Click to verify

From Wikipedia:Logical Volume Manager (Linux):Logical Volume Manager (LVM) is a device mapper framework that provides logical volume management for the Linux kernel.BackgroundLVM building blocksLogical Volume Management utilizes the kernel's device-mapper feature to provide a system of partitions independent of underlying disk layout. With LVM you abstract your storage and have "virtual partitions", making extending/shrinking easier (subject to potential filesystem limitations).Virtual partitions allow addition and removal without worry of whether you have enough contiguous space on a particular disk, getting caught up fdisking a disk in use (and wondering whether the kernel is using the old or new partition table), or, having to move other partitions out of the way.Basic building blocks of LVM:Physical volume (PV)Unix block device node, usable for storage by LVM. Examples: a hard disk, an MBR or GPT partition, a loopback file, a device mapper device (e.g. dm-crypt). It hosts an LVM header.Volume group (VG)Group of PVs that serves as a container for LVs. PEs are allocated from a VG for a LV.Logical volume (LV)"Virtual/logical partition" that resides in a VG and is composed of PEs. LVs are Unix block devices analogous to physical partitions, e.g. they can be directly formatted with a file system.Physical extent (PE)The smallest contiguous extent (default 4 MiB) in the PV that can be assigned to a LV. Think of PEs as parts of PVs that can be allocated to any LV.Example:Physical disks Disk1 (/dev/sda): Partition1 50 GiB (Physical volume) Partition2 80 GiB (Physical volume) /dev/sda1 /dev/sda2 Disk2 (/dev/sdb): Partition1 120 GiB (Physical volume) /dev/sdb1 LVM logical volumes Volume Group1 (/dev/MyVolGroup/ = /dev/sda1 + /dev/sda2 + /dev/sdb1): Logical volume1 15 GiB Logical volume2 35 GiB Logical volume3 200 GiB /dev/MyVolGroup/rootvol /dev/MyVolGroup/homevol /dev/MyVolGroup/mediavol Note: Logical volumes are accessible at both /dev/VolumeGroupName/LogicalVolumeName and /dev/mapper/VolumeGroupName-LogicalVolumeName. However, lvm(8) VALID NAMES recommends the former format for "software and scripts" (e.g. fstab) since the latter is intended for "internal use" and subject to possible "change between releases and distributions".AdvantagesLVM gives you more flexibility than just using normal hard drive partitions:Use any number of disks as one big disk.Have logical volumes stretched over several disks (RAID, mirroring, striping which offer advantages such as additional resilience and performance [1]).Create small logical volumes and resize them "dynamically" as they get filled up.Resize logical volumes regardless of their order on disk. It does not depend on the position of the LV within VG, there is no need to ensure surrounding available space.Resize/create/delete logical and physical volumes online. File systems on them still need to be resized, but some (such as Ext4 and Btrfs) support online resizing.Online/live migration of LV (or segments) being used by services to different disks without having to restart services.Snapshots allow you to backup a frozen copy of the file system, while keeping service downtime to a minimum and easily merge the snapshot into the original volume later.Support for unlocking separate volumes without having to enter a key multiple times on boot (make LVM on top of LUKS).Built-in support for caching of frequently used data (lvmcache(7)).DisadvantagesAdditional steps in setting up the system (may require changes to mkinitcpio configuration), more complicated. Requires (multiple) daemons to constantly run.If dual-booting, note that Windows does not support LVM; you will be unable to access any LVM partitions from Windows. 3rd Party software may allow to mount certain kinds of LVM setups. [2]If your physical volumes are not on a RAID-1, RAID-5 or RAID-6 losing one disk can lose one or more logical volumes if you span (or extend) your logical volumes across multiple non-redundant disks.It is not always easy to shrink the space used by the logical volume manager, meaning the physical volumes used for the logical volumes. If the physical extents are scattered across the physical volume until the end you might need to inspect the segments and move them (potentially to another physical device) or the same device with custom allocation settings (e.g. --alloc anywhere). If you want to dual-boot with other operating systems (e.g. with Microsoft Windows), the only space left on the device for Microsoft Windows is the space not used by LVM / not used as physical volume.Potentially worse performance than using plain partitions. [3]May not work well with all file systems, especially those that are designed to be (multi-)device aware. For example, Btrfs offers some of the same functionality (multi device support, (sub)volumes, snapshots and RAID) which could clash (read further about issues with LVM snapshots with Btrfs).InstallationMake sure the lvm2 package is installed.If you have LVM volumes not activated via the initramfs, enable lvm2-monitor.service, which is provided by the lvm2 package.Volume operationsPhysical volumesCreatingTo create a PV on /dev/sda1, run:# pvcreate /dev/sda1You can check the PV is created using the following command:# pvsGrowingAfter extending or prior to reducing the size of a device that has a physical volume on it, you need to grow or shrink the PV using pvresize(8).To expand the PV on /dev/sda1 after enlarging the partition, run:# pvresize /dev/sda1This will automatically detect the new size of the device and extend the PV to its maximum.Note: This command can be done while the volume is online.ShrinkingTo shrink a physical volume prior to reducing its underlying device, add the --setphysicalvolumesize size parameters to the command, e.g.:# pvresize --setphysicalvolumesize 40G /dev/sda1The above command may leave you with this error:/dev/sda1: cannot resize to 25599 extents as later ones are allocated.0 physical volume(s) resized / 1 physical volume(s) not resizedIndeed pvresize will refuse to shrink a PV if it has allocated extents after where its new end would be. One needs to run pvmove beforehand to relocate these elsewhere in the volume group if there is sufficient free space.Move physical extentsBefore freeing up physical extents at the end of the volume, one must run pvdisplay -v -m to see them. An alternative way to view segments in a tabular form is pvs --segments -v.In the below example, there is one physical volume on /dev/sdd1, one volume group vg1 and one logical volume backup.# pvdisplay -v -m Finding all volume groups. Using physical volume(s) on command line. --- Physical volume --- PV Name /dev/sdd1 VG Name vg1 PV Size 1.52 TiB / not usable 1.97 MiB Allocatable yes PE Size 4.00 MiB Total PE 399669 Free PE 153600 Allocated PE 246069 PV UUID MR9J0X-zQB4-wi3k-EnaV-5ksf-hN1P-Jkm5mW --- Physical Segments --- Physical extent 0 to 153600: FREE Physical extent 153601 to 307199: Logical volume/dev/vg1/backup Logical extents1 to 153599 Physical extent 307200 to 307200: FREE Physical extent 307201 to 399668: Logical volume/dev/vg1/backup Logical extents153601 to 246068One can observe FREE space are split across the volume. To shrink the physical volume, we must first move all used segments to the beginning.Here, the first free segment is from 0 to 153600 and leaves us with 153601 free extents. We can now move this segment number from the last physical extent to the first extent. The command will thus be:# pvmove --alloc anywhere /dev/sdd1:307201-399668 /dev/sdd1:0-92467/dev/sdd1: Moved: 0.1%/dev/sdd1: Moved: 0.2%.../dev/sdd1: Moved: 99.9%/dev/sdd1: Moved: 100.0%Note: This command moves 399668 - 307201 + 1 = 92468 PEs from the last segment to the first segment. This is possible as the first segment encloses 153600 free PEs, which can contain the 92467 - 0 + 1 = 92468 moved PEs.The --alloc anywhere option is used as we move PEs inside the same partition. In case of different partitions, the command would look something like this: # pvmove /dev/sdb1:1000-1999 /dev/sdc1:0-999This command may take a long time (one to two hours) in case of large volumes. It might be a good idea to run this command in a tmux or GNU Screen session. Any unwanted stop of the process could be fatal.Once the operation is complete, run fsck to make sure your file system is valid.Resize physical volumeOnce all your free physical segments are on the last physical extents, run vgdisplay with root privileges and see your free PE.Then you can now run again the command:# pvresize --setphysicalvolumesize size PhysicalVolumeSee the result:# pvs PV VG Fmt Attr PSize PFree /dev/sdd1 vg1 lvm2 a-- 1t 500gResize partitionLast, you need to shrink the partition with your favorite partitioning tool.Volume groupsCreating a volume groupTo create a VG MyVolGroup with an associated PV /dev/sdb1, run:# vgcreate MyVolGroup /dev/sdb1You can check the VG MyVolGroup is created using the following command:# vgsYou can bind multiple PVs when creating a VG like this:# vgcreate MyVolGroup /dev/sdb1 /dev/sdb2Activating a volume groupNote: You can restrict the volumes that are activated automatically by setting the auto_activation_volume_list in /etc/lvm/lvm.conf. If in doubt, leave this option commented out.