mdadm Quick Reference

Common mdadm commands I found a really great, if somewhat dated, article at . This is mainly a copy of that article, updated for what I do under Debian.

Query Array or Member

```
mdadm --examine /dev/sda # get RAID information on sda if it is an array member
mdadm --query /dev/md0 # get information on a RAID array, or member if this is a disk
mdadm --detail /dev/md0 # gives more information about array, including information about each individual member
```

Generate mdadm.conf

First, you have to determine where mdadm.conf is. On CentOS, it is located at /etc/mdadm.conf, while on Debian it is located in /etc/mdadm/mdadm.conf.

The basic way to create a new mdadm.conf is to use mdadm's scan command, which will find existing md's and send them to STDOUT. I also like to add an e-mail user for warnings.

```
cp /etc/mdadm.conf /etc/mdadm.conf.save
mdadm --verbose --detail --scan > /etc/mdadm.conf
echo MAILADDR user1@dom1.com, user2@dom2.com >> /etc/mdadm.conf
```

Debian based systems have a mkconf command which creates a basic mdadm.conf with the MAILADDR built in (though you still have to edit it).

```
cp /etc/mdadm/mdadm.conf /etc/mdadm/mdadm.conf.save
/usr/share/mdadm/mkconf --generate > /etc/mdadm/mdadm.conf
```

Create RAID

```
mdadm --create /dev/md2 --assume-clean --raid-devices=3 --spare-devices=0 --level=5 --run /dev/sd[cde]1
```

Note: –assume-clean is for SSD's which are **known** to be clean (no bad blocks). It will not do a resync on the underlying system, 'assuming they are clean', which will reduce the wear on the block devices. However, it is not recommended by the authors of mdadm (see man page)

Note: see Setting GRUB on a drive if you are setting up a bootable RAID-1

Remove disk from RAID

```
mdadm --fail /dev/md0 /dev/sda1
mdadm --remove /dev/md0 /dev/sda1
```

Copy the partition structure (when replacing a failed drive)

```
sfdisk -d /dev/sda | sfdisk /dev/sdb
mdadm --zero-superblock /dev/sdb
```

Add a disk to a RAID array (to replace a removed failed drive)

```
mdadm --add /dev/md0 /dev/sdf1
```

Check RAID status

```
cat /proc/mdstat
mdadm --detail /dev/md0
```

Reassemble a group of RAID disks

This works to move an assembly from one physical machine to another.

```
mdadm --assemble /dev/md0 /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
```

Convert a RAID 1 array to RAID 5

(follow the steps to add a disk after running this command) The most secure way of converting a RAID-1 to a RAID-5 is to create two degraded arrays, then copy the data. Note: you will be running your system with two degraded RAID arrays and losing any single drive can result in a total loss of data, so either back up or prepare for a loss of data.

Example shows md1 (the RAID-1) and md5 (the RAID-5 we will convert to). Note to people unfamiliar with software RAID, there is nothing special about me choosing md1 and md5; I just chose them to make the example easier to follow. On my system, /dev/md1 was the RAID-1, and I created /dev/md0 as the RAID-5.

We assume md1 is composed of /dev/sda and /dev/sdb, and we are wanting md5 to eventually consist of /dev/sdc, /dev/sda and /dev/sdb. One of the drives from md1 will be removed from the RAID-1 (md1) and used to create the RAID-5 (md5, degraded). It doesn't matter which, but I'll choose /dev/sdb.

I have not actually done this yet, but intend to as soon as I have some data backed up.

```
# remove /dev/sdb from md1 (the RAID-1)
mdadm /dev/md1 --fail /dev/sdb
mdadm /dev/md1 --remove /dev/sdb
# clean up disk /dev/sdb
mdadm --zero-superblock /dev/sdb
dd if=/dev/zero of=/dev/sdb bs=512 count=1
# and, create the RAID 5 with one disk missing
mdadm --create /dev/md5 --level=5 --raid-devices=3 /dev/sdb /dev/sdc missing
# watch /proc/mdstat to wait for /dev/md5 to be built
# following assumes /dev/md1 was the Physical Volume for an LVM group
# named virtuals. Skip this if you are not working with LVM.
# Simply mount both RAID sets and copy (cp -axv) all files over
#
# mark md5 as a physical volume for LVM
pvcreate /dev/md5
# Add it to volume group 'virtuals'
vgextend /dev/virtuals /dev/md5
# now, move all data off the old RAID-1 to the RAID-5. This can take a
while.
# In the test system (two quad core xeon's with 2G free RAM) it took almost
an
# hour to move 150G of data
pvmove -v /dev/md1
# and, when that is done, remove the RAID-1 from the volume group
vgreduce virtuals /dev/md1
# flag md1 as not a PV
pvremove /dev/md1
# at this point, md1 is a degraded RAID-1 not being used by anything, so
destroy the RAID set
mdadm --stop /dev/md1
mdadm --remove /dev/md1
# clean up and add /dev/sda to md5
mdadm --zero-superblock /dev/sda
dd if=/dev/zero of=/dev/sda bs=512 count=1
mdadm /dev/md5 --add /dev/sda
# you should now see /dev/md5 rebuilding in /proc/mdstat. I'd recommend you
# create a new mdadm.conf (see above)
```

This is what was in the original post. It worked on mdadm v0.9, but appears not to work now

this is no longer a viable option. Upgrades to mdadm result in this being # a high risk of losing all data # I found a description of the problem in the article # http://www.arkf.net/blog/?p=47

```
mdadm --create /dev/md0 --level=5 -n 2 /dev/sda1 /dev/sdb1
```

