**15. Устройства хранения**

В предыдущих главах мы познакомились с приемами работы с данными на уровне файлов. В этой главе мы будем рассматривать данные на уровне устройств. Linux обладает удивительными возможностями работы с устройствами хранения, такими как жесткие диски, сетевые хранилища или виртуальные устройства хранения, например RAID (redundant array of independent disks — избыточный массив из независимых дисков) и LVM (logical volume manager — диспетчер логических томов).

Однако поскольку эта книга не о системном администрировании, мы не будем пытаться охватить эту тему во всех подробностях, а всего лишь познакомимся с некоторыми понятиями и ключевыми командами, которые используются для управления устройствами хранения данных.

Для выполнения упражнений к этой главе нам понадобится флеш-диск (флешка), подключаемый к порту USB компьютера, диск CD-RW (для систем, оборудованных пишущим приводом CD-ROM) и такой раритет, как гибкий диск (опять же, если система оборудована этим устройством).

Мы познакомимся со следующими командами:

• mount — монтирует файловые системы.

• umount — размонтирует файловые системы.

• fdisk — инструмент для работы с таблицей разделов.

• fsck — проверяет и восстанавливает файловые системы.

• fdformat — форматирует гибкий диск.

• mkfs — создает файловые системы.

• dd — выполняет запись данных блоками непосредственно в устройство.

• genisoimage (mkisofs) — создает файл образа ISO 9660.

• wodim (cdrecord) — записывает данные на оптический носитель.

• md5sum — вычисляет контрольную сумму MD5.

**Монтирование и размонтирование устройств хранения**

Последние достижения Linux на настольных компьютерах сделали управление устройствами хранения чрезвычайно простым для обычных пользователей. Достаточно подключить устройство к компьютеру, и оно тут же готово к работе. Раньше (года этак до 2004-го) все необходимые операции требовалось выполнять вручную. В серверных системах эти операции по большей части все еще выполняются вручную, потому что серверы часто предъявляют особые требования к устройствам хранения и настройкам.

Первый шаг в управлении устройствами хранения — подключение самого устройства к дереву файловой системы. Этот процесс называется *монтированием* и позволяет устройству участвовать в работе операционной системы. Как рассказывалось в главе 2, Unix-подобные операционные системы, такие как Linux, поддерживают единое дерево файловой системы, к разным точкам которого подключаются дополнительные устройства. Этот подход отличается от используемого в MS-DOS и Windows, где каждому устройству соответствует отдельное дерево файлов и каталогов (например, *C:\*, *D:\* и т.д.).

В файле с именем */etc/fstab* перечисляются устройства (обычно разделы жесткого диска), монтируемые на этапе загрузки. Ниже приводится пример содержимого */etc/fstab* из системы Fedora 7:

LABEL=/12        /                     ext3    defaults        1 1

LABEL=/home      /home                 ext3    defaults        1 2

LABEL=/boot      /boot                 ext3    defaults        1 2

tmpfs            /dev/shm              tmpfs   defaults        0 0

devpts           /dev/pts              devpts  gid=5,mode=620  0 0

sysfs            /sys                  sysfs   defaults        0 0

proc             /proc                 proc    defaults        0 0

LABEL=SWAP-sda3  swap                  swap    defaults        0 0

Большинство файловых систем из перечисленных в приведенном примере являются виртуальными, и наше обсуждение к ним неприменимо. Наибольший интерес для нас в рамках исследования данной темы представляют первые три:

LABEL=/12        /                     ext3    defaults        1 1

LABEL=/home      /home                 ext3    defaults        1 2

LABEL=/boot      /boot                 ext3    defaults        1 2

Это разделы жесткого диска. Каждая строка включает шесть полей, описание которых приводится в табл. 15.1.

