

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

INTRODUCTION

This article is about resizing ext3 partitions without losing data. It shows how to shrink and enlarge existing ext3 partitions and how to merge two ext3 partitions. This can be quite useful if you do not use LVM and you realize that your existing partitioning does not meet your actual needs anymore. There are many ways of achieving this goal but this is the way I take. I do not issue any guarantee that this will work for you!

I run all the commands in this tutorial as the root user, so make sure you're logged in as root. If you are on a Ubuntu system, you can become root like this:

sudo su

I have tested this on a Ubuntu Edgy Eft desktop system that has all files in one large partition (around 10 GB, device /dev/sda1). The partitioning looks like this:

df -h

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	9.5G	4.1G	4.9G	46%	/
varrun	94M	132K	94M	1%	/var/run
varlock	94M	0	94M	0%	/var/lock
udev	10M	52K	10M	1%	/dev
devshm	94M	0	94M	0%	/dev/shm
lrm	94M	18M	77M	19%	/lib/modules/2.6.17-10-generic/volatile

The partition that is to be resized must be unmounted when we do the resizing; obviously this is not possible if this is the partition that holds all important system files like in this example. Therefore we download a Live Linux-CD such as Knoppix from which we boot later on (if you have physical access to the system). If it is a remote system that you don't have physical access to, you need a rescue system on that system (a lot of hosting companies offer dedicated servers with rescue systems nowadays) that you can boot into (instead of Knoppix), and this rescue system must have the following tools: fdisk, umount, fsck, tune2fs, e2fsck, resize2fs.

If the partition that you want to resize doesn't hold any system files, such as /home partitions, partitions for backups, etc., you don't need a Knoppix Live-CD or a rescue system, because all steps can be run from the original system.

If you want to resize partitions on production systems, please back up your data before, because it is possible you lose all your data if you don't calculate the size of your new partition correctly (especially when shrinking a partition)! You have been warned! Tutorials about backups can be found here: http://www.howtoforge.com/taxonomy_menu/1/34

I'm going to resize /dev/sda1 in this tutorial. If your partition is named differently, please replace /dev/sda1 with your own device, e.g. /dev/hda5, /dev/sdb3, etc.

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

SHRINKING AN EXT3 PARTITION

This chapter is about shrinking an ext3 partition. I want to shrink /dev/sda1 in this example. First we gather some details on our original system:

df

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	9859036	4234908	5123304	46%	/
varrun	95480	132	95348	1%	/var/run
varlock	95480	0	95480	0%	/var/lock
udev	10240	52	10188	1%	/dev
devshm	95480	0	95480	0%	/dev/shm
lrm	95480	17580	77900	19%	/lib/modules/2.6.17-10-generic/volatile

df -B 4k

Filesystem	4K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	2464759	1058727	1280826	46%	/
varrun	23870	33	23837	1%	/var/run
varlock	23870	0	23870	0%	/var/lock
udev	2560	13	2547	1%	/dev
devshm	23870	0	23870	0%	/dev/shm
lrm	23870	4395	19475	19%	/lib/modules/2.6.17-10-generic/volatile

df -h

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	9.5G	4.1G	4.9G	46%	/
varrun	94M	132K	94M	1%	/var/run
varlock	94M	0	94M	0%	/var/lock
udev	10M	52K	10M	1%	/dev
devshm	94M	0	94M	0%	/dev/shm
lrm	94M	18M	77M	19%	/lib/modules/2.6.17-10-generic/volatile

fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

```
Device Boot Start End Blocks Id System
/dev/sda1 * 1 1247 10016496 83 Linux
/dev/sda2 1248 1305 465885 5 Extended
/dev/sda5 1248 1305 465853+ 82 Linux swap / Solaris
```

fdisk -s /dev/sda1

10016496

Then we shut down the system and boot into our Knoppix Live-CD, or your rescue system. If the partition you want to resize doesn't hold any system files, you can do everything from the original system; the steps are the same, just omit booting into Knoppix/your rescue system.

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

shutdown -r now

After Knoppix has booted, open a terminal and become root by running

su

/dev/sda1 should be unmounted by default, but you can run

umount /dev/sda1

to go sure.

