

Linux Disk Management

Ken Hess

You might have worked with Linux for years and never added an additional disk to your system or perhaps you were too frustrated by Linux's strange ways of dealing with disks to attempt it. In either case, here's your opportunity to work through the steps required in adding a new disk to your system. The process, once you have your disk physically installed into the system, is simple.

Most of us install a new Linux system with a series of mouse clicks or taps on the keyboard without regard for how the disk setup takes place. Desktop users generally accept the default installation with little thought. But, what happens when you run out of space on a disk or filesystem? Server system operators know the procedure well but perennial desktop jockeys might never have treaded the multiple disk waters. Well, it's time to dive in to the deep end.

Initial Setup

The first step is to install the new disk into your system. Find an open bay in your computer, set the drive's jumpers, plug the drive into system power and into a controller cable. Boot up your system and watch your BIOS for the new drive. Login to your system and su to root or use sudo and issue the following commands.

```
debian:~# sfdisk -l
```

```
Disk /dev/hda: 10402 cylinders, 16 heads, 63 sectors/track
Units = cylinders of 516096 bytes, blocks of 1024 bytes, counting from 0
```

Device	Boot	Start	End	#cyls	#blocks	Id	System
/dev/hda1		0	-	0	0	0	Empty
/dev/hda2		0	-	0	0	0	Empty
/dev/hda3		0	-	0	0	0	Empty
/dev/hda4		0	-	0	0	0	Empty

```
Disk /dev/sda: 652 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0
```

Device	Boot	Start	End	#cyls	#blocks	Id	System
/dev/sda1	*	0+	616	617-	4956021	83	Linux
/dev/sda2		617	651	35	281137+	5	Extended
/dev/sda3		0	-	0	0	0	Empty
/dev/sda4		0	-	0	0	0	Empty
/dev/sda5		617+	651	35-	281106	82	Linux swap / Solaris

I have two disks in the example system, /dev/sda (System SCSI disk) and /dev/hda (Empty IDE disk). The IDE disk is the new disk that we'll use for this example. Since you know the device name of the new disk (/dev/hda), it's easy to configure it.

```
debian:~# fdisk /dev/hda
```

```
The number of cylinders for this disk is set to 10402.
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)

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

Linux Disk Management

Ken Hess

You're now in the fdisk (fixed disk) utility, where you'll setup partitions on the new disk. For simplicity, we'll create one single partition on this disk. Please realize that you can split the disk into several partitions or "slices" as your needs dictate. It's time to create that partition.

Begin by telling fdisk that you want to create a new partition and then provide the partition number that you want to create (1). To use the entire disk, hit ENTER to select the default cylinder (1) and hit ENTER again to select the last cylinder for the end of the partition. Write the changes to disk (w). Once you've written the changes to disk, fdisk closes and returns you to the system prompt.

```
Command (m for help): n

Command action
  e   extended
  p   primary partition (1-4)
p

Partition number (1-4): 1

First cylinder (1-10402, default 1):

Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-10402, default 10402):

Using default value 10402

Command (m for help): p

Disk /dev/hda: 5368 MB, 5368709120 bytes
16 heads, 63 sectors/track, 10402 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes
Disk identifier: 0x365a714a

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1            1        10402    5242576+   83  Linux

Command (m for help): w

The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

debian:~#
```

You're now ready to create the filesystem. Because we used the entire disk, you only have to do this once.

```
debian:~# mkfs.ext3 /dev/hda1
mke2fs 1.41.3 (12-Oct-2008)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
327680 inodes, 1310644 blocks
65532 blocks (5.00%) reserved for the super user
```

Linux Disk Management

Ken Hess

```
First data block=0
Maximum filesystem blocks=1342177280
40 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736
```

```
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

This filesystem will be automatically checked every 24 mounts or 180 days, whichever comes first. Use `tune2fs -c` or `-i` to override.

```
debian:~#
```

Mounting the Disk

You've created the filesystem (ext3) on your new disk. You can't "see" or use the disk yet until it's mounted. Mounting a disk means to make its contents available to the system and its users by providing a mount point on which the disk will reside. To do this, you need to use an existing directory or create a new one onto which the disk will mount. Since this disk will provide space for backing up valuable files, create a directory named `/backup`.

```
debian:~# mkdir /backup
```

Mount the disk using the standard mount command syntax: `mount device_name mount_point`

```
debian:~# mount /dev/hda1 /backup
```

Congratulations, you've installed a new hard drive into your system, setup the disk in `fdisk`, created a mount point, mounted the drive onto the mount point and you're ready to start filling up the new space with backups. And, not a moment too soon as you can see.

```
debian:~# df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
<code>/dev/sda1</code>	4.7G	4.2G	314M	94%	<code>/</code>
<code>tmpfs</code>	253M	0	253M	0%	<code>/lib/init/rw</code>
<code>udev</code>	10M	588K	9.5M	6%	<code>/dev</code>
<code>tmpfs</code>	253M	0	253M	0%	<code>/dev/shm</code>
<code>/dev/hda1</code>	5.0G	139M	4.6G	3%	<code>/backup</code>

Are we done here? Yes and no. We're done as far as using this new disk for backups and moving off some non-essential files to free up some space. What happens if you reboot the system? You wouldn't see the new disk. You might think it disappeared or failed. It's still there, just like you left it—in manual mount mode.

And, that's OK, if you want the disk available to you on an occasional basis and would prefer to mount it only when needed. If you'd like to have it available on a persistent basis, you need to do a bit more work.

Linux Disk Management

Ken Hess

Creating a Persistent Mount

Danger, Will Robinson! You have to edit a system file (/etc/fstab) for this one but rather than bore you with vi the vi version of file editing, we're going to cheat. The fstab file has the following format:

```
file system    mount point    type    options    dump    pass
```

File system is /dev/hda1, mount point is /backup, type is ext3, options is defaults, dump is 0 and pass is 2. I'll save the explanations of these options for a future entry. To add that list of parameters to the /etc/fstab file, we'll use the echo command.

```
debian:~# echo "/dev/hda1    /backup    ext3    defaults    0    2" >> /etc/fstab
```

Now, your new disk will always mount automatically in read/write mode for you upon subsequent boot ups.

Certain aspects of Linux can challenge the saltiest (and most patient) among computer users. Adding new disks is a common sore point for users unfamiliar with the unusual methods used in Linux. Remember to be careful when editing system files (files in the /etc directory) and using fdisk, since both can cause irreparable (at worst) or time-consuming damage to your system.