# vgchange -a y MyVolGroupBy default, this will reactivate the volume group when applicable. For example, if you had a drive failure in a mirror and you swapped the drive; and ran (1) pvcreate, (2) vgextend and (3) vgreduce --removemissing --force.Repairing a volume groupTo start the rebuilding process of the degraded mirror array in this example, you would run:# lvconvert --repair /dev/MyVolGroup/mirrorYou can monitor the rebuilding process (Cpy%Sync Column output) with:# lvs -a -o +devicesDeactivating a volume groupJust invoke # vgchange -a n MyVolGroupThis will deactivate the volume group and allow you to unmount the container it is stored in.Renaming a volume groupUse the vgrename(8) command to rename an existing volume group.Either of the following commands renames the existing volume group MyVolGroup to my_volume_group# vgrename /dev/MyVolGroup /dev/my_volume_group# vgrename MyVolGroup my_volume_groupMake sure to update all configuration files (e.g. /etc/fstab or /etc/crypttab) that reference the renamed volume group.Add physical volume to a volume groupYou first create a new physical volume on the block device you wish to use, then extend your volume group# pvcreate /dev/sdb1# vgextend MyVolGroup /dev/sdb1This of course will increase the total number of physical extents on your volume group, which can be allocated by logical volumes as you see fit.Note: It is considered good form to have a partition table on your storage medium below LVM. Use the appropriate partition type: 8e for MBR, and E6D6D379-F507-44C2-A23C-238F2A3DF928 for GPT partitions (8e00 type code in gdisk, lvm type alias in fdisk).If you created a logical volume on the partition, remove it first.All of the data on that partition needs to be moved to another partition. Fortunately, LVM makes this easy:# pvmove /dev/sdb1If you want to have the data on a specific physical volume, specify that as the second argument to pvmove:# pvmove /dev/sdb1 /dev/sdf1Then the physical volume needs to be removed from the volume group:# vgreduce MyVolGroup /dev/sdb1Or remove all empty physical volumes:# vgreduce --all MyVolGroupFor example: if you have a bad disk in a group that cannot be found because it has been removed or failed:# vgreduce --removemissing --force MyVolGroupAnd lastly, if you want to use the partition for something else, and want to avoid LVM thinking that the partition is a physical volume:# pvremove /dev/sdb1Logical volumesNote: lvresize(8) provides more or less the same options as the specialized lvextend(8) and lvreduce(8) commands, while allowing to do both types of operation. Notwithstanding this, all those utilities offer a -r/--resizefs option which allows to resize the file system together with the LV using fsadm(8) (ext2, ext3, ext4, ReiserFS and XFS supported). Therefore it may be easier to simply use lvresize for both operations and use --resizefs to simplify things a bit, except if you have specific needs or want full control over the process.Warning: While enlarging a file system can often be done on-line (i.e. while it is mounted), even for the root partition, shrinking will nearly always require to first unmount the file system so as to prevent data loss. Make sure your file system supports what you are trying to do.Tip: If a logical volume will be formatted with ext4, leave at least 256 MiB free space in the volume group to allow using e2scrub(8). After creating the last volume with -l 100%FREE, this can be accomplished by reducing its size with lvreduce -L -256M volume_group/logical_volume.Creating a logical volumeTo create a LV homevol in a VG MyVolGroup with 300 GiB of capacity, run:# lvcreate -L 300G MyVolGroup -n homevolor, to create a LV homevol in a VG MyVolGroup with the rest of capacity, run:# lvcreate -l 100%FREE MyVolGroup -n homevolTo create the LV while restricting it to specific PVs within the VG, append them to the command:# lvcreate -L 300G MyVolGroup -n homevol /dev/sda1The new LV will appear as /dev/MyVolGroup/homevol. Now you can format the LV with an appropriate file system.You can check the LV is created using the following command:# lvsRenaming a logical volumeTo rename an existing logical volume, use the lvrename(8) command.Either of the following commands renames logical volume old_vol in volume group MyVolGroup to new_vol.# lvrename /dev/MyVolGroup/old_vol /dev/MyVolGroup/new_vol# lvrename MyVolGroup old_vol new_volMake sure to update all configuration files (e.g. /etc/fstab or /etc/crypttab) that reference the renamed logical volume.Resizing the logical volume and file system in one go Extend the logical volume mediavol in MyVolGroup by 10 GiB and resize its file system all at once:# lvresize -L +10G --resizefs MyVolGroup/mediavolSet the size of logical volume mediavol in MyVolGroup to 15 GiB and resize its file system all at once:# lvresize -L 15G --resizefs MyVolGroup/mediavolIf you want to fill all the free space on a volume group, use the following command:# lvresize -l +100%FREE --resizefs MyVolGroup/mediavolSee lvresize(8) for more detailed options.Resizing the logical volume and file system separatelyFor file systems not supported by fsadm(8) will need to use the appropriate utility to resize the file system before shrinking the logical volume or after expanding it.To extend logical volume mediavol within volume group MyVolGroup by 2 GiB without touching its file system:# lvresize -L +2G MyVolGroup/mediavolNow expand the file system (ext4 in this example) to the maximum size of the underlying logical volume:# resize2fs /dev/MyVolGroup/mediavolFor Btrfs, btrfs-filesystem(8) expects the mountpoint instead of the device, the equivalent is:# btrfs filesystem resize max /mnt/my-mountpointTo reduce the size of logical volume mediavol in MyVolGroup by 500 MiB, first calculate the resulting file system size and shrink the file system (Ext4 in this example) to the new size:# resize2fs /dev/MyVolGroup/mediavol NewSizeUnlike Ext4, Btrfs supports online shrinking (again, a mountpoint should be specified) e.g.:# btrfs filesystem resize -500M /mnt/my-mountpointWhen the file system is shrunk, reduce the size of logical volume:# lvresize -L -500M MyVolGroup/mediavolTo calculate the exact logical volume size for ext2, ext3, ext4 file systems, use a simple formula: LVM_EXTENTS = FS_BLOCKS FS_BLOCKSIZE LVM_EXTENTSIZE.# tune2fs -l /dev/MyVolGroup/mediavol | grep BlockBlock count: 102400000Block size: 4096Blocks per group: 32768# vgdisplay MyVolGroup | grep "PE Size"PE Size 4.00 MiBNote: The file system block size is in bytes. Make sure to use the same units for both block and extent size.102400000 blocks 4096 bytes/block 4 MiB/extent = 100000 extentsPassing --resizefs will confirm that the correctness.# lvreduce -l 100000 --resizefs /dev/MyVolGroup/mediavol...The filesystem is already 102400000 (4k) blocks long. Nothing to do!...Logical volume sysvg/root successfully resized.See lvresize(8) for more detailed options.Removing a logical volumeWarning: Before you remove a logical volume, make sure to move all data that you want to keep somewhere else; otherwise, it will be lost!First, find out the name of the logical volume you want to remove. You can get a list of all logical volumes with:# lvsNext, look up the mountpoint of the chosen logical volume:$ lsblkThen unmount the filesystem on the logical volume:# umount /mountpointFinally, remove the logical volume:# lvremove volume_group/logical_volumeFor example:# lvremove MyVolGroup/homevolConfirm by typing in y.Make sure to update all configuration files (e.g. /etc/fstab or /etc/crypttab) that reference the removed logical volume.You can verify the removal of the logical volume by typing lvs as root again (see first step of this section).SnapshotsLVM supports CoW (Copy-on-Write) snapshots. A CoW snapshot initially points to the original data. When data blocks are overwritten, the original copy is left intact and the new blocks are written elsewhere on-disk. This has several desirable properties:Creating snapshots is fast, because it does not copy data (just the much shorter list of pointers to the on-disk locations).Snapshots require just enough free space to hold the new data blocks (plus a negligible amount for the pointers to the new blocks). For example, a snapshot of 35 GiB of data, where you write only 2 GiB (on both the original and snapshot), only requires 2 GiB of free space.LVM snapshots are at the block level. They make a new block device, with no apparent relationship to the original except when dealing with the LVM tools. Therefore, deleting files in the original copy does not free space in the snapshots. If you need filesystem-level snapshots, you rather need btrfs, ZFS or bcachefs.Warning: A CoW snapshot is not a backup, because it does not make a second copy of the original data. For example, a damaged disk sector that affects original data also affects the snapshots. That said, a snapshot can be helpful while using other tools to make backups, as outlined below.Btrfs expects different filesystems to have different UUIDs. If you snapshot a LVM volume that contains a btrfs filesystem, make sure to change the UUID of the original or the copy, before both are mounted (or made visible to the kernel, for example if an unrelated daemon triggers a btrfs device scan). For details see btrfs wiki Gotcha's.ConfigurationYou create snapshot logical volumes just like normal ones.