Then run

fsck -n /dev/sda1

The output looks like this:

```
fsck 1.38 (30-Jun-2005)
e2fsck 1.38 (30-Jun-2005)
/dev/sda1: clean, 159037/1254176 files, 1095299/2504124 blocks
```

Next we remove the journal from /dev/sda1, thus turning it into an ext2 partition:

tune2fs -O ^has_journal /dev/sda1

The output looks like this:

```
tune2fs 1.38 (30-Jun-2005)
```

Then run

e2fsck -f /dev/sda1

```
e2fsck 1.38 (30-Jun-2005)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/sda1: 164178/1254176 files (0.6% non-contiguous), 1051617/2504124 blocks
```

Now we resize our file system with `resize2fs`. `resize2fs` can resize ext2 file systems, but not ext3 file systems, that's why we had to turn /dev/sda1 to ext2. Currently, 4.1GB are used on /dev/sda1 (see the `df -h` output above), So it's safe to shrink it from 10GB to about 6GB (if you make it smaller than 4.1GB, you will lose data!). Therefore we run

resize2fs /dev/sda1 6000M

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

The output is as follows:

```
resize2fs 1.38 (30-Jun-2005)
Resizing the filesystem on /dev/sda1 to 1536000 (4k) blocks.
The filesystem on /dev/sda1 is now 1536000 blocks long.
```

Please take note of the amount of blocks (1536000) and their size (4k). We need that soon.

Now we delete our /dev/sda1 partition (don't be afraid, no data will be lost) and create a new, smaller one (but still big enough to hold our resized file system!). We can do this with fdisk:

fdisk /dev/sda

(Yes, it's /dev/sda, not /dev/sda1.)

```
The number of cylinders for this disk is set to 1305.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
 1) software that runs at boot time (e.g., old versions of LILO)
 2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)
```

Type m to get a list of all commands:

```
Command (m for help): m
Command action
a toggle a bootable flag
b edit bsd disklabel
c toggle the dos compatibility flag
d delete a partition
l list known partition types
m print this menu
n add a new partition
o create a new empty DOS partition table
p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)
```

Now we delete partition no. 1 (/dev/sda1):

```
Command (m for help): d
Partition number (1-5): 1
```

Next we create a new /dev/sda1 partition. It was a primary partition before, so we choose p again, and again it is our partition no. 1:

```
Command (m for help): n
Command action
l logical (5 or over)
```

Revised December 31, 2006

Page 1 of 14

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

p primary partition (1-4)

p

Partition number (1-4): 1

Now comes the crucial part - we are asked about the size of the new partition. The first cylinder is no problem, it is the one from the fdisk -l output at the beginning of this chapter (1).

First cylinder (1-1305, default 1): 1

But we don't have a value for the last cylinder of our new partition. Fortunately, we can specify the size in kilobytes (K), so we calculate the size like this:

We multiply the amount of blocks from the resize2fs output (1536000) by the size of a block (4k), and to go sure the partition is big enough, we add 3 to 5% to it (3% was enough for me, but if you want to go sure take 5%):

1536000 * 4k * 1.03 = 6328320k

So we prepend that value with a + sign and replace the small k with a capital one (K) and enter it:

Last cylinder or +size or +sizeM or +sizeK (1-1247, default 1247): +6328320K

Our original /dev/sda1 had the bootable flag (see the fdisk -l output from the beginning of this chapter), so we must add it to our new /dev/sda1 again:

Command (m for help): a

Partition number (1-5): 1

Now let's write our new partition table and exit fdisk:

Command (m for help): w

The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy.

The kernel still uses the old table.

The new table will be used at the next reboot.

Syncing disks.

Now we reboot the system, and again we boot into our Knoppix system (rescue system; original system if resized partition doesn't hold system files):

shutdown -r now

Become root again

su

and then run this:

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

```
fsck -n /dev/sda1
```

The output should look like this:

```
fsck 1.38 (30-Jun-2005)
e2fsck 1.38 (30-Jun-2005)
/dev/sda1: clean, 159036/765536 files, 1047239/1536000 blocks
```

Then we create the journal on our new /dev/sda1, thus turning it into an ext3 partition again:

```
tune2fs -j /dev/sda1
```

```
tune2fs 1.38 (30-Jun-2005)
Creating journal inode: done
```

This filesystem will be automatically checked every 30 mounts or 0 days, whichever comes first. Use tune2fs -c or -i to override.

