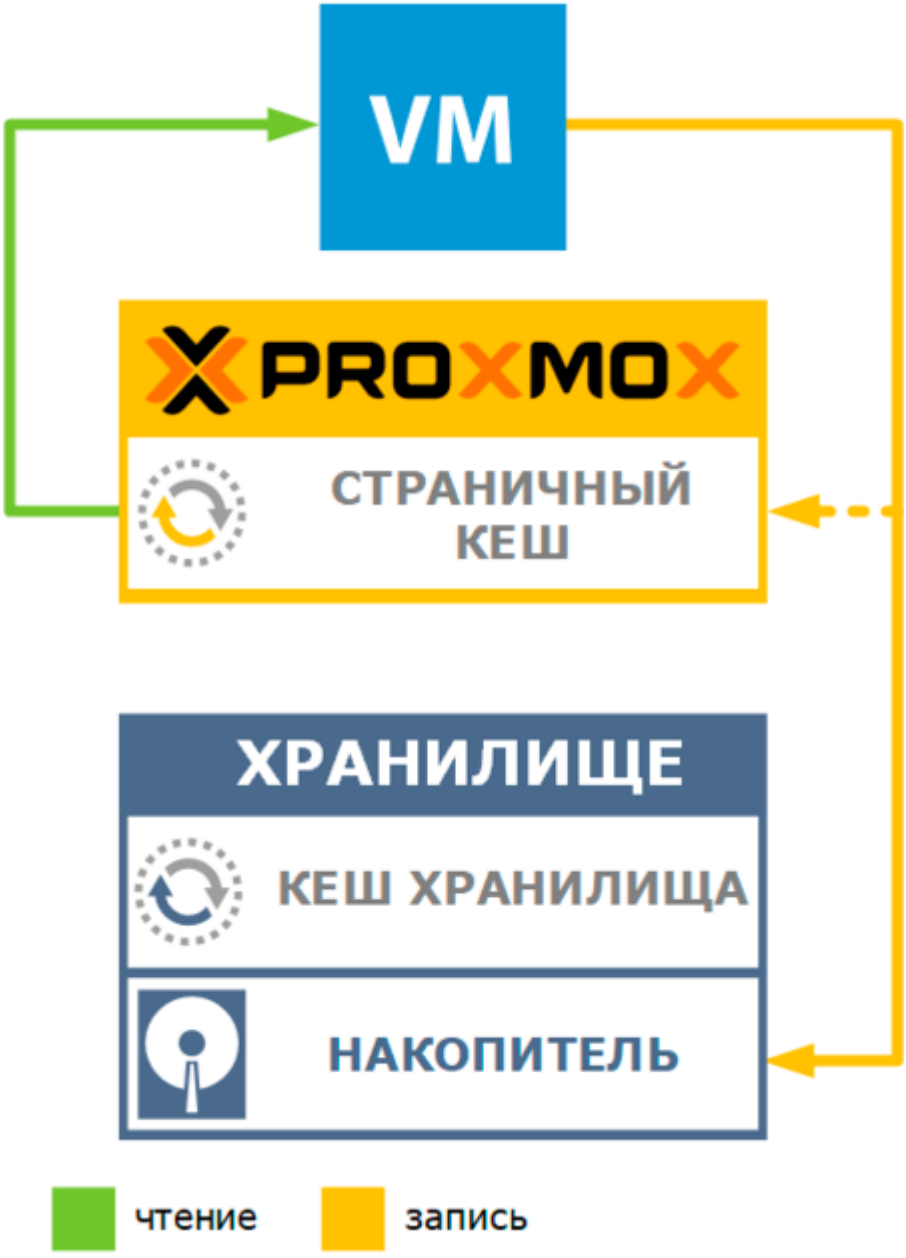
Внутри виртуальной машины используется режим сквозной записи (writethrough), что также аналогично использованию fsync для каждой записи. Поскольку вся запись производится в синхронном режиме необходимости использовать sync для гостевой системы нет.

В каком случае может пригодиться этот режим? Тогда, когда целостность данных имеет для вас решающее значение и вы готовы для этого пожертвовать производительностью записи. В качестве примера такой системы можно привести контроллер домена, для которого задачей первоначальной важности является целостность и непротиворечивость базы.

Не рекомендуется использовать совместно с дисками формата qcow2, это может привести к серьезному снижению производительности.