# lvcreate --size 100M --snapshot --name snap01vol /dev/MyVolGroup/lvolWith that volume, you may modify less than 100 MiB of data, before the snapshot volume fills up.Reverting the modified lvol logical volume to the state when the snap01vol snapshot was taken can be done with# lvconvert --merge /dev/MyVolGroup/snap01volIn case the origin logical volume is active, merging will occur on the next reboot (merging can be done even from a LiveCD). Note: The snapshot will no longer exist after merging.Also multiple snapshots can be taken and each one can be merged with the origin logical volume at will.BackupsA snapshot provides a frozen copy of a file system to make backups. For example, a backup taking two hours provides a more consistent image of the file system than directly backing up the partition.The snapshot can be mounted and backed up with dd or tar. The size of the backup file done with dd will be the size of the files residing on the snapshot volume. To restore just create a snapshot, mount it, and write or extract the backup to it. And then merge it with the origin.See Create root filesystem snapshots with LVM for automating the creation of clean root file system snapshots during system startup for backup and rollback.This article or section needs expansion.Reason: Show scripts to automate snapshots of root before updates, to rollback... updating menu.lst to boot snapshots (maybe in a separate article?) (Discuss in Talk:LVM)EncryptionSee dm-crypt/Encrypting an entire system#LUKS on LVM and dm-crypt/Encrypting an entire system#LVM on LUKS for the possible schemes of combining LUKS with LVM.CacheThis article or section needs expansion.Reason: LVM also supports --type writecache which uses dm-writecache. (Discuss in Talk:LVM)From lvmcache(7):The cache logical volume type uses a small and fast LV to improve the performance of a large and slow LV. It does this by storing the frequently used blocks on the faster LV. LVM refers to the small fast LV as a cache pool LV. The large slow LV is called the origin LV. Due to requirements from dm-cache (the kernel driver), LVM further splits the cache pool LV into two devices - the cache data LV and cache metadata LV. The cache data LV is where copies of data blocks are kept from the origin LV to increase speed. The cache metadata LV holds the accounting information that specifies where data blocks are stored (e.g. on the origin LV or on the cache data LV). Users should be familiar with these LVs if they wish to create the best and most robust cached logical volumes. All of these associated LVs must be in the same VG.Create cacheConvert your fast disk (/dev/fastdisk) to PV and add to your existing VG (MyVolGroup):# vgextend MyVolGroup /dev/fastdiskCreate a cache pool with automatic meta data on /dev/fastdisk and convert the existing LV MyVolGroup/rootvol to a cached volume, all in one step:# lvcreate --type cache --cachemode writethrough -l 100%FREE -n root_cachepool MyVolGroup/rootvol /dev/fastdiskTip: Instead of using -l 100%FREE to allocate 100% of available space from PV /dev/fastdisk, you can use -L 20G instead to allocate only 20 GiB for cachepool.Cachemode has two possible options:writethrough ensures that any data written will be stored both in the cache pool LV and on the origin LV. The loss of a device associated with the cache pool LV in this case would not mean the loss of any data;writeback ensures better performance, but at the cost of a higher risk of data loss in case the drive used for cache fails.If a specific --cachemode is not indicated, the system will assume writethrough as default.Tip: Cache hit and miss counts can be viewed with lvdisplay or alternatively with lvm-cache-stats from libblockdev-lvm which also shows them in percentages.If you ever need to undo the one step creation operation above:# lvconvert --uncache MyVolGroup/rootvolThis commits any pending writes still in the cache back to the origin LV, then deletes the cache. Other options are available and described in lvmcache(7).RAIDLVM may be used to create a software RAID. It is a good choice if the user does not have hardware RAID and was planning on using LVM anyway. From lvmraid(7):lvm(8) RAID is a way to create a Logical Volume (LV) that uses multiple physical devices to improve performance or tolerate device failures. In LVM, the physical devices are Physical Volumes (PVs) in a single Volume Group (VG).LVM RAID supports RAID 0, RAID 1, RAID 4, RAID 5, RAID 6 and RAID 10. See Wikipedia:Standard RAID levels for details on each level.Tip: mdadm may also be used to create a software RAID. It is arguably simpler, more popular, and easier to setup.Setup RAIDCreate physical volumes:# pvcreate /dev/sda2 /dev/sdb2Create volume group on the physical volumes:# vgcreate MyVolGroup /dev/sda2 /dev/sdb2New volumesCreate logical volumes using lvcreate --type raidlevel, see lvmraid(7) and lvcreate(8) for more options.# lvcreate --type RaidLevel [OPTIONS] -n Name -L Size VG [PVs]RAID 0For example:# lvcreate -n myraid1vol -i 2 -I 64 -L 70G VolGroup00 /dev/nvme1n1p1 /dev/nvme0n1p1will create a 70 GiB striped (raid0) logical volume named "myraid1vol" in VolGroup00. Stripes will be spread over /dev/nvme1n1p1 and /dev/nvme0n1p1. Stripesize is set to be 64K.RAID 1For example:# lvcreate --type raid1 --mirrors 1 -L 20G -n myraid1vol MyVolGroup /dev/sda2 /dev/sdb2will create a 20 GiB mirrored logical volume named "myraid1vol" in VolGroup00 on /dev/sda2 and /dev/sdb2.RAID 10For example:# lvcreate -n myraid1vol -L 100G --type raid10 -m 1 -i 2 MyVolGroup /dev/sdd1 /dev/sdc1 /dev/sdb1 /dev/sda5will create a 100 GiB RAID10 logical volume named "myraid1vol" in VolGroup00 on /dev/sdd1, /dev/sdc1, /dev/sdb1, and /dev/sda5.Existing volumesYou can convert easily a non-RAID (e.g. linear) volume to pretty much any other raid configuration provided that you have enough physical devices to meet the RAID requirements. Some of them will require you to go through intermediate steps which lvconvert will inform you about and prompt you to agree. raid10 below can be replaced with raid0, raid1, raid5 etc.# lvconvert --type raid10 /dev/vg01/lv01Use specific PVs:# lvconvert --type raid10 /dev/vg01/lv01 /dev/sda1 /dev/sdb2 /dev/nvme0n1p1 ...You can keep track of the progress of conversion with:# watch lvs -o name,vg_name,copy_percentThin provisioningNote: When mounting a thin LV file system, always remember to use the discard option or to use fstrim regularly, to allow the thin LV to shrink as files are deleted.From lvmthin(7):Blocks in a standard lvm(8) Logical Volume (LV) are allocated when the LV is created, but blocks in a thin provisioned LV are allocated as they are written. Because of this, a thin provisioned LV is given a virtual size, and can then be much larger than physically available storage. The amount of physical storage provided for thin provisioned LVs can be increased later as the need arises.Example: implementing virtual private serversHere is the classic use case. Suppose you want to start your own VPS service, initially hosting about 100 VPSes on a single PC with a 930 GiB hard drive. Hardly any of the VPSes will actually use all of the storage they are allotted, so rather than allocate 9 GiB to each VPS, you could allow each VPS a maximum of 30 GiB and use thin provisioning to only allocate as much hard drive space to each VPS as they are actually using. Suppose the 930 GiB hard drive is /dev/sdb. Here is the setup.Prepare the volume group, MyVolGroup.# vgcreate MyVolGroup /dev/sdbCreate the thin pool LV, MyThinPool. This LV provides the blocks for storage.