Now we are done. Shut down the system and boot into the original system:

```
shutdown -r now
```

If everything goes well, the original system will boot up, and no data has been lost. Now we can gather some details about our new partitioning and compare them with the information we collected at the beginning of this chapter:

```
df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	6047868	4224140	1639408	73%	/
varrun	95480	132	95348	1%	/var/run
varlock	95480	0	95480	0%	/var/lock
udev	10240	52	10188	1%	/dev
devshm	95480	0	95480	0%	/dev/shm
lrm	95480	17580	77900	19%	/lib/modules/2.6.17-10-generic/volatile

```
df -B 4k
```

Filesystem	4K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	1511967	1056035	409852	73%	/
varrun	23870	33	23837	1%	/var/run
varlock	23870	0	23870	0%	/var/lock
udev	2560	13	2547	1%	/dev
devshm	23870	0	23870	0%	/dev/shm
lrm	23870	4395	19475	19%	/lib/modules/2.6.17-10-generic/volatile

```
df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	5.8G	4.1G	1.6G	73%	/
varrun	94M	132K	94M	1%	/var/run

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

varlock	94M	0	94M	0%	/var/lock
udev	10M	52K	10M	1%	/dev
devshm	94M	0	94M	0%	/dev/shm
lrn	94M	18M	77M	19%	/lib/modules/2.6.17-10-generic/volatile

fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	789	6337611	83	Linux
/dev/sda2		1248	1305	465885	5	Extended
/dev/sda5		1248	1305	465853+	82	Linux swap / Solaris

fdisk -s /dev/sda1

6337611

ENLARGING AN EXT3 PARTITION

In this example we have a /dev/sda1 partition with about 6GB of size, and right behind that partition we have about 4GB of unused space. We want to add those 4GB of unused space to our /dev/sda1 partition (this doesn't work if these 4GB don't come right behind our /dev/sda1 partition, but are elsewhere on the hard disk!).

First, we collect some details again about our current partitioning:

df

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	6047868	4224140	1639408	73%	/
varrun	95480	132	95348	1%	/var/run
varlock	95480	0	95480	0%	/var/lock
udev	10240	52	10188	1%	/dev
devshm	95480	0	95480	0%	/dev/shm
lrn	95480	17580	77900	19%	/lib/modules/2.6.17-10-generic/volatile

df -B 4k

Filesystem	4K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	1511967	1056035	409852	73%	/
varrun	23870	33	23837	1%	/var/run
varlock	23870	0	23870	0%	/var/lock
udev	2560	13	2547	1%	/dev
devshm	23870	0	23870	0%	/dev/shm
lrn	23870	4395	19475	19%	/lib/modules/2.6.17-10-generic/volatile

df -h

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	5.8G	4.1G	1.6G	73%	/

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

varrun	94M	132K	94M	1%	/var/run
varlock	94M	0	94M	0%	/var/lock
udev	10M	52K	10M	1%	/dev
devshm	94M	0	94M	0%	/dev/shm
lrm	94M	18M	77M	19%	/lib/modules/2.6.17-10-generic/volatile

fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

```
Device Boot Start End Blocks Id System
/dev/sda1 * 1 789 6337611 83 Linux
/dev/sda2 1248 1305 465885 5 Extended
/dev/sda5 1248 1305 465853+ 82 Linux swap / Solaris
```

fdisk -s /dev/sda1

6337611

Then we shut down the system and boot into our Knoppix Live-CD (or your rescue system) (if the partition you want to resize doesn't hold any system files, you can do everything from the original system; the steps are the same, just omit booting into Knoppix/your rescue system).

shutdown -r now

After Knoppix has booted, open a terminal and become root by running

su

/dev/sda1 should be unmounted by default, but you can run

umount /dev/sda1

to go sure.