**Таблица 15.1. Поля в файле /etc/fstab**

|  |  |  |
| --- | --- | --- |
| **Поле** | **Содержит** | **Описание** |
| 1 | Устройство | Традиционно это поле содержит фактическое имя файла устройства, связанного с физическим устройством, такое как */dev/hda1* (первый раздел ведущего (master) устройства на первом канале IDE). Но учитывая, что в современных компьютерах может быть множество динамически подключаемых устройств (таких, как устройства USB), многие современные дистрибутивы Linux связывают устройства с текстовыми метками. Такая метка (записываемая в устройство хранения во время форматирования) читается операционной системой в момент подключения устройства. Благодаря этому становится неважным, с каким файлом устройства связано физическое устройство, оно в любом случае будет идентифицировано верно |
| 2 | Точка монтирования | Каталог в файловой системе, к которому подключается устройство |
| 3 | Тип файловой системы | Linux позволяет монтировать множество типов файловых систем. Наиболее близкой к Linux является файловая система ext3, но точно так же поддерживаются другие типы, такие как FAT16 (msdos), FAT32 (vfat), NTFS (ntfs), CD-ROM (iso9660) и пр. |
| 4 | Параметры | Файловые системы могут монтироваться с разными параметрами. Например, можно смонтировать файловую систему в режиме «только для чтения» или запретить выполнять какие-либо программы из нее (очень полезная мера предосторожности для съемных носителей) |
| 5 | Частота | Единственное число, определяющее, когда должно выполняться резервное копирование файловой системы командой dump |
| 6 | Порядок | Единственное число, определяющее, в каком порядке файловая система должна проверяться командой fsck |

**Просмотр списка смонтированных файловых систем**

Для монтирования файловых систем используется команда mount. Если ввести команду без аргументов, она выведет список файловых систем, смонтированных в настоящий момент:

[me@linuxbox ~]$ **mount**

/dev/sda2 on / type ext3 (rw)

proc on /proc type proc (rw)

sysfs on /sys type sysfs (rw)

devpts on /dev/pts type devpts (rw,gid=5,mode=620)

/dev/sda5 on /home type ext3 (rw)

/dev/sda1 on /boot type ext3 (rw)

tmpfs on /dev/shm type tmpfs (rw)

none on /proc/sys/fs/binfmt\_misc type binfmt\_misc (rw)

sunrpc on /var/lib/nfs/rpc\_pipefs type rpc\_pipefs (rw)

fusectl on /sys/fs/fuse/connections type fusectl (rw)

/dev/sdd1 on /media/disk type vfat (rw,nosuid,nodev,noatime, uhelper=hal,uid=500,utf8,shortname=lower)

twin4:/musicbox on /misc/musicbox type nfs4 (rw,addr=192.168.1.4)

Список имеет следующий формат: *устройство* on *точка\_монтирования* type *тип\_файловой\_системы* (*параметры*). Например, первая строка соответствует устройству */dev/sda2*, смонтированному как корневая файловая система типа ext3, доступная для чтения и записи (параметр rw). В конце списка можно заметить две интересные записи. Предпоследняя запись соответствует 2-гигабайтной SD-карте памяти в устройстве для чтения карт памяти, смонтированной в каталог */media/disk*, последняя запись соответствует сетевому приводу, смонтированному в каталог */misc/musicbox*.

Для первого эксперимента возьмем привод CD-ROM. Сначала посмотрим, что имеется в системе, перед тем как вставить компакт-диск:

[me@linuxbox ~]$ **mount**

/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)

proc on /proc type proc (rw)

sysfs on /sys type sysfs (rw)

devpts on /dev/pts type devpts (rw,gid=5,mode=620)

/dev/hda1 on /boot type ext3 (rw)

tmpfs on /dev/shm type tmpfs (rw)

none on /proc/sys/fs/binfmt\_misc type binfmt\_misc (rw)

sunrpc on /var/lib/nfs/rpc\_pipefs type rpc\_pipefs (rw)