# lvcreate --type thin-pool -n MyThinPool -l 95%FREE MyVolGroupThe thin pool is composed of two sub-volumes, the data LV and the metadata LV. This command creates both automatically. But the thin pool stops working if either fills completely, and LVM currently does not support the shrinking of either of these volumes. This is why the above command allows for 5% of extra space, in case you ever need to expand the data or metadata sub-volumes of the thin pool.For each VPS, create a thin LV. This is the block device exposed to the user for their root partition.# lvcreate -n SomeClientsRoot -V 30G --thinpool MyThinPool MyVolGroupThe block device /dev/MyVolGroup/SomeClientsRoot may then be used by a VirtualBox instance as the root partition.Use thin snapshots to save more spaceThin snapshots are much more powerful than regular snapshots, because they are themselves thin LVs. See Red Hat's guide [4] for a complete list of advantages thin snapshots have.Instead of installing Linux from scratch every time a VPS is created, it is more space-efficient to start with just one thin LV containing a basic installation of Linux:# lvcreate -n GenericRoot -V 30G --thinpool MyThinPool MyVolGroup*** install Linux at /dev/MyVolGroup/GenericRoot ***Then create snapshots of it for each VPS:# lvcreate -s MyVolGroup/GenericRoot -n SomeClientsRootThis way, in the thin pool there is only one copy the data common to all VPSes, at least initially. As an added bonus, the creation of a new VPS is instantaneous.Since these are thin snapshots, a write operation to GenericRoot only causes one COW operation in total, instead of one COW operation per snapshot. This allows you to update GenericRoot more efficiently than if each VPS were a regular snapshot.Example: zero-downtime storage upgradeThere are applications of thin provisioning outside of VPS hosting. Here is how you may use it to grow the effective capacity of an already-mounted file system without having to unmount it. Suppose, again, that the server has a single 930 GiB hard drive. The setup is the same as for VPS hosting, only there is only one thin LV and the LV's size is far larger than the thin pool's size.# lvcreate -n MyThinLV -V 16T --thinpool MyThinPool MyVolGroupThis extra virtual space can be filled in with actual storage at a later time by extending the thin pool.Suppose some time later, a storage upgrade is needed, and a new hard drive, /dev/sdc, is plugged into the server. To upgrade the thin pool's capacity, add the new hard drive to the VG:# vgextend MyVolGroup /dev/sdcNow, extend the thin pool:# lvextend -l +95%FREE MyVolGroup/MyThinPoolSince this thin LV's size is 16 TiB, you could add another 15.09 TiB of hard drive space before finally having to unmount and resize the file system.Note: You will probably want to use reserved blocks or a disk quota to prevent applications from attempting to use more physical storage than there actually is.CustomizingSome customisation is available by editing /etc/lvm/lvm.conf. You may find it useful to customize the output of lvs and pvs which by default does not include the% sync (useful to see progress of conversion between e.g. linear and raid type) and type of logical volume:/etc/lvm/lvm.confreport { lvs_cols = "lv_name,lv_attr,lv_active,vg_name,lv_size,lv_layout,lv_allocation_policy,copy_percent,chunk_size"pvs_cols = "pv_name,vg_name,pv_size,pv_free,pv_used,dev_size"}TroubleshootingLVM commands do not work # modprobe dm_modThe dm_mod module should be automatically loaded. In case it does not, explicitly load the module at boot. Try preceding commands with lvm like this:# lvm pvdisplayLogical volumes do not show upIf you are trying to mount existing logical volumes, but they do not show up in lvscan, you can use the following commands to activate them:# vgscan# vgchange -ayLVM on removable mediaSymptoms:# vgscan Reading all physical volumes. This may take a while... /dev/backupdrive1/backup: read failed after 0 of 4096 at 319836585984: Input/output error /dev/backupdrive1/backup: read failed after 0 of 4096 at 319836643328: Input/output error /dev/backupdrive1/backup: read failed after 0 of 4096 at 0: Input/output error /dev/backupdrive1/backup: read failed after 0 of 4096 at 4096: Input/output error Found volume group "backupdrive1" using metadata type lvm2 Found volume group "networkdrive" using metadata type lvm2Cause: removing an external LVM drive without deactivating the volume group(s) first. Before you disconnect, make sure to:# vgchange -an volume_group_nameFix: assuming you already tried to activate the volume group with vgchange -ay vg, and are receiving the Input/output errors:# vgchange -an volume_group_nameUnplug the external drive and wait a few minutes:# vgscan# vgchange -ay volume_group_nameSuspend/resume with LVM and removable mediaThe factual accuracy of this article or section is disputed. In order for LVM to work properly with removable media like an external USB drive the volume group of the external drive needs to be deactivated before suspend. If this is not done, you may get buffer I/O errors on the dm device (after resume). For this reason, it is not recommended to mix external and internal drives in the same volume group.To automatically deactivate the volume groups with external USB drives, tag each volume group with the sleep_umount tag in this way:# vgchange --addtag sleep_umount vg_externalOnce the tag is set, use the following unit file for systemd to properly deactivate the volumes before suspend. On resume, they will be automatically activated by LVM./etc/systemd/system/ext_usb_vg_deactivate.service[Unit]Description=Deactivate external USB volume groups on suspendBefore=sleep.target[Service]Type=oneshotExecStart=-/etc/systemd/system/deactivate_sleep_vgs.sh[Install]WantedBy=sleep.targetand this script:/etc/systemd/system/deactivate_sleep_vgs.sh#!/bin/shTAG=@sleep_umountvgs=$(vgs --noheadings -o vg_name $TAG)echo "Deactivating volume groups with $TAG tag: $vgs"# Unmount logical volumes belonging to all the volume groups with tag $TAGfor vg in $vgs; do for lv_dev_path in $(lvs --noheadings -o lv_path -S lv_active=active,vg_name=$vg); do echo "Unmounting logical volume $lv_dev_path" umount $lv_dev_path donedone# Deactivate volume groups tagged with sleep_umountfor vg in $vgs; do echo "Deactivating volume group $vg" vgchange -an $vgdoneFinally, enable the unit.Resizing a contiguous logical volume failsIf trying to extend a logical volume errors with:" Insufficient suitable contiguous allocatable extents for logical volume "The reason is that the logical volume was created with an explicit contiguous allocation policy (options -C y or --alloc contiguous) and no further adjacent contiguous extents are available.[5]To fix this, prior to extending the logical volume, change its allocation policy with lvchange --alloc inherit logical_volume. If you need to keep the contiguous allocation policy, an alternative approach is to move the volume to a disk area with sufficient free extents. See [6].Command "grub-mkconfig" reports "unknown filesystem" errorsMake sure to remove snapshot volumes before generating grub.cfg.Thinly-provisioned root volume device times outWith a large number of snapshots, thin_check runs for a long enough time so that waiting for the root device times out. To compensate, add the rootdelay=60 kernel boot parameter to your boot loader configuration. Or, make thin_check skip checking block mappings (see [7]) and regenerate the initramfs:/etc/lvm/lvm.confthin_check_options = ["-q", "--clear-needs-check-flag", "--skip-mappings"]Delay on shutdownIf you use RAID, snapshots or thin provisioning and experience a delay on shutdown, make sure lvm2-monitor.service is started. See FS#50420.Hibernating into a thinly-provisioned swap volumeSee Power management/Suspend and hibernate#Hibernation into a thinly-provisioned LVM volume.See also Related articlesYou should create your LVM Volumes between the partitioning and formatting steps of the installation procedure. Instead of directly formatting a partition to be your root file system, the file system will be created inside a logical volume (LV).Quick overview: Install the required packages. (refer to LVM#Installation)Create partition(s) where your physical volumes (PVs) will reside.Create your PVs. If you have one disk it is best to just create one PV in one large partition. If you have multiple disks you can create partitions on each of them and create a PV on each partition.Create your volume group (VG) and add all PVs to it.Create logical volumes (LVs) inside that VG.Continue with Installation guide#Format the partitions.When you reach the Installation guide#Initramfs section in the Installation guide, add the lvm2 hook to /etc/mkinitcpio.conf (see below for details).Warning: /boot cannot reside in LVM when using a boot loader which does not support LVM (only GRUB is known to support LVM). You must create a separate /boot partition and format it directly.InstallationYou will follow along with the installation guide until you come to Installation guide#Partition the disks. At this point you will diverge and doing all your partitioning with LVM in mind.Create partitionsFirst, partition your disks as required before configuring LVM.Create the partitions:If you use GUID Partition Table, set the partition type GUID to E6D6D379-F507-44C2-A23C-238F2A3DF928 (partition type Linux LVM in fdisk and 8e00 in gdisk).If you use Master Boot Record partition table, set the partition type ID to 8e (partition type Linux LVM in fdisk).Create physical volumesTo list all your devices capable of being used as a physical volume:# lvmdiskscanWarning: Make sure you target the correct device, or below commands will result in data loss!Create a physical volume on them:# pvcreate DEVICEThis command creates a header on each device so it can be used for LVM. As defined in LVM#LVM building blocks, DEVICE can be any block device, e.g. a disk /dev/sda, a partition /dev/sda2 or a loop back device. For example: # pvcreate /dev/sda2You can track created physical volumes with:# pvdisplayYou can also get summary information on physical volumes with:# pvscanTip: If you run into trouble with a pre-existing disk signature, you can delete it using wipefs.Create and extend your volume groupFirst you need to create a volume group on any one of the physical volumes:# vgcreate volume_group physical_volumeFor example:# vgcreate VolGroup00 /dev/sda2See lvm(8) for a list of valid characters for volume group names.Extending the volume group is just as easy:# vgextend volume_group physical_volumeFor example, to add both sdb1 and sdc to your volume group:# vgextend VolGroup00 /dev/sdb1# vgextend VolGroup00 /dev/sdcYou can track how your volume group grows with:# vgdisplayThis is also what you would do if you wanted to add a disk to a RAID or mirror group with failed disks.Note: You can create more than one volume group if you need to, but then you will not have all your storage presented as a single disk.Combined creation of physical volumes and volume groupsLVM allows you to combine the creation of a volume group and the physical volumes in one easy step. For example, to create the group VolGroup00 with the three devices mentioned above, you can run:# vgcreate VolGroup00 /dev/sda2 /dev/sdb1 /dev/sdcThis command will first set up the three partitions as physical volumes (if necessary) and then create the volume group with the three volumes. The command will warn you if it detects an existing filesystem on any devices.Create logical volumesTip: If you wish to use snapshots, logical volume caching, thin provisioned logical volumes or RAID see LVM#Logical volumes.If a logical volume will be formatted with ext4, leave at least 256 MiB free space in the volume group to allow using e2scrub(8). After creating the last volume with -l 100%FREE, this can be accomplished by reducing its size with lvreduce -L -256M volume_group/logical_volume.Now we need to create logical volumes on this volume group. You create a logical volume with the next command by specifying the new volume's name and size, and the volume group it will reside on:# lvcreate -L size volume_group -n logical_volumeFor example:# lvcreate -L 10G VolGroup00 -n lvolhomeThis will create a logical volume that you can access later with /dev/VolGroup00/lvolhome. Just like volume groups, you can use any name you want for your logical volume when creating it besides a few exceptions listed in lvm(8) VALID_NAMES.You can also specify one or more physical volumes to restrict where LVM allocates the data. For example, you may wish to create a logical volume for the root filesystem on your small SSD, and your home volume on a slower mechanical drive. Simply add the physical volume devices to the command line, for example:# lvcreate -L 10G VolGroup00 -n lvolhome /dev/sdc1To use all the free space left in a volume group, use the next command:# lvcreate -l 100%FREE volume_group -n logical_volumeYou can track created logical volumes with:# lvdisplayNote: You may need to load the device-mapper kernel module (modprobe dm_mod) for the above commands to succeed.Tip: You can start out with relatively small logical volumes and expand them later if needed. For simplicity, leave some free space in the volume group so there is room for expansion.Format and mount logical volumesYour logical volumes should now be located in /dev/YourVolumeGroupName/. If you cannot find them, use the next commands to bring up the module for creating device nodes and to make volume groups available:# modprobe dm_mod# vgscan# vgchange -ayNow you can format your logical volumes and mount them as normal partitions (see mount a file system for additional details):# mkfs.fstype /dev/volume_group/logical_volume# mount /dev/volume_group/logical_volume /mountpointFor example:# mkfs.ext4 /dev/VolGroup00/lvolhome# mount /dev/VolGroup00/lvolhome /homeWarning: When choosing mountpoints, just select your newly created logical volumes (use: /dev/Volgroup00/lvolhome). Do not select the actual partitions on which logical volumes were created (do not use: /dev/sda2).Configure the systemMake sure the lvm2 package is installed.Tip: lvm2 provides the lvm2 hook. If you are running mkinitcpio in an arch-chroot for a new installation, lvm2 must be installed inside the arch-chroot for mkinitcpio to find the lvm2 hook. If lvm2 only exists outside the arch-chroot, mkinitcpio will output Error: Hook 'lvm2' cannot be found.Adding mkinitcpio hooksIn case your root filesystem is on LVM, you will need to enable the appropriate mkinitcpio hooks, otherwise your system might not boot. Enable:udev and lvm2 for the default busybox-based initramfssystemd and lvm2 for systemd-based initramfsudev is there by default. Edit the file and insert lvm2 between block and filesystems like so:/etc/mkinitcpio.confHOOKS=(base udev ... block lvm2 filesystems)For systemd based initramfs:/etc/mkinitcpio.confHOOKS=(base systemd ... block lvm2 filesystems)Afterwards, you can continue in normal installation instructions with the recreate the initramfs image step.Kernel boot optionsIf the root file system resides in a logical volume, the root= kernel parameter must be pointed to the mapped device, e.g /dev/vg-name/lv-name.Alles Notwendige, was Sie ber die logische Volume-Verwaltung unter Linux wissen und lernen mssen.Dies ist eine vollstndige Einsteigeranleitung fr LVM (Logical Volume Management) unter Linux.In diesem Tutorial lernen Sie das Konzept von LVM, seine Komponenten und warum Sie es verwenden sollten.Ich werde mich nicht nur auf die theoretische Erklrung beschrnken. Auerdem zeige ich praktische Beispiele fr die Erstellung und Verwaltung von LVMs unter Linux.Kurz gesagt, ich werde Ihnen alle notwendigen Informationen geben, die Sie bentigen, um mit LVM in der realen Welt zu arbeiten.Was ist LVM unter Linux?LVM steht fr Logical Volume Management. Dies ist eine alternative Methode zur Verwaltung von Speichersystemen als die herkmmliche, auf Partitionen basierende. In LVM erstellen Sie statt Partitionen logische Volumes und knnen diese Volumes dann genauso einfach in Ihr Dateisystem einbinden wie eine Festplattenpartition.Komponenten von LVMLVM besteht aus drei Hauptkomponenten:Physische DatentrgerVolumengruppenLogische VolumesObwohl die Liste aus drei Komponenten besteht, sind nur zwei davon direkte Gegenstcke zum Partitionierungssystem. Die folgende Tabelle protokolliert dies.PartitionsLogische VolumesDisksVolumengruppenPhysische Bnde haben kein direktes Gegenstck, aber darber werde ich bald sprechen.Warum LVM verwenden?Der Hauptvorteil von LVM besteht darin, wie einfach es ist, die Gre eines logischen Volumes oder einer Volume-Gruppe zu ndern. Es abstrahiert alle hsslichen Teile (Partitionen, Raw-Festplatten) und hinterlsst einen zentralen Speicherpool, mit dem wir arbeiten knnen. Wenn Sie jemals den Schrecken einer Partitionsgrennderung erlebt haben, sollten Sie LVM verwenden.Laborvorbereitung fr praktische bungenDieser Artikel ist nicht nur Theorie. Unterwegs zeige ich konkrete Befehlsbeispiele und der beste Weg, etwas zu lernen, besteht darin, praktisch damit zu arbeiten. Dafr empfehle ich Ihnen die Verwendung einer virtuellen Maschine.Um Ihnen dabei zu helfen, habe ich bereits eine einfache Vagrant-Datei vorbereitet, mit der Sie mit VirtualBox eine sehr einfache virtuelle Maschine starten knnen. Diese virtuelle Maschine verfgt ber drei zustzliche Festplatten, die Sie und ich fr die folgenden Befehlsbeispiele verwenden knnen.Erstellen Sie irgendwo in Ihrem Dateisystem