Then run

fsck -n /dev/sda1

```
fsck 1.38 (30-Jun-2005)
e2fsck 1.38 (30-Jun-2005)
/dev/sda1: clean, 159036/765536 files, 1080014/1536000 blocks
```

Next we remove the journal from /dev/sda1, thus turning it into an ext2 partition:

tune2fs -O ^has_journal /dev/sda1

The output looks like this:

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

tune2fs 1.38 (30-Jun-2005)

Now we use fdisk to delete our current /dev/sda1 partition and create a bigger one (don't be afraid, no data will be lost):

fdisk /dev/sda

The number of cylinders for this disk is set to 1305.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)

Type m to get a list of all commands:

Command (m for help): m
Command action
a toggle a bootable flag
b edit bsd disklabel
c toggle the dos compatibility flag
d delete a partition
l list known partition types
m print this menu
n add a new partition
o create a new empty DOS partition table
p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)

Let's print out the partition table:

Command (m for help): p

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	789	6337611	83	Linux
/dev/sda2		1248	1305	465885	5	Extended
/dev/sda5		1248	1305	465853+	82	Linux swap / Solaris

Now we delete partition no. 1 (/dev/sda1):

Command (m for help): d
Partition number (1-5): 1

Next we create a new /dev/sda1 partition. It was a primary partition before, so we choose p again, and again it is our partition no. 1:

Revised December 31, 2006

Page 1 of 14

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

```
Command (m for help): n
Command action
l logical (5 or over)
p primary partition (1-4)
p
Partition number (1-4): 1
```

Now we must specify the first and the last cylinder of our new /dev/sda1 partition. We know the first cylinder, can take it from the fdisk -l output before:

```
First cylinder (1-1305, default 1): 1
```

Now fdisk tells us the highest possible cylinder of our new partition (1247 in this example), so we simply enter this number:

```
Last cylinder or +size or +sizeM or +sizeK (1-1247, default 1247): 1247
```

Let's print out our new partition table:

```
Command (m for help): p
```

```
Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
Device Boot Start End Blocks Id System
/dev/sda1 1 1247 10016496 83 Linux
/dev/sda2 1248 1305 465885 5 Extended
/dev/sda5 1248 1305 465853+ 82 Linux swap / Solaris
```

Our original /dev/sda1 had the bootable flag (see the fdisk -l output from the beginning of this chapter), so we must add it to our new /dev/sda1 again:

```
Command (m for help): a
Partition number (1-5): 1
```

Now let's write our new partition table and exit fdisk:

```
Command (m for help): w
The partition table has been altered!
```

```
Calling ioctl() to re-read partition table.
```

```
WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
```

Now we reboot the system, and again we boot into our Knoppix system (rescue system; original system if resized partition doesn't hold system files):

```
shutdown -r now
```

Revised December 31, 2006

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

Become root again

su

Then run

e2fsck -f /dev/sda1

Now we must resize the file system in our /dev/sda1 partition. If we don't specify a size for the `resize2fs` command, it will assume the biggest possible size so we don't have to calculate. So we run

resize2fs /dev/sda1

The output looks like this:

```
resize2fs 1.38 (30-Jun-2005)
Resizing the filesystem on /dev/sda1 to 2504124 (4k) blocks.
The filesystem on /dev/sda1 is now 2504124 blocks long.
```

Next we run

fsck -n /dev/sda1

```
fsck 1.38 (30-Jun-2005)
e2fsck 1.38 (30-Jun-2005)
/dev/sda1: clean, 159036/1254176 files, 1062544/2504124 blocks
```

and create the journal on /dev/sda1, thus turning it into an ext3 partition again:

tune2fs -j /dev/sda1

```
tune2fs 1.38 (30-Jun-2005)
Creating journal inode: done
This filesystem will be automatically checked every 30 mounts or
0 days, whichever comes first. Use tune2fs -c or -i to override.
```

Now we are done. Shut down the system and boot into the original system:

shutdown -r now

If everything goes well, the original system will boot up, and no data has been lost. Now we can gather some details about our new partitioning and compare them with the information we collected at the beginning of this chapter:

df

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	9859036	4224032	5234348	45%	/
varrun	95480	132	95348	1%	/var/run