Этот список получен в системе CentOS 5, где для создания корневой файловой системы используется диспетчер LVM. Подобно многим современным дистрибутивам Linux, эта система пытается автоматически монтировать компакт-диски. Вставив в привод компакт-диск, мы увидим следующее:

[me@linuxbox ~]$ **mount**

/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)

proc on /proc type proc (rw)

sysfs on /sys type sysfs (rw)

devpts on /dev/pts type devpts (rw,gid=5,mode=620)

/dev/hda1 on /boot type ext3 (rw)

tmpfs on /dev/shm type tmpfs (rw)

none on /proc/sys/fs/binfmt\_misc type binfmt\_misc (rw)

sunrpc on /var/lib/nfs/rpc\_pipefs type rpc\_pipefs (rw)

/dev/hdc on /media/live-1.0.10-8 type iso9660 (ro,noexec,nosuid,nodev,uid=500)

Это практически тот же список, с одной дополнительной записью. Последняя ­запись в списке сообщает, что компакт-диск в приводе CD-ROM (устройство */dev/hdc* в этой системе) смонтирован в каталог */media/live-1.0.10-8* и имеет файловую систему iso9660 (типичную для компакт-дисков). Обратите внимание на имя устройства. Когда вы будете проводить эксперимент в своей системе, очень вероятно, что имя устройства у вас будет отличаться.

**ВНИМАНИЕ**

В примерах, демонстрируемых ниже, особое внимание обращайте на фактические имена устройств в вашей системе и не используйте имена, приводящиеся в примерах здесь!

Также отметьте, что аудиодиск — это не то же самое, что CD-ROM. Аудиодиск не имеет файловой системы и потому не может быть смонтирован в общепринятом смысле.

Теперь, когда мы знаем имя устройства для привода CD-ROM, размонтируем диск и повторно смонтируем его в другой каталог в дереве файловой системы. Для этого необходимо получить права суперпользователя (способом, соответствующим вашей системе) и размонтировать диск командой umount:

[me@linuxbox ~]$ **su -**

Password:

[root@linuxbox ~]# **umount /dev/hdc**

Следующий шаг: создать новую точку монтирования диска. *Точка монтирования* — это самый обычный каталог где-то в дереве файловой системы. В таком каталоге нет ничего необычного. Он даже не должен быть пустым каталогом, правда, монтирование устройства в непустой каталог сделает его прежнее содержимое недоступным, пока устройство не будет размонтировано. Итак, создаем новый каталог:

[root@linuxbox ~]# **mkdir /mnt/cdrom**

И наконец, смонтируем CD-ROM в новую точку монтирования. Параметр -t позволяет указать тип файловой системы:

[root@linuxbox ~]# **mount -t iso9660 /dev/hdc /mnt/cdrom**

После этого можно исследовать содержимое компакт-диска в новой точке монтирования:

[root@linuxbox ~]# **cd /mnt/cdrom**

[root@linuxbox cdrom]# **ls**

Обратите внимание, что происходит при попытке размонтировать компакт-диск:

[root@linuxbox cdrom]# **umount /dev/hdc**

umount: /mnt/cdrom: device is busy

В чем причина? Устройство нельзя размонтировать, если оно используется каким-то пользователем или другим процессом. В данном случае мы изменили текущий рабочий каталог, перенеся его в точку монтирования компакт-диска, что и стало причиной занятости устройства. Эту проблему легко исправить, перенеся текущий рабочий каталог куда-нибудь в другое место за пределами точки монтирования:

[root@linuxbox cdrom]# **cd**

[root@linuxbox ~]# **umount /dev/hdc**

Теперь устройство было успешно размонтировано.