ein Verzeichnis und speichern Sie dort Folgendes in einer Datei mit dem Namen Vagrantfile.Vagrant.configure "2" do |config| config.vm.box = "ubuntu/focal64" config.vm.hostname = "lvm" 3.times {|i| config.vm.disk :disk, size: "5GB", name: "drive-#{i}"} config.vm.provider :virtualbox do |machine| machine.memory = 1024 machine.cpus = 1 machine.customize ["modifyvm", :id, "--cpuexecutioncap", "50"] endend# vi: set ft=rubyWenn Sie mchten, knnen Sie die Datei auch mit wget oder curl von meinem Gist herunterladen.wget Sie sicher, dass Vagrant und VirtualBox installiert sind.Sobald die Vagrant-Datei vorhanden ist, setzen Sie die Umgebungsvariable VAGRANT_EXPERIMENTAL auf disks.export VAGRANT_EXPERIMENTAL=disksStarten Sie abschlieend die virtuelle Maschine mit dem folgenden Befehl (stellen Sie sicher, dass Sie sich im selben Verzeichnis wie die Vagrantfile befinden):vagrant upSobald die Maschine luft, knnen Sie mit vagrant ssh eine SSH-Verbindung herstellen und die Beispielbefehle aus diesem Artikel ausfhren.Denken Sie daran, vagrant destroy aus demselben Verzeichnis wie die Vagrant-Datei auszufhren, nachdem Sie fertig sind.LVM installierenBevor Sie einen der Befehle verwenden knnen, mssen Sie das Paket lvm2 installieren. Dies sollte in den meisten modernen Distributionen vorinstalliert sein, insbesondere in den Ubuntu-basierten. Dennoch musste ich dies erwhnen, bevor ich fortfuhr. Informationen zur Installation von lvm2 finden Sie in der Dokumentation Ihrer Distribution.Praktisch mit LVMFr diesen praktischen Rundgang habe ich eine virtuelle Maschine mit 40 GB Root-Speicher (unwichtig) und drei externen Festplatten der Gre 5 GB erstellt. Die Gre dieser Festplatten ist beliebig.vagrant@lvm:~$ lsblk -o name,size,fstypeNAME SIZE FSTYPEloop0 55.5M squashfsloop1 32.3M squashfsloop2 70.4M squashfssda 40G sda1 40G ext4sdb 10M iso9660sdc 5G sdd 5G sde 5GWie Sie sehen, werde ich folgende Gerte verwenden: sdc, sdd und sde.Erinnern Sie sich, dass ich Ihnen gesagt habe, dass LVM aus drei Hauptkomponenten besteht?Physische DatentrgerVolumengruppenLogische VolumesEs ist Zeit, sie einzeln zu sehen.1. Physische DatentrgerDas allererste, was Sie ber LVM wissen mssen, sind physische Volumes. Physische Datentrger sind die Rohmaterialien oder Bausteine, die verwendet werden, um die Abstraktion logischer Datentrger zu erreichen. Vereinfacht ausgedrckt sind physische Volumes die logische Einheit eines LVM-Systems.Ein physisches Volume kann alles sein, eine Raw-Festplatte oder eine Festplattenpartition. Das Erstellen und Initialisieren eines physischen Volumes ist dasselbe. Beides bedeutet, dass Sie lediglich die Bausteine (z. B. Partitionen, Festplatten) fr weitere Vorgnge vorbereiten. Das wird gleich klarer.Dienstprogramme: Alle Dienstprogramme, die physische Volumes verwalten, beginnen mit den Buchstaben pv fr Physical Volume. Z.B. pvcreate, pvchange, pvs, pvdisplay usw.Alles, was nach diesem Feld erwhnt wird, ist destruktiv. Es sei denn, Sie verwenden eine virtuelle Umgebung oder einen Server in der Cloud, auf dem keine wichtigen Daten gespeichert sind oder auf die versehentlich zugegriffen werden kann. Ich rate Ihnen, sofort damit aufzuhren.Sie knnen ein physisches Volume mithilfe einer unformatierten, nicht partitionierten Festplatte oder der Partitionen selbst erstellen.Wie ich bereits erwhnt habe, habe ich drei externe Laufwerke an meine virtuelle Maschine angeschlossen. Beginnen wir mit /dev/sdc.Wir verwenden den Befehl pvcreate, um ein physisches Volume zu erstellen. bergeben Sie ihm einfach den Gertenamen und sonst nichts.sudo pvcreate /dev/sdcSie sollten etwa Folgendes sehen: -vagrant@lvm:~$ sudo pvcreate /dev/sdc Physical volume "/dev/sdc" successfully created.Als nchstes werde ich /dev/sdd in gleiche Teile partitionieren. Verwenden Sie ein beliebiges Tool, cfdisk, parted, fdisk usw., es gibt viele Tools, um diese Aufgabe zu erfllen.vagrant@lvm:~$ lsblk -o name,size,fstype | grep sddsdd 5G sdd1 2.5G sdd2 2.5GSie knnen jetzt in einem einzigen Schritt schnell zwei weitere physische Volumes aus diesen beiden Partitionen erstellen und beide Gerte gleichzeitig an pvcreate bergeben.sudo pvcreate /dev/sdd1 /dev/sdd2Schau mal:-vagrant@lvm:~$ sudo pvcreate /dev/sdd1 /dev/sdd2 Physical volume "/dev/sdd1" successfully created. Physical volume "/dev/sdd2" successfully created.Auflistung der verfgbaren physischen DatentrgerEs gibt drei Befehle, mit denen Sie die Liste der verfgbaren physischen Volumes abrufen knnen: pvscan, pvs und pvdisplay. Im Allgemeinen mssen Sie diesen Befehlen nichts bergeben.pvscan:-vagrant@lvm:~$ sudo pvscan PV /dev/sdc lvm2 [5.00 GiB] PV /dev/sdd1 lvm2 [2.50 GiB] PV /dev/sdd2 lvm2 [/etc/mdadm.conf Kontrolle: cat /etc/mdadm.conf DEVICE /dev/sd* ARRAY /dev/md1 metadata=0.90 UUID=f0045bf9:ddcdb29b:a24daeb9:4e1c5a02 ARRAY /dev/md2 metadata=0.90 UUID=0611c9b8:1aa41ea9:a24daeb9:4e1c5a02 ARRAY /dev/md3 metadata=0.90 UUID=97fcbd65:935c835b:a24daeb9:4e1c5a02 Nun mssen die neuen RAIDPartitionen formatiert werden: for i in 2 3; do mkfs.ext4 /dev/md$i; done Filesystem label= OS type: Linux Block size=4096 (log=2) Fragment size=4096 (log=2) 328000 inodes, 1311280 blocks 65564 blocks (5.00%) reserved for the super user First data block=0 Maximum filesystem blocks=1346371584 41 block groups 32768 blocks per group, 32768 fragments per group 8000 inodes per group Superblock backups stored on blocks: 32768, 98304, 163840, 229376, 294912, 819200, 884736 Writing inode tables: 0/41 Creating journal (32768 blocks): done Writing superblocks and filesystem accounting information: done This filesystem will be automatically checked every 34 mounts or 180 days, whichever comes first. Use tune2fs -c or -i to override. Filesystem label= OS type: Linux Block size=4096 (log=2) Fragment size=4096 (log=2) 124160 inodes, 495984 blocks 24799 blocks (5.00%) reserved for the super user First data block=0 Maximum filesystem blocks=511705088 16 block groups 32768 blocks per group, 32768 fragments per group 7760 inodes per group Superblock backups stored on blocks: 32768, 98304, 163840, 229376, 294912 Writing inode tables: 0/16 Creating journal (8192 blocks): done Writing superblocks and filesystem accounting information: done This filesystem will be automatically checked every 37 mounts or 180 days, whichever comes first. Use tune2fs -c or -i to override. Nun noch das swap device initialisieren: mkswap /dev/md1 Nun muss dafr gesorgt werden, dass beim Booten auch das Startscript fr den RAID ausgefhrt wird. insserv /etc/init./boot.md aufrufen. Nun mssen alle Daten von den OriginalPartitionen auf die RaidPartitionen kopiert werden. Das dauert ein wenig. Achtung! Die / am Ende sind wichtig! mount /dev/md2 /mnt; rsync -AHPSXavx / /mnt/; umount /mnt/ mount /dev/md3 /mnt; rsync -AHPSXavx /home /mnt/; umount /mnt/ Jetzt kann getestet werden, ob das System auch vom RAID bootet. Dazu muss aber nochdie neue /etc/fstab so gendert werden, dass dort die RAIDPartitionen benutzt werden.Dazu muss die neue RAIDRootPartition (im Beispiel /dev/md2) noch einmal als /mnt gemountet werden und dann dort in der /etc/fstab /dev/sda1 in /dev/md1 gendert werden. Dasselbe natrlich auch fr /dev/sda2 und /dev/sda3. Ausserdem muss die initrd noch Informationen erhalten, so dass ein Booten die RAIDs konfiguriert werden. mkinitrd -f md mount /dev/md2 /mnt Anschliessend muss die /etc/fstab wie folgt aus sehen (Die alten Zeilen sind mit einem # am Anfang auskommentiert). #/dev/sad1 swap swap defaults 0 0 /dev/md1 swap swap defaults 0 0 #/dev/sda2 / ext4 acl,user_xattr 1 1 /dev/md2 / ext4 acl,user_xattr 1 1 #/dev/sda3 /home ext4 acl,user_xattr 1 2 /dev/md3 /home ext4 acl,user_xattr 1 2 proc /proc proc defaults 0 0 sysfs /sys sysfs noauto 0 0 debugfs /sys/kernel/debug debugfs noauto 0 0 usbfs /proc/bus/usb usbfs noauto 0 0 devpts /dev/pts devpts mode=0620,gid=5 0 0 Dann muss die /boot/grub/menu.lst so gendert werden, dass das Betriebssystem sowohl von dre alten Partition als auch dre neuen RAIDPartition gestartet werden kann. Dazu muss der erste Eintrag dupliziert und am Ende angehngt werden und sda2 bzw sda1 ind md2 bzw md1 gendert werden. # Modified by YaST2. Last modification on Sun Jun 13 00:10:27 CEST 2010 # THIS FILE WILL BE PARTIALLY OVERWRITTEN by perl-Bootloader # Configure custom boot parameters for updated kernels in /etc/sysconfig/bootloader default 0 timeout 8 ##YaST - generic_mbr gfxmenu (hd0,1)/boot/message ##YaST - activate ###Don't change this comment - YaST2 identifier: Original name: linux### title openSUSE 11.2 - 2.6.31.5-0.1 root (hd0,1) kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/sda2 resume=/dev/sda1 splash=silent quiet showopts vga=0x314 initrd /boot/initrd-2.6.31.5-0.1-default ###Don't change this comment - YaST2 identifier: Original name: failsafe### title