Revised December 31, 2006

Page 1 of 14

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

varlock	95480	0	95480	0%	/var/lock
udev	10240	52	10188	1%	/dev
devshm	95480	0	95480	0%	/dev/shm
lrm	95480	17580	77900	19%	/lib/modules/2.6.17-10-generic/volatile

df -B 4k

Filesystem	4K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	2464759	1056008	1308587	45%	/
varrun	23870	33	23837	1%	/var/run
varlock	23870	0	23870	0%	/var/lock
udev	2560	13	2547	1%	/dev
devshm	23870	0	23870	0%	/dev/shm
lrm	23870	4395	19475	19%	/lib/modules/2.6.17-10-generic/volatile

df -h

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	9.5G	4.1G	5.0G	45%	/
varrun	94M	132K	94M	1%	/var/run
varlock	94M	0	94M	0%	/var/lock
udev	10M	52K	10M	1%	/dev
devshm	94M	0	94M	0%	/dev/shm
lrm	94M	18M	77M	19%	/lib/modules/2.6.17-10-generic/volatile

fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	1247	10016496	83	Linux
/dev/sda2		1248	1305	465885	5	Extended
/dev/sda5		1248	1305	465853+	82	Linux swap / Solaris

fdisk -s /dev/sda1

10016496

MERGE TWO EXT3 PARTITIONS

In this example I have my system partition /dev/sda1 again (about 6GB of size) which is followed directly by the partition /dev/sda3 (about 4GB of size) on the hard disk. /dev/sda3 is mounted to the /data directory and doesn't hold files needed by the Linux system, just user data. The current partitioning looks like this:

df -h

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	5.8G	4.1G	1.6G	73%	/
varrun	94M	132K	94M	1%	/var/run

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

varlock	94M	0	94M	0%	/var/lock
udev	10M	56K	10M	1%	/dev
devshm	94M	0	94M	0%	/dev/shm
lrm	94M	18M	77M	19%	/lib/modules/2.6.17-10-generic/volatile
/dev/sda3	3.5G	72M	3.3G	3%	/data

To merge /dev/sda1 and /dev/sda3, we have to delete /dev/sda3 and then enlarge /dev/sda1 as described in chapter 3. This means that all data on /dev/sda3 will be lost, so if you need it later on please back it up somewhere else and then restore it to the new and bigger /dev/sda1 afterwards!

Now we open /etc/fstab and remove the line for /dev/sda3 there if it exists:

vi /etc/fstab

The new file without /dev/sda3 could look like this:

```
# /etc/fstab: static file system information.
#
#
proc          /proc        proc defaults    0    0
# /dev/sda1
UUID=566fd9e9-098f-4aae-9908-51efe171d8ba /            ext3
defaults,errors=remount-ro 0    1
# /dev/sda5
UUID=82102b65-35db-469a-9532-03d619d8cffb none          swap  sw          0
0
/dev/hdc      /media/cdrom0  udf,iso9660 user,noauto  0    0
/dev/         /media/floppy0 auto  rw,user,noauto 0    0
```

Then we unmount /dev/sda3 and run fdisk to delete it. This can still be done on the original system as /dev/sda3 doesn't contain system files:

umount /dev/sda3

fdisk /dev/sda

(Yes, it's /dev/sda, not /dev/sda1.)

The number of cylinders for this disk is set to 1305.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)

Type m to get a list of all commands:

Command (m for help): m
Command action
a toggle a bootable flag

Revised December 31, 2006

Page 1 of 14

HOW TO RESIZE EXT3 PARTITIONS WITHOUT LOSING DATA

By Falko Timme

http://www.howtoforge.com/linux_resizing_ext3_partitions

```
b edit bsd disklabel
c toggle the dos compatibility flag
d delete a partition
l list known partition types
m print this menu
n add a new partition
o create a new empty DOS partition table
p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)
Now let's delete /dev/sda3:
Command (m for help): d
Partition number (1-5): 3
```

Afterwards we write the new partition table to the disk:

```
Command (m for help): w
The partition table has been altered!
```

Calling `ioctl()` to re-read partition table.

```
WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
```

Now we shut down the system:

```
shutdown -r now
```

and boot into our Knoppix Live-CD (or your rescue system). From here on the steps are identical to chapter 3, beginning with

```
su
umount /dev/sda1
```

so please refer to that chapter.