How to create and use additional volumes for VMs?

qubist ⋮ 17-22 minutes ⋮ 1/22/2024

January 22, 2024, 12:55pm 1

Hi,

I am looking for a way to:

- create a volume in some pool (e.g. pool00 or other)
- mount it in read-write mode to one VM
- write some data to it
- unmount it
- attach it to one or more VMs (simultaneously) as read-only

How can I do this?

balko January 22, 2024, 6:01pm 2

Consider an alternative option to just use 3rd qube with a file of desired size and `losetup` this file to make a connectable device with one command. This way you will not affect LVM in any way. If it fits your scenarios.

qubist January 22, 2024, 7:31pm 3

I have never used losetup. Could you please explain how you do this?

balko January 23, 2024, 5:24am 4

What you should do for using my approach (not using LVM pool directly):

1. Create a new simple template-based qube, e.g. fedora or debian based (versions do not matter). Let's name it `storageqube`.
2. Increase its private storage size by the volume you want, so it will be have enough space. E.g. if we want 10GiB disk, set private storage size to 11 GiB.
3. Run in `mystorage` qube:

```
dd if=/dev/zero of='/home/user/mystorage.disk' bs=1M count=10240
```

This will create a file `mystorage.disk` with size 10GiB. You can change the file location and size.

4. Run in `mystorage` qube:

```
sudo losetup --show '/dev/loop4' '/home/user/mystorage.disk'
```

It will create a new device that will be visible in Qubes OS and possible to attach to any other qube (even with `Qubes Devices` GUI applet in the tray!).

Note: Number four in `loop4` is just an example, you can use any number (0-63), or use automatic unoccupied search using `--find` argument. But for scripts it is useful to have it fixed and not default `loop0`.

5. Attach it to any qube you like using GUI tools or in `dom0` terminal:

```
qvm-block attach 'qube1' 'mystorage:loop4'
```

5. Format it to any filesystem (in case of ext4 you can free extra 5% of space with `sudo tune2fs -m 0` command).

If you want to connect it to multiple qubes, just unmount, detach and attach again and mount as you would do with a real flash drive in Qubes OS.

This approach also works for Windows, you can format the disk to NTFS and use without any issues.

If this post will draw interest, I will make a tutorial in the corresponding forum section.

solene January 23, 2024, 9:28am 5

Thanks for sharing this, I guess this would be a nice “community guides” entry 🙂

In addition, it’s possible to encrypt the volume using luks or veracrypt as it’s a raw volume we can do what we want with it 👍

qubist January 23, 2024, 10:33am 6

Thanks for explaining.

IIUC (please correct me if I am wrong), this approach has the following specifics:

- It requires pre-writing of dummy data before writing the actual one
- Access to such volume requires a running qube hosting it
- There is a limit of 64 volumes (per hosting qube)
- All volumes created this way will show up in Qubes Devices menu

Is there a way to avoid those specifics?

My XY goal is to create a minimalist non-duplicating modular system (a split template of sorts) in which individual packages reside on separate volumes (modules) that can be attached selectively to particular AppVMs (similar to the way we currently have the root.img of the template attached). IOW, I am looking for possible workarounds of this.

solene January 23, 2024, 10:52am 7

For this use case, I wonder if a simple solution like temporarily installing the packages you want upon the qube start in `/rw/config/rc.local` would not be suitable?

Using a local caching qube would prevent from downloading all the packages every time, so the installation would be really fast.

Debian or Arch Linux package managers would be way faster than Fedora IMO. Flatpak would be terribly slow, Nix would be slow due to untarring nixpkgs before actually doing anything (except if you pin it and have it ready from the template).

qubist January 23, 2024, 11:57am 8

I have considered that during my quest for efficiency but I decided it would be contrary to the goal (non-duplication, resource usage efficiency, management simplicity). It also requires the AppVM to have networking which is a problem for offline and firewall-restricted qubes.