Failsafe -- openSUSE 11.2 - 2.6.31.5-0.1 root (hd0,1) kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/sda2 showopts apm=off noresume nosmp maxcpus=0 edd=off powersaved=off nohz=off highres=off processor.max_cstate=1 x11failsafe vga=0x314 initrd /boot/initrd-2.6.31.5-0.1-default ###Don't change this comment - YaST2 identifier: Original name: floppy### title Floppy rootnoverify (fd0) chainloader +1 ###Don't change this comment - YaST2 identifier: Original name: linux### title openSUSE 11.2 - 2.6.31.5-0.1 - RAID root (hd0,1) kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/md2 resume=/dev/md1 splash=silent quiet showopts vga=0x314 initrd /boot/initrd-2.6.31.5-0.1-default Jetzt kann nach einem Reboot der letzte Menupunkt ausgewhlt werden und das System startet vom RAID. Falls was schiefgeht kann man immer noch wieder den ersten Menupunkt auswhlen und von der OriginalsystemPartition booten. Danach sollte man das halbe RAID System noch einmal prfen ob swap sowie root und home Partition auf dem RAID liegen: swapon -s Filename Type Size Used Priority /dev/md1 partition 1156536 0 -1 mount /dev/md2 on / type ext4 (rw,acl,user_xattr) proc on /proc type proc (rw) sysfs on /sys type sysfs (rw) debugfs on /sys/kernel/debug type debugfs (rw) udev on /dev type tmpfs (rw) devpts on /dev/pts type devpts (rw,mode=0620,gid=5) /dev/md3 on /home type ext4 (rw,acl,user_xattr) securityfs on /sys/kernel/security type securityfs (rw) none on /proc/fs/vmblock/mountPoint type vmblock (rw) Nun erfolgt das Einbinden der OriginalPartitionen in den RAID1 so dass es ein wirklicher RAID1 wird. Wenn dabei etwas schief geht besteht die grosse Gefahr, dass man seinen Backup zurckspielen muss. Deshalb sind die folgenden Schritte sehr sorgfltig vorzunehmen. Alle Partitions auf /dev/sda mssen nun als RAIDpartitions gekennzeichnet werden. (Die Nummern 1 2 3 bezeichnen die Partitionen, die gendert werden sollen) for i in 1 2 3; do sfdisk --change-id /dev/sda $i fd; done; sfdisk -R /dev/sda DiePatritionen werde nun ins RAID eingebunden: for i in 1 2 3; do mdadm /dev/md$i -a /dev/sda$i; done Die laufende Plattensynchronisation kann man verfolgen und erst nach dem Ende der Synchronisation darf weitergemacht werden, da die grub EIntrge nun nicht mehr stimmen. watch cat /proc/mdstat Personalities : [raid1] [raid0] [raid6] [raid5] [raid4] md3 : active raid1 sda3[2] sdb3[1] 1983936 blocks [2/1] [_U] resync=DELAYED md2 : active raid1 sda2[2] sdb2[1] 5245120 blocks [2/1] [_U] [=>..................] recovery = 9.6% (504768/5245120) finish=2.8min speed=28042K/sec md1 : active raid1 sda1[0] sdb1[1] 1156544 blocks [2/2] [UU] unused devices: und sollte zum Schluss wie folgt aussehen: Personalities : [raid1] [raid0] [raid6] [raid5] [raid4] md3 : active raid1 sda3[0] sdb3[1] 1983936 blocks [2/2] [UU] md2 : active raid1 sda2[0] sdb2[1] 5245120 blocks [2/2] [UU] md1 : active raid1 sda1[0] sdb1[1] 1156544 blocks [2/2] [UU] unused devices: Zum Schluss muss noch das grub Bootmenu in /boot/grub/menu.lst angepasst werden sowie die bootloader auf beiden Platten erstellt werden. Die /boot/grub/menu.lst muss am Kopf eine weitere zeile fr fallback bekommen und die weiteren Eintrge mssen von sda auf md umgestellt werden.Ausserdem ist ein weiterer Eintrag zu erstellen der im Ausfall der ersten Platte von der zweiten Platte bootet (root ist gendert). Wichtig ist auch dass der erste Eintrag gendert wird und dann dahinter noch einmal kopiert wirdund dort root gendert wird. Ansonsten stimmt die Zahl 1 nach dem fallback nicht. # Modified by YaST2. Last modification on Sun Jun 13 00:10:27 CEST 2010 # THIS FILE WILL BE PARTIALLY OVERWRITTEN by perl-Bootloader # Configure custom boot parameters for updated kernels in /etc/sysconfig/bootloader default 0 # added for RAID fallback 1 timeout 8 ##YaST - generic_mbr gfxmenu (hd0,1)/boot/message ##YaST - activate ###Don't change this comment - YaST2 identifier: Original name: linux### title openSUSE 11.2 - 2.6.31.5-0.1 - RAID1 Disk 1 root (hd0,1) # kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/sda2 resume=/dev/sda1 splash=silent quiet showopts vga=0x314 kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/md2 resume=/dev/md1 splash=silent quiet showopts vga=0x314 initrd /boot/initrd-2.6.31.5-0.1-default # added for RAID title openSUSE 11.2 - 2.6.31.5-0.1 - RAID1 Disk 2 root (hd1,1) kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/md2 resume=/dev/md1 splash=silent quiet showopts vga=0x314 ###Don't change this comment - YaST2 identifier: Original name: failsafe### title Failsafe -- openSUSE 11.2 - 2.6.31.5-0.1 root (hd0,1) # kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/sda2 showopts apm=off noresume nosmp maxcpus=0 edd=off powersaved=off nohz=off highres=off processor.max_cstate=1 x11failsafe vga=0x314 kernel /boot/vmlinuz-2.6.31.5-0.1-default root=/dev/md2 showopts apm=off noresume nosmp maxcpus=0 edd=off powersaved=off nohz=off highres=off processor.max_cstate=1 x11failsafe vga=0x314 initrd /boot/initrd-2.6.31.5-0.1-default ###Don't change this comment - YaST2 identifier: Original name: floppy### title Floppy rootnoverify (fd0) chainloader +1 Nun muss die /boot/grub/devices.map noch erweiter werden um die neue Platte /dev/sdb: (fd0) /dev/fd0 (hd0) /dev/sda (hd1) /dev/sdb Auf beiden Platten muss der grub Bootloader installiert werden. grub grub> root (hd0,1) Filesystem type is ext2fs, partition type 0xfd grub> setup (hd0) Checking if "/boot/grub/stage1" exists... yes Checking if "/boot/grub/stage2" exists... yes Checking if "/boot/grub/e2fs_stage1_5" exists... yes Running "embed /boot/grub/e2fs_stage1_5 (hd0)"... 20 sectors are embedded. succeeded Running "install /boot/grub/stage1 (hd0) (hd0)1+20 p (hd0,1)/boot/grub/stage2 /boot/grub/menu.lst"... succeeded Done. grub grub> root (hd1,1) Filesystem type is ext2fs, partition type 0xfd grub> setup (hd1) Checking if "/boot/grub/stage1" exists... yes Checking if "/boot/grub/stage2" exists... yes Checking if "/boot/grub/e2fs_stage1_5" exists... yes Running "embed /boot/grub/e2fs_stage1_5 (hd1)"... 20 sectors are embedded. succeeded Running "install /boot/grub/stage1 (hd1) (hd1)1+20 p (hd1,1)/boot/grub/stage2 /boot/grub/menu.lst"... succeeded Done. Der letzte Schritt besteht darin, noch einmal die initrd neu zu bauen mkinitrd -f md Damit ist das System von einem nicht RAID System zu einem RAID1 System ungewandelt worden. Damit man auch sofort eine eMail bekommt wenn eine Platte ausfllt muss noch die /etc/sysconfig/mdad.conf angepasst werden und eine gltige eMail konfiguriert werden. Denn leider ntzt ein RAID1 nichts wenn man erst beim Ausfall der zweiten Platte mitbekommt dass schon viele Tage vorher die erste Platte ausgefallen ist.Page 4 Bewertung:5/5 Vor lngerer Zeit hat mich mal jemand im irc chat auf 'screen' hingewiesen. Das Thema war eigentlich nicht screen - aber derjenige hatte gerade die Mchtigkeit von screen kennengelernt und wollte dieses Wissen gleich an mich weiterreichen. Danach habe ich screen oft benutzt wenn auf einem remoten System aktiv war und andere sehen sollten, was ich in der Linux Befehlszeile eingebe und was fr Ergebnisse ich sehe. War besonders hilfreich wenn ich von Leuten bei Linuxproblem gebeten wurde, diese auf ihren Systemen zu beheben, und ich sicherstellen wollte, dass diese zuknftig diese Fehler selbst beheben knnen. Mittlerweile benutze ich screen regelmig auch in weiteren Scenarien. Eben habe ich mir noch mal die man Page von screen angesehen und festgestellt, dass es noch wesentliche weitere Funktionen in screen gibt. Ich habe im Folgenden mal kurz beschrieben was man so alles mit screen anstellen kann, wo es hilft und welche Tastaturbefehle besonders hilfreich sind. Typische fr mich sinnvolle Einsatzflle von screen: Mehrere Leute wollen gleichzeitig sehen, was jemand auf einem System auf der Commandline tut und Ergebnisse gemeinsam ansehen. User 1 gibt screen bei sich ein User 2 verbindet sich von remote auf das System von User 1 und gibt screen -x ein. Danach sehen beide dieselbe Konsole. D.h dann auch dass beide gleichzeitig Befehle eingeben knnen. Da ist also eine gewisse Synchronisation notwendig. Als ganz hilfreich hat es sich erwiesen dann immer direkt auf der Konsole miteinander zu kommunizieren. Das geht ganz einfach wenn man vor jeden Befehl in der Befehlszeile einen Gartenzaun (#) eingibt so dass die bash die Eingabe nur als Kommentar versteht und keine Fehler ausgibt. Man arbeitet remote per ssh auf einem System und braucht mehrere parallele Bildschirme Nach Erstellung von einem Screen durch Eingabe von screen werden weitere Screens durch STRG-a c erstellt und mit STRG-a n und STRG-a p durch die Screens gewechselt. Lngerlaufende Aktivitten sollen von unterschiedlichen remoten Systemen kontrolliert und gesteuert