By Falko Timme
Published: 2008-12-10 17:26

How To Resize RAID Partitions (Shrink & Grow) (Software RAID)

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 11/24/2008

This article describes how you can shrink and grow existing software RAID partitions. I have tested this with **non-LVM** RAID1 partitions that use ext3 as the file system. I will describe this procedure for an intact RAID array and also a degraded RAID array.

If you use LVM on your RAID partitions, the procedure will be different, so do not use this tutorial in this case!

I do not issue any guarantee that this will work for you!

1 Preliminary Note

A few days ago I found out that one of my servers had a degraded RAID1 array (*/dev/md2*, made up of */dev/sda3* and */dev/sdb3*; */dev/sda3* had failed, */dev/sdb3* was still active):

```
server1:~# cat /proc/mdstat
Personalities : [raid1]
md2 : active raid1 sdb3[1]
      4594496 blocks [2/1] [_U]

md1 : active raid1 sda2[0] sdb2[1]
      497920 blocks [2/2] [UU]

md0 : active raid1 sda1[0] sdb1[1]
      144448 blocks [2/2] [UU]
```

```
unused devices: <none>
server1:~#
```

I tried to fix it (using this **tutorial**), but unfortunately at the end of the sync process (with 99.9% complete), the sync stopped and started over again. As I found out, this happened because there were some defect sectors at the end of the (working) partition */dev/sdb3* - this was in */var/log/kern.log*:

```
Nov 22 18:51:06 server1 kernel: sdb: Current: sense key: Aborted Command
Nov 22 18:51:06 server1 kernel: end_request: I/O error, dev sdb, sector 1465142856
```

So this was the worst case that could happen - */dev/sda* dead and */dev/sdb* about to die. To fix this, I imagined I could shrink */dev/md2* so that it leaves out the broken sectors at the end of */dev/sdb3*, then add the new */dev/sda3* (from the replaced hard drive) to */dev/md2*, let the sync finish, remove */dev/sdb3* from the array and replace */dev/sdb* with a new hard drive, add the new */dev/sdb3* to */dev/md2*, and grow */dev/md2* again.

This is one of the use cases for the following procedures (I will describe the process for an intact array and a degraded array).

Please note that */dev/md2* is my system partition (mount point */*), so I had to use a rescue system (e.g. **Knoppix Live-CD**) to resize the array. If the array you want to resize is not your system partition, you probably don't need to boot into a rescue system; but in either case, make sure that the array is unmounted!

2 Intact Array

I will describe how to resize the array */dev/md2*, made up of */dev/sda3* and */dev/sdb3*.

2.1 Shrinking An Intact Array

Boot into your rescue system and activate all needed modules:

```
modprobe md

modprobe linear

modprobe multipath
```

```
modprobe raid0

modprobe raid1

modprobe raid5

modprobe raid6

modprobe raid10
```

Then activate your RAID arrays:

```
cp /etc/mdadm/mdadm.conf /etc/mdadm/mdadm.conf_orig

mdadm --examine --scan >> /etc/mdadm/mdadm.conf
```

```
mdadm -A --scan
```

Run

```
e2fsck -f /dev/md2
```

to check the file system.

/dev/md2 has a size of 40GB; I want to shrink it to 30GB. First we have to shrink the file system with *resize2fs*; to make sure that the file system fits into the 30GB, we make it a little bit smaller (25GB) so we have a little security margin, shrink */dev/md2* to 30GB, and the resize the file system (again with *resize2fs*) to the max. possible value:

```
resize2fs /dev/md2 25G
```

Now we shrink */dev/md2* to 30GB. The *--size* value must be in KiBytes (30 x 1024 x 1024 = 31457280); make sure it can be divided by 64:

```
mdadm --grow /dev/md2 --size=31457280
```

Next we grow the file system to the largest possible value (if you don't specify a size, *resize2fs* will use the largest possible value)...

```
resize2fs /dev/md2
```

... and run a file system check again:

```
e2fsck -f /dev/md2
```

That's it - you can now boot into the normal system again.

2.2 Growing An Intact Array

Boot into your rescue system and activate all needed modules:

```
modprobe md

modprobe linear

modprobe multipath

modprobe raid0

modprobe raid1

modprobe raid5
```

```
modprobe raid6

modprobe raid10
```

Then activate your RAID arrays:

```
cp /etc/mdadm/mdadm.conf /etc/mdadm/mdadm.conf_orig

mdadm --examine --scan >> /etc/mdadm/mdadm.conf
```

```
mdadm -A --scan
```

Now we can grow `/dev/md2` as follows:

```
mdadm --grow /dev/md2 --size=max
```

`--size=max` means the largest possible value. You can as well specify a size in KiBytes (see previous chapter).