**Почему важно размонтировать устройства**

Если взглянуть на вывод команды free, показывающей статистику использования памяти, можно увидеть статистику с названием *buffers* (буферы). Компьютерные системы проектируются так, чтобы работать максимально быстро. Но медленные устройства препятствуют этому. Ярким примером служат принтеры. Даже самый быстрый принтер выглядит чрезвычайно медлительным по компьютерным стандартам. Компьютеры работали бы крайне медленно, если бы действительно были вынуждены ждать, пока принтер завершит печать страницы. В давние времена (когда персональные компьютеры еще не были многозадачными) это представляло настоящую проблему. При попытке распечатать электронную таблицу или текстовый документ компьютер мог стать недоступным до конца печати. Компьютер не мог посылать данные принтеру быстрее, чем тот мог их обработать, а принтеры не могли работать быстрее, потому что не могли быстро печатать. Эта проблема была решена созданием *буфера печати*, устройства, содержащего некоторый объем ОЗУ и находящегося между компьютером и принтером. При наличии буфера печати компьютер мог послать данные в буфер печати, который сохранял их в быстрой памяти ОЗУ, и компьютер возвращался к работе, не дожидаясь конца печати. В то же время буфер печати мог передавать данные принтеру из своей памяти со скоростью, приемлемой для принтера.

Идея буферизации широко используется для увеличения производительности компьютеров — необходимость работы с медленными устройствами не должна ухудшать производительность системы. Операционные системы хранят данные, прочитанные с устройства и предназначенные для записи в устройство, так долго, насколько это возможно, и используют их, прежде чем фактически обратиться к медленному устройству. В системе Linux, например, можно заметить, что при продолжительной работе она заполняет всю память. Это не означает, что Linux «использует» всю память, это означает лишь то, что Linux использует в своих интересах всю доступную память и буферизует как можно больше данных.

Буферизация позволяет очень быстро выполнять запись в устройства хранения, потому что запись в физическое устройство откладывается «на потом». Данные, предназначенные для устройства, накапливаются в памяти. Время от времени операционная система записывает эти данные в физическое устройство.

Размонтирование устройства влечет за собой запись всех оставшихся данных в это устройство, чтобы его можно было безопасно извлечь. Если носитель извлечь, не выполнив размонтирование, есть вероятность, что не все данные, предназначенные для устройства, будут записаны в него. Иногда эти данные могут включать жизненно важные обновления каталогов, отсутствие которых может привести к повреждению файловой системы — одной из самых больших неприятностей, которые могут случиться с компьютером.

**Определение названий устройств**

Иногда сложно определить название (имя) устройства. В прошлом это было проще. Устройство всегда находилось в одном месте и никогда не менялось. Unix-подобные системы именно так и действовали. Во времена, когда разрабатывалась система Unix, для «смены дискового устройства» требовалось использовать подъемник, чтобы извлечь из комнаты с ЭВМ устройство размером со стиральную машину. В последние годы типовая аппаратная конфигурация настольного компьютера стала намного динамичнее, и система Linux вынуждена быть более гибкой, чем ее предшественницы.

В примерах, приведенных выше, мы использовали способность современной системы Linux «как по волшебству» монтировать устройства, чтобы узнавать их названия постфактум. Но как быть тем, кто управляет сервером или каким-то другим окружением, где автоматическое монтирование не поддерживается? Как в этом случае определить название устройства?

Сначала давайте посмотрим, как система выбирает названия для устройств. Если вывести содержимое каталога */dev* (где живут все устройства), можно увидеть значительное число устройств:

[me@linuxbox ~]$ **ls /dev**

Содержимое списка показывает некоторые шаблоны в именовании устройств, неполный список которых приводится в табл. 15.2.