Add a disk to an existing RAID and resize the filesystem

```
mdadm --add /dev/md0 /dev/sdg1
mdadm --grow /dev/md0 -n 5
```

```
e2fsck -f /dev/md0
resize2fs /dev/md0
e2fsck -f /dev/md0
```

Replace all disks in an array with larger drives and resize

For each drive in the existing array

```
mdadm --fail /dev/md0 /dev/sda1
mdadm --remove /dev/md0 /dev/sda1
# physically replace the drive
mdadm --add /dev/md0 /dev/sda1
# now, wait until md0 is rebuilt.
# this can literally take days
```

End of the For

All drives have been replaced and sync'd, but they still use the original size. Issue the following command to use all available space:

```
mdadm --grow /dev/md0  --size=max
```

Do not forget to resized the file system which sits on the raid set:

```
# for ext2/3/4
e2fsck -f /dev/md0 && resize2fs /dev/md0 && e2fsck -f /dev/md0
# for lvm pv
pvresize /dev/md0
# for ntfs
ntfsresize /dev/md0
# note, most likely ntfs is NOT exported as a single partition. In the case
# of a Xen hvm machine, it is a "disk device" so you will need to resize the
# partition itself, then resize ntfs.
```

Stop and remove the RAID device

```
mdadm --stop /dev/md0
mdadm --remove /dev/md0
```

Destroy an existing array

```
mdadm --manage /dev/md2 --fail /dev/sd[cde]1
mdadm --manage /dev/md2 --remove /dev/sd[cde]1
mdadm --manage /dev/md2 --stop
mdadm --zero-superblock /dev/sd[cde]1
```

Re-use a disk from another RAID set

If a disk has been used in another RAID set, it has a superblock on it that really, really can cause problems. Simply clear the superblock to re-use it

```
mdadm --zero-superblock /dev/sdb
```

You might also want to delete the partition table and MBR from a disk, in which case you can issue this command

```
dd if=/dev/zero of=/dev/hda bs=512 count=1
Speed up a sync (after drive replacement)
cat /proc/sys/dev/raid/speed_limit_max
#200000
cat /proc/sys/dev/raid/speed_limit_min
# 1000
```

This means you are running a minimum of 1000 KB/sec/disk and a maximum of 200,000. To speed it up:

```
echo 50000 >/proc/sys/dev/raid/speed_limit_min
```

which will set the minimum to 50,000 KB/sec/disk (ie, 50 times greater). Expect your processor and disk subsystem to be a lot slower (this is kind of like messing with the nice value of your processes).

Rename an existing array

Had a situation where re-using an array resulted in Debian renaming it as md127, which really upset a lot of stuff. To rename it, simply stop the array, then re-assemble it.

```
mdadm --stop /dev/md127
mdadm -A /dev/md0 -m127 --update=super-minor /dev/sd[bcd]
```

This stops the array as /dev/md127 and then reassembles it as /dev/md0. The reassembly looks for devices which have an existing minor number of 127, not 0 (-m127), and then updates the minors in the superblocks to the new number. I included the original members (sdb, sdc and sdd) as /dev/sd[bcd]

Note: If you have LVM2 running on top of your RAID set, you must do the following for each LV listed.

```
for lv in `ls /dev/vgname/`
do
   lvchange -an $lv
done
vgchange -an vgname
```

change vgname to the actual name of you volume group. You must do this before you can stop the md

Converting from one RAID level to another

You can convert one raid level to another with the –grow parameter. NOTE: I have not done this yet, so am basing it on an older document at http://neil.brown.name/blog/20090817000931

Basically, you use the grow and include the level/number of disks/whatever, it appears you can simply perform the following. This assumes md0 is a 3 disk RAID5, and we are adding a new disk, sde, so we can convert to a RAID-6.

```
mdadm --add /dev/md0 /dev/sde
mdadm --grow /dev/md0 --level=6 --raid-devices=4 --backup-
file=/nfs/media/tmp/md3.backup
```

The backup-file appears to be required, or was in 2010, though the documentation says it is not if you have a spare disk. It should be on a very fast drive as appearantly every sector in the whole array gets copied. For example, if the above raid set was full of 1T drives, it would write 1 tera's to the backup file (one block at a time, it would not grow past one block, normally 512k).

You can also, appearantly, grow and change the layout in one –grow command. Assume we have a 3 disk RAID-5, and we are adding /dev/sde & /dev/sdf to it, and we want to convert it to a RAID 6 and add the space of the other disk.

```
mdadm --add /dev/md0 /dev/sd[ef]
mdadm --grow /dev/md0 --size max --level=6 --raid-devices=5 --backup-
file=/nfs/media/tmp/md3.backup
```

Note: This can take a very long time. Days is not abnormal. See https://forums.gentoo.org/viewtopic-t-810498-start-0.html for more info.

Links

- [http://www.ducea.com/2006/06/25/increase-the-speed-of-linux-software-raid-reconstruction/]
- [http://www.excaliburtech.net/archives/19]
- [http://neil.brown.name/blog/20090817000931]
- [https://forums.gentoo.org/viewtopic-t-810498-start-0.html]

From:
https://wiki.linuxservertech.com/ - **Unix Server Tech Knowledge Base**

Permanent link:
https://wiki.linuxservertech.com/quickreference/mdadm

Last update: **2026/04/14 01:50**