Write Through

Сквозная запись, режим чем-то похож на прямую синхронизацию, но имеет некоторые особенности. В частности, записываемые данные дополнительно кешируются на уровне гипервизора используя страничный кеш.



Запись также ведется в синхронном режиме, в т.ч. и внутри виртуальной машины, необходимости дополнительно использовать fsync для гостевой системы нет. Для операций чтения в первую очередь используется страничный кеш, что позволяет значительно повысить производительность чтения, при условии, конечно, что данные есть в кеше. Если же их там нет, что чтение будет произведено из хранилища.

Данный режим по надежности не уступает прямой синхронизации, но позволяет ускорить операции чтения за счет кеширования на уровне гипервизора и уменьшить обращение к хранилищу, что косвенно окажет положительное влияние на работу дискового ввода-вывода в целом.

Так что именно использовать: прямую синхронизацию или сквозную запись? Смотрите на реальные сценарии, если запись достаточно редкая и записанные данные используются преимущественно для чтения, то режим сквозной записи будет предпочтителен, если же у вас идет интенсивная запись и не связанное с ней интенсивное чтение, то вы не сможете нормально прогреть страничный кеш и только загрузите его (а с ним и ценную оперативную память) бесполезным мусором.

Также, как и прямую синхронизацию данный вид кеширования не следует использовать с qcow2 виртуальными дисками по причине сильного снижения производительности.

Write Back

До этого мы, хоть и говорили о кешировании, по сути, занимались обратным - обеспечением синхронной записи на диск. И вот теперь мы будем говорить именно о кеше. Обратная запись - это режим кеширования по максимуму использующий страничный кеш гипервизора, это позволяет значительно укорить как операции чтения, так и записи. Но при этом следует помнить, что страничный кеш будет сбрасываться в хранилище в обычном режиме, т.е. с большой вероятностью попадет еще в кеш хранилища. При том, что запись будет подтверждена уже при помещении данных в страничный кеш.

Это не очень безопасно, так как при аварийном отключении питания шансы потерять данные значительно вырастают. Внутри виртуальной машины также используется кеширование с обратной записью. Контроллер виртуального диска осведомлен о наличии обратного кеша и будет использовать вызовы fsync для фиксации данных.

Для каких сценариев можно выбирать такой метод кеширования? В случаях, когда прежде всего важна скорость работы с данными, а их целостность не представляет особой ценности. Какие это могут быть системы? Скажем, кеширующий веб-сервер, при условии, что база данных расположена на отдельном узле. Максимум, что мы потеряем при аварии - это кеш, зато сможем быстро и без просадок обслуживать клиентские запросы. При этом, в случае необходимости зафиксировать важные данные система всегда может вызвать sync и осуществить синхронную запись.

Write Back (unsafe)

Кеширование с обратной записью и дополнительной пометкой **не безопасно** - что это такое? Это все та же обратная запись, но с одной тонкостью, причем тонкостью очень существенной. Давайте внимательно посмотрим на схему ниже.

Подождите, скажет внимательный читатель, ведь в начале статьи было написано, что вызов fsync гарантирует то, что запись будет подтверждена только после физической записи на устройство хранения. Но здесь нет никакого противоречия, в **небезопасном** режиме обратной записи гипервизор **игнорирует** вызовы fsync гостевой системы.

К чему это может привести? Да к самым тяжелым последствиям, потому как именно fsync является залогом того, что критически важные данные, например, метаданные файловой системы будут действительно записаны на устройство и не будут потеряны при сбое. В нашем случае это не гарантируется и при аварийном отключении вы можете не только потерять данные, которые не успели записать, но и получить **разрушение файловой системы** или иных структур данных (СУБД и т.п.).

Данный вид кеширования однозначно не следует использовать в производственных средах, да и вообще не следует использовать, кроме исключительных случаев, когда вам нужна максимальная производительность ввода - вывода, а на данные вам плевать. Ну, например, в тестовых средах или для разовой обработки какого-то объема данных, которые не находятся в единственном экземпляре.

Заключение

Статья получилась относительно небольшая, мы сознательно не стали углубляться в тонкости процесса, чтобы сделать ее максимально простой и доступной для начинающих. Надеемся, что после ее прочтения вопрос выбора режима кеширования перестанет быть для вас чем-либо непонятным и вы сможете подойти к нему осознанно и с учетом выполняемых виртуальной машиной задач.

- **Категории:**
- Виртуализация,
- Системному администратору
- **Теги:**
- Proxmox,
- Виртуализация,
- Производительность