So far, the best (or should I say “least bad”) approach (in regards to efficiency) which I have found is the decremental approach. I have optimized it even a little more than what is shown in the post I link to, but I have not published anything yet, because it is work in progress. So, meanwhile, I keep looking for a better way because I still have this security concern.

solene January 23, 2024, 11:58am 9

A proxy could be used to reach the cache. As the data wouldn’t be persistent it’s quite efficient (except for the CPU used for installing at each boot, but it’s pretty minimal).

qubist January 23, 2024, 12:28pm 10

With the method you suggest, there are also the additional SSD writes on each boot of the AppVM for doing the same thing over and over again. That is a duplicating approach + bad for hardware. I am trying to avoid that.

parulin January 23, 2024, 7:53pm 11

Maybe you can use `truncate -s 10G mystorage.disk` instead of dd ?

parulin January 23, 2024, 7:54pm 12

Just curious about this: why not create a second pool?

qubist January 23, 2024, 8:33pm 13

> Just curious about this: why not create a second pool?

Creating a pool is not a problem. The question is how to create a volume in it and how to use that volume after that. That is what I am trying to figure.

balko January 24, 2024, 6:09am 14

Yes. But not sure about 64 limit, many not, maybe I am confusing it with `veracrypt`.

I personally use scripts to automate all that, so I do not have to run hosting qube manually nor run any commands.

Yes, but only after you `losetup` it. Before that it is a regular file in the qube.

Also positive thing: you can backup this disk with Qubes OS backup systems as it is a regular file inside

the qube. The space overhead is tiny.

Because it can affect Qubes OS in theory. I **had 2 cases of Qubes OS LVM corruption in the past** (due to Qubes OS bug), so I prefer not to make it even less reliable as it is.
And backing up would be different, as I described, I can use `qvm-backup` the same way I do for other qubes or I can just compress manually a regular file inside the qube, it is nice. For huge disks I would use additional drive, but for small ones it is a nice approach.

qubist January 24, 2024, 11:46am 15

> Yes, but only after you `losetup` it. Before that it is a regular file in the qube.

The problem is that the volumes need to be available at any time to any qube which may start and need them, and the control mechanism for that becomes dependent on one or more domUs, not on dom0 only. Additionally, many volumes will “spam” the GUI for no reason (e.g. 100 volumes).

> I **had 2 cases of Qubes OS LVM corruption in the past** (due to Qubes OS bug), so I prefer not to make it even less reliable as it is.

But the qube(s) in which you suggest to create the loop devices are also in the same LVM where the qube itself resides. So, if the LVM gets corrupt, it affects it anyway. I don’t see how keeping the volume inside a qube will make it more reliable in that sense. Am I missing something?

Security-wise, it seems to me this whole approach may be problematic, because the volume resides in a read-write private.img, i.e. the volume security is handled by a domU, not by dom0.

It is also challenging in regards to the particular thing I am trying to do. Just imagine the following example:

- 100 packages total should be installed.
- 20 packages are the main ones, the rest are dependencies. (20 volumes minimum in case there are no common dependencies)
- Some packages *will* have common dependencies with other packages. To avoid duplication, a common dependency should be on its own volume, “pluggable” where necessary. The number of such volumes is difficult to calculate, but in any case, since the total number of packages is 100, the maximum number of volumes is also 100. For simplicity, let’s say we host 50 packages in qube-X and 50 in qube-Y. If there is a package in qube-X, depending on a package in qube-Y, which itself depends on another package in qube-X. An algorithm will be required to decide the most efficient placement of individual or common dependencies, and all that package management is actually accomplished through a template which accesses these 100 volumes - to my mind, this will be a management nightmare.

And that is a very simplified example. In reality many more packages may be installed, interrelated, requiring many qubes running at the same time, perhaps introducing performance overhead and bottlenecks, maybe even other things I can’t think of right now.