**Таблица 15.2. Названия устройств хранения данных в Linux**

|  |  |
| --- | --- |
| **Шаблон** | **Устройство** |
| */dev/fd\** | Приводы гибких дисков. |
| */dev/hd\** | Диски IDE (PATA) в старых системах. Обычно материнские платы содержат два разъема, или *канала* IDE, к каждому из которых можно подключить шлейф, рассчитанный на два устройства. Первое устройство, подключенное к такому шлейфу, называется *ведущим устройством* (master device), а второе — *подчиненным устройством* (slave device). Имена устройств упорядочены так, что ведущее устройство, подключенное к первому каналу, получает имя */dev/hda*, а ведомое устройство, подключенное к первому каналу, получает имя */dev/hdb*; ведущее устройство, подключенное ко второму каналу, получает имя */dev/hdc*, и т.д. Цифра в конце определяет номер раздела на устройстве. Например, имя */dev/hda1* соответствует первому разделу на первом жестком диске в системе, тогда как имя */dev/hda* соответствует всему устройству в целом |
| */dev/lp\** | Принтеры |
| */dev/sd\** | Диски SCSI. В последних версиях системы Linux ядро интерпретирует все дисковые устройства (включая жесткие диски PATA/SATA, флеш-диски и съемные накопители USB, такие как портативные музыкальные плееры и цифровые камеры) как диски SCSI. В остальном система именования напоминает прежнюю систему */dev/hd\**, описанную выше |
| */dev/sr\** | Приводы оптических дисков (приводы CD/DVD, как пишущие, так и нет) |

Кроме того, во многих системах можно увидеть такие символические ссылки, как */dev/cdrom*, */dev/dvd* и */dev/floppy*, которые ссылаются на фактические файлы устройств и предусмотрены для удобства.

Если вам доведется работать в системе, которая не монтирует автоматически съемные носители, вы можете использовать следующий прием для определения названий таких устройств после их подключения. Сначала запустите мониторинг содержимого файла */var/log/messages* в режиме реального времени (для этого могут потребоваться права суперпользователя):

[me@linuxbox ~]$ **sudo tail -f /var/log/messages**

Эта команда выведет несколько последних строк из файла и приостановится. Далее подключите извлекаемое устройство. В этом примере мы использовали 16-мегабайтный флеш-диск. Практически сразу же ядро обнаружит новое устройство и проверит его:

Jul 23 10:07:53 linuxbox kernel: usb 3-2: new full speed USB device using uhci\_hcd and address 2

Jul 23 10:07:53 linuxbox kernel: usb 3-2: configuration #1 chosen from 1 choice

Jul 23 10:07:53 linuxbox kernel: scsi3 : SCSI emulation for USB Mass Storage devices

Jul 23 10:07:58 linuxbox kernel: scsi scan: INQUIRY result too short (5), using 36

Jul 23 10:07:58 linuxbox kernel: scsi 3:0:0:0: Direct-Access Easy Disk 1.00 PQ: 0 ANSI: 2

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] 31263 512-byte hardware sectors (16 MB)

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] Write Protect is off

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] Assuming drive cache: write through

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] 31263 512-byte hardware sectors (16 MB)

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] Write Protect is off

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] Assuming drive cache: write through

Jul 23 10:07:59 linuxbox kernel: sdb: sdb1

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] Attached SCSI removable disk

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: Attached scsi generic sg3 type 0

Когда вывод опять приостановится, нажмите **CTRL+C**, чтобы вернуться в приглашение командной строки. Наибольший интерес для нас представляют строки с упоминанием имени устройства [sdb], соответствующего нашим ожиданиям в отношении названия устройства диска SCSI. В этом отношении следующие две строки являются для нас особенно показательными:

Jul 23 10:07:59 linuxbox kernel: sdb: sdb1

Jul 23 10:07:59 linuxbox kernel: sd 3:0:0:0: [sdb] Attached SCSI removable disk

Они сообщают, что имя */dev/sdb* соответствует всему устройству, а имя */dev/sdb1* — первому разделу на этом устройстве. Как видите, работая с Linux, иногда приходится проводить массу интересных детективных расследований!

**ПРИМЕЧАНИЕ**

Прием с использованием команды tail -f /var/log/messages представляет собой отличный способ наблюдения за происходящим в системе в режиме реального времени.