Then we run a file system check...

```
e2fsck -f /dev/md2
```

..., resize the file system...

```
resize2fs /dev/md2
```

... and check the file system again:

```
e2fsck -f /dev/md2
```

Afterwards you can boot back into your normal system.

3 Degraded Array

I will describe how to resize the degraded array */dev/md2*, made up of */dev/sda3* and */dev/sdb3*, where */dev/sda3* has failed:

```
server1:~# cat /proc/mdstat
Personalities : [raid1]
md2 : active raid1 sdb3[1]
      4594496 blocks [2/1] [_U]

md1 : active raid1 sda2[0] sdb2[1]
      497920 blocks [2/2] [UU]

md0 : active raid1 sda1[0] sdb1[1]
      144448 blocks [2/2] [UU]

unused devices: <none>
server1:~#
```

3.1 Shrinking A Degraded Array

Before we boot into the rescue system, we must make sure that */dev/sda3* is really removed from the array:

```
mdadm --manage /dev/md2 --fail /dev/sda3

mdadm --manage /dev/md2 --remove /dev/sda3
```

Then we overwrite the superblock on */dev/sda3* (this is very important - if you forget this, the system might now boot anymore after the resize!):

```
mdadm --zero-superblock /dev/sda3
```

Boot into your rescue system and activate all needed modules:

```
modprobe md

modprobe linear

modprobe multipath

modprobe raid0

modprobe raid1

modprobe raid5

modprobe raid6

modprobe raid10
```

Then activate your RAID arrays:

```
cp /etc/mdadm/mdadm.conf /etc/mdadm/mdadm.conf_orig

mdadm --examine --scan >> /etc/mdadm/mdadm.conf
```

```
mdadm -A --scan
```

Run

```
e2fsck -f /dev/md2
```

to check the file system.

/dev/md2 has a size of 40GB; I want to shrink it to 30GB. First we have to shrink the file system with *resize2fs*; to make sure that the file system fits into the 30GB, we make it a little bit smaller (25GB) so we have a little security margin, shrink */dev/md2* to 30GB, and the resize the file system (again with *resize2fs*) to the max. possible value:

```
resize2fs /dev/md2 25G
```

Now we shrink */dev/md2* to 30GB. The *--size* value must be in KiBytes (30 x 1024 x 1024 = 31457280); make sure it can be divided by 64:

```
mdadm --grow /dev/md2 --size=31457280
```

Next we grow the file system to the largest possible value (if you don't specify a size, *resize2fs* will use the largest possible value)...

```
resize2fs /dev/md2
```

... and run a file system check again:

```
e2fsck -f /dev/md2
```

Then boot into the normal system again and run the following two commands to add */dev/sda3* back to the array */dev/md2*:

```
mdadm --zero-superblock /dev/sda3

mdadm -a /dev/md2 /dev/sda3
```

Take a look at

```
cat /proc/mdstat
```

and you should see that `/dev/sdb3` and `/dev/sda3` are now being synced.

3.2 Growing A Degraded Array

Before we boot into the rescue system, we must make sure that `/dev/sda3` is really removed from the array:

```
mdadm --manage /dev/md2 --fail /dev/sda3

mdadm --manage /dev/md2 --remove /dev/sda3
```

Then we overwrite the superblock on `/dev/sda3` (this is very important - if you forget this, the system might now boot anymore after the resize!):

```
mdadm --zero-superblock /dev/sda3
```

Boot into your rescue system and activate all needed modules:

```
modprobe md

modprobe linear

modprobe multipath

modprobe raid0

modprobe raid1
```

```
modprobe raid5

modprobe raid6

modprobe raid10
```

Then activate your RAID arrays:

```
cp /etc/mdadm/mdadm.conf /etc/mdadm/mdadm.conf_orig

mdadm --examine --scan >> /etc/mdadm/mdadm.conf
```

```
mdadm -A --scan
```

Now we can grow `/dev/md2` as follows:

```
mdadm --grow /dev/md2 --size=max
```

`--size=max` means the largest possible value. You can as well specify a size in KiBytes (see previous chapter).

Then we run a file system check...

```
e2fsck -f /dev/md2
```

..., resize the file system...

```
resize2fs /dev/md2
```

... and check the file system again:

```
e2fsck -f /dev/md2
```

Then boot into the normal system again and run the following two commands to add */dev/sda3* back to the array */dev/md2*:

```
mdadm --zero-superblock /dev/sda3

mdadm -a /dev/md2 /dev/sda3
```

Take a look at

```
cat /proc/mdstat
```

and you should see that */dev/sdb3* and */dev/sda3* are now being synced.

4 Links

- Knoppix: **http://www.knoppix.net/**