werden. Nach Eingabe von screen werden Aktivitten gestartet und der Screen durch STRG-a d in den Background geschickt. Logoff von dem System. Danach logon zu dem System von einem anderen Client (z.b. nun von zu Hause) und mit screen -r den Screen wieder ffnen und die Aktivitt fortsetzen. Will man gleich zum Startzeitpunkt des Systems ein Programm per Screen starten fgt man folgende Zeile in die /etc/crontab ein und kann dann mit den vorgenannten Befehlen nachtrglich zu der Screensession verbinden:@reboot root /usr/bin/screen -dmS "" Man springt auf einem System zwecks Problemanalyse in diversen Verzeichnissen rum und mchte diese schnell wechseln und Ausgaben aus den Verzeichnissen schnell vergleichen. Nach Erstellung von einem Screen durch Eingabe von screen werden weitere Screens durch STRG-a c erstellt und mit STRG-a n und STRG-a p durch die Screens gewechselt. Benutzt man eine graphische Oberflche, kann man verschiedene Fenster parallel ffnen und alle o.g. Einsatzflle abdecken. Allerdings ist eine GUI Overkill bei kleinen kurzen administrativen Ttigkeiten und sogar gefhrlich, wenn man auf Server im Internet per X zugreift - und das noch als root. Ein Server im unsicheren Internet darf aus SecurityGrnden kein laufendes X haben! D.h. smtliche Administration muss in der Befehlszeile per ssh durchgefhrt werden. Auch gehen viele Dinge in der Befehlszeile einfach schneller als sich durch ein GUI Menu zu clicken. screen ist also ein virtueller ConsolenVervielfltiger, d.h. das Tool erlaubt, beliebig viele Konsolen ber eine ssh Session aufzubauen und unter diesen sehr schnell zu wechseln.Ausserdem sind solche screen Sessions auch von weiteren ssh Sessions zugreifbar, d.h. mehrere Personen knnen sich eine Konsole teilen. Nachdem man per 'screen' eine neue Session aufgebaut hat stehen in dieser Session diverse 'screen Befehle' zur Verfgung. Dazu gehren einen weiteren Screen an der aktuellen Session zu erzeugen STRG-a C Screens zu wechseln und auszuwhlen STRG-a n und STRG-a p, STRG-a 0, STRG-a 1 usw, -A " Screens zu beenden STRG-a k Screens zu detachen, also im Hintergrund weiterlaufen zu lassen STRG-a d Screen Protokollierung anzuschalten (damit man nachvollziehen kann was man so auf einem remoten System getrieben hat ...) STRG-a H Benachrichtigung bei Inaktivitt oder auch Aktivitt in einem Screen steuern STRG-a M Die Steuerung ist relativ einfach: Man drckt Strg, dann den Buchstaben a und dann den eigentlichen Befehlsshortcut ein. Die folgende Tabelle enthlt alle Shortcuts, die ich fr sinnvoll erachte. Es gibt noch viele weitere, die man per man screen oder einfach googeln nach screen linux finden kann oder einfach durch Eingabe von STRG-a ? in einem screen Fenster. Parameter Funktion Kommentar Erzeugen einer neuen Screen Session kein Parameter -ls Listet wie ls die bestehenden Screen Sessions -x ffnet eine Verbindung zu einer schon offenen Session Damit knnen sich mehrere Personen eine Session gemeinsam teilen und ansehen -r ffnet eine Verbindung zu einer nicht verbundenen Session Damit kann man Sessions wieder von anderen Clienst aufnehmen. -L Einschalten des LogModus Shortcuts in screen Fenstern STRG-a heisst immer gleichzeitiges drcken von STRG und dem Buchstaben a gefolgt von dem eigentlichen Shortcut Befehl Funktion Kommentar STRG-a " Anzeige aller Screens STRG-a A Vergabe eines Namens fr ein Fenster (Anzeige bzw Auswahl mit C-a ") STRG-a 0 Anzeige von Screen Nr 0 0-9 mglich STRG-a A ndern des Screennamens STRG-a c ffnen eines neuen Screens (c = create) STRG-a d Schliessen eines Screens in den Background (d = detach) STRG-a k Schliessen eines Screens (k = kill) STRG-a H Screen hardcopy in Datei screenlog ein/ausschalten STRG-a n Nchsten Screen ffnen (n = next) STRG-a p Vorherigen Screen ffnen (p = previous) STRG-a ? Anzeigen der Keybindings STRG-a * Anzeigen der momentanen Screens in der Session Page 5 Details Kategorie: General Erstellt: 20. Oktober 2011 Zuletzt aktualisiert: 23. Oktober 2022 Zugriffe: 4558 Regelmig gibt es neue Releases bei Linux Distributionen. Das bedeutet man installatiert parallel zur altenRelease die neue Release und zieht dann die Benutzerdaten und Systemkonfigurationen vom lteren zum neueren Release um. Das dumme ist, dass man im alten Release Softwarepakete installiert hat, die standardmig im neuen release nicht installiert sind da man sie mit dem jeweiligen Paketmanager manuell nachinstalliert hat. Dabei gibt es i.d.R. zwei Typen: Einmal Software, die man zu

lieben und schtzen gelernt hat und auch weiterhin im neuen Release benutzen will und irgendwelche Installationen um mal eben was auszuprobieren. Die Distributionen bieten Mglichkeiten an, diePakete der alten Release automatisch im neuen Release einzuspielen, nur leider wird dann auch smtliches Test- und Spielpaket wieder installiert. Genaugenommen muss man die rpm Listen des alten Releases und des neuen Releases vergleichen und eine List erstellen von den Paketetn, die im neuen Release nicht gefunden werden. Danach kann man die (hoffentlich relativ kurze) Liste durchgehen und die wichtigen Pakete identifizieren und dann manuell nachinstallieren. Um mir das zu vereinfachen habe ich ein kleines Python Script geschrieben, welches zwei rpm Listings vergleicht und die fehlenden Pakete in einer Liste ausgibt. Eingabe fr das Script sind sortierte rpm Listings des alten und des neuen Releases. Ausgabe ist die Liste der Pakete, die im alten Release vorhanden sind aber im neuen Release fehlen. Bedienungsanleitung: 1) Ausfhren des folgenden Befehls auf dem alten Release rpm -qa | sort > release_old 2) Ausfhren des folgenden Befehls auf dem neuen Release rpm -qa | sort > release_new 3) Aufruf des Vergleichsscripts mit einer Liste von Paketen, die ignoriert werden sollen compareDistroRPMs release_old release_new "(lib|yast2|sax2|bundle)" Beim letzten Parameter fr das Script ist zu bercksichtigen, dass es ein regulrer Ausdruck sein muss. Faustregel: Die Klammern lassen und mit | getrennt Teile des Paketnamens angeben, die ausgefiltert werden sollen, weil es Pakate sind, die in der Ausgabeliste zu ignorieren sin. Beispiel: Um Pakete auszunehmen "(lib|yast2|sax2|bundle|game)" als dritten Parameter benutzen. LVM stands for Logical Volume Manager, a mechanism that provides an alternative method of managing storage systems than the traditional partition-based one. In LVM, instead of creating partitions, you create logical volumes, and then you can just as easily mount those volumes in your filesystem as you'd a disk partition.One exception to the previous statement is that you can not use logical volumes for /boot. That is because GRUB (the most common bootloader for Linux) can't read from logical volumes. The well-known alternative to GRUB, systemd-boot on the other hand reads only vfat filesystems, so that's not going to work either.Components of LVMThere are three main components to LVM:Physical VolumesVolume GroupsLogical VolumesThe main advantage of LVM is how easy it is to resize a logical volume or volume group. It abstracts away all the ugly parts (partitions, raw disks) and leaves us with a central storage pool to work with.If you've ever experienced the horror of partition resizing, you'd wanna use LVM. LVM InstallationInstall Ubuntu with encrypted LVM using your main hard drive (assumed /dev/sda). Layer 1: Hard Drives, Partitions, Physical VolumesHard DrivesStart with disk scan to review available disks:sudo lvmdiskscanPartitionsIf your secondary hard drive (/dev/sdb) is blank, create a full sized partition on it (/dev/sdb1). Do another lvmdiskscan and you'll see the /dev/sdb1 partition listed as well (if it wasn't already there before). Physical VolumesUsing pvscan you'll see all your physical volumes:sudo pvscanCreate a new physical volume for your /dev/sdb1 partition: sudo pvcreate /dev/sdb1root@gerbil:/home/dwebb# pvcreate /dev/sdb1WARNING: ext4 signature detected on /dev/sdb1 at offset 1080. Wipe it? [y/n]: y Wiping ext4 signature on /dev/sdb1. Physical volume "/dev/sdb1" successfully created.WARNING: If you wish to preserve the content of /dev/sdb1, type n when prompted to wipe the disk. Layer 2: Volume GroupsNow we need to add the new Physical Volume to the Volume Group. List your Volume Groups using vgscan: vgscanFound volume group "vgubuntu" using metadata type lvm2Add the new Physical Volume to the Volume Group using vgextend:vgextend vgubuntu /dev/sdb1Volume group "vgubuntu" successfully extendedNow let's check out the current status of the Volume Group: vgdisplay vgubuntu---Volume group ---VG Name vgubuntuSystem ID Format lvm2Metadata Areas 2Metadata Sequence No 4VG Access read/writeVG Status resizableMAX LV 0Cur LV 2Open LV 2Max PV 0Cur PV 2Act PV 2VG Size 267.05 GiBPE Size 4.00 MiBTotal PE 68366Alloc PE / Size 30207 /