I suppose the losetup approach is suitable for simpler things.

balko January 25, 2024, 4:32am 16

The major difference is that in my approach the volume is just a regular file inside of VM. If power goes

down or anything bad happens, including getting out of space on this virtual disk, LVM will never be affected by that. Worse case scenario - you have corrupted filesystem inside this file. Alternative solution of using the existing Qubes LVM pool or creating another one on the same physical drive obviously has higher chances to ruin something within LVM.

I see no issue that the file is stored inside the regular qube, it is like having qube for USB flash-drive, always better that do it in dom0.
But OK, I just provided an easy way that I use, it obviously cannot fit everybody 🙂

qubist January 25, 2024, 10:07am 17

> The major difference is that in my approach the volume is just a regular file inside of VM. If power goes down or anything bad happens, including getting out of space on this virtual disk, LVM will never be affected by that. Worse case scenario - you have corrupted filesystem inside this file. Alternative solution of using the existing Qubes LVM pool or creating another one on the same physical drive obviously has higher chances to ruin something within LVM.

I have re-read this multiple times and I really don't understand.

The comparison is between:

(A) A read-write virtual file on a virtual filesystem, stored in a read-write volume, which is a file stored on the LVM

(B) Another volume (which is supposed to be read-only most of the time) - also a file stored on the same LVM

How exactly is (A) more immune to data corruption than (B)?

> I see no issue that the file is stored inside the regular qube, it is like having qube for USB flash-drive, always better that do it in dom0.

It seems to me this is quite different.

Volumes stored in dom0 are not something that one "does" in dom0 in the sense: dom0 does not process the data inside the volume. It simply organizes volumes as "passive data" which it handles to domUs for actual processing. That is quite different from USB drives which are active I/O devices. That's why we have sys-usb which is disposable and USB devices are generally distrusted. If volumes would be distrusted the same way, we would have e.g. sys-volume and the whole LVM storing the volumes would be distrusted (and isolated from dom0).

> But OK, I just provided an easy way that I use, it obviously cannot fit everybody 🙂

I thank you for this! Unfortunately, it seems unsuitable for my XY, so I keep looking.

balko January 25, 2024, 2:50pm 18

Because it has additional layer of isolation. You literally do not issue any LVM-related commands, only Qubes OS does and it do it in the same manner as it usually does it. In (B) you will run LVM-related commands directly in the way Qubes OS may not be aware (of cause, nothing should be broken, but it can and does sometimes).
Well, does not matter.

Loop devices can use sparse files:

```
$ truncate -s 100G foo.img
$ sudo losetup --find --show foo.img
/dev/loop23
```

*(**Edit:** Oh, @parulin had already pointed that out)*

You could also do this in dom0 if you have a dom0 filesystem with enough free space to actually write your data to the file later. The easiest way to get that would be to install Qubes OS with the Btrfs layout, or the so-called “Standard Partition” layout with XFS.

qubist March 7, 2024, 3:13pm 20

Thanks.
So, in short, what I am looking for seems not possible.

unman March 8, 2024, 12:32am 21

?

1. `sudo lvcreate -n storage -V 20G --thinpool vm-pool qubes_dom0`
2. `qvm-block` - shows DEVID - like `dom0:dm-200`
3. `qvm-block attach QUBE dom0:dm-200`
4. in QUBE, as root: `mkfs /dev/xvdi`, or whatever frontend is used
5. in QUBE, as root: `mount /dev/xvdi /mnt` ; do stuff in /mnt
6. in QUBE, as root: `umount /mnt`
7. `qvm-block detach QUBE dom0:dm-200`
8. `qvm-block attach --ro QUBE1 dom0:dm-200`
9. `qvm-block attach --ro QUBE2 dom0:dm-200`
10. `qvm-block attach --ro QUBE3 dom0:dm-200`

qubist March 8, 2024, 8:35am 22

Thanks @unman! I will definitely give this a try.

A question about step 8 and next: Is it necessary to detach from previous QUBE before attaching to next one, or can the same block be attached (as read-only) to multiple qubes in parallel, so they can read from it simultaneously?