Зная имя устройства, можно смонтировать флеш-диск:

[me@linuxbox ~]$ **sudo mkdir /mnt/flash**

[me@linuxbox ~]$ **sudo mount /dev/sdb1 /mnt/flash**

[me@linuxbox ~]$ **df**

Filesystem           1K-blocks      Used Available Use% Mounted on

/dev/sda2             15115452   5186944   9775164  35% /

/dev/sda5             59631908  31777376  24776480  57% /home

/dev/sda1               147764     17277    122858  13% /boot

tmpfs                   776808         0    776808   0% /dev/shm

/dev/sdb1                15560         0     15560   0% /mnt/flash

Имя устройства сохраняется неизменным, пока оно остается физически подключенным к компьютеру и до перезагрузки компьютера.

**Создание новых файловых систем**

Представьте, что вам нужно отформатировать флеш-диск и вместо файловой системы FAT32 создать на нем файловую систему, родную для Linux. Для этого следует выполнить две операции: сначала (при необходимости) создать новое распределение разделов, если имеющееся вас не устраивает, а затем создать новую, пустую файловую систему.

**ВНИМАНИЕ**

Следующее упражнение производит форматирование флеш-диска. Используйте диск, не содержащий ничего, что вам было бы нужно, потому что вся информация на диске будет стерта! И снова: **убедитесь, что используете имя устройства, верное для вашей системы, а не то, которое показано в примере. Игнорирование этого предупреждения может привести к форматированию (то есть к стиранию) другого диска!**

**Управление разделами с помощью fdisk**

Программа fdisk позволяет напрямую выполнять низкоуровневые операции с дисковыми устройствами (такими, как жесткие диски и флеш-диски). С помощью этого инструмента можно изменять, удалять и создавать разделы на устройстве. Чтобы приступить к работе с флеш-диском, его нужно сначала размонтировать (если прежде он был смонтирован) и затем запустить программу fdisk, как показано ниже:

[me@linuxbox ~]$ **sudo umount /dev/sdb1**

[me@linuxbox ~]$ **sudo fdisk /dev/sdb**

Обратите внимание, что здесь нужно указать имя, соответствующее устройству, целиком, без номера раздела. После запуска программы вы увидите следующее приглашение:

Команда (m для справки)::

)

После ввода команды m на экране появится меню программы:

Справка:

  DOS (MBR)

   a   изменить флаг загрузочного раздела

   b   изменить вложенную BSD-метку диска

   c   переключить флаг совместимости с DOS

  Общие

   d   удалить раздел

   F   список свободного нераспределенного пространства

   l   список известных типов разделов

   n   добавить новый раздел

   p   показать таблицу разделов

   t   изменить тип раздела

   v   проверить таблицу разделов

   i   печать информации о разделах

  Разное

   m   показать это меню

   u   изменить единицы отображения/ввода

   x   дополнительные функции (только для экспертов)

  Script

   I   загрузка разметки диска из файла сценария sfdisk

   O   сохранение разметки диска в файле сценария sfdisk

  Сохранить и выйти

   w   сохранить таблицу на диск и выйти

   q   выйти без сохранения изменений

  Создать новую метку

   g   создать новую пустую таблицу разделов GPT

   G   создать новую пустую таблицу разделов SGI (IRIX)

   o   создать новую пустую таблицу разделов DOS

   s   создать новую пустую таблицу разделов Sun

Первое, что следует сделать, — исследовать список имеющихся разделов. Для этого введите команду p, она выведет таблицу разделов на устройстве:

Command (m for help): **p**

Disk /dev/sdb: 16 MB, 16006656 bytes

1 heads, 31 sectors/track, 1008 cylinders

Units = cylinders of 31 \* 512 = 15872 bytes

  Device Boot       Start         End      Blocks   Id  System

/dev/sdb1               2        1008       15608+   b  W95 FAT32

Обратите внимание, что устройство имеет объем 16 Мбайт и единственный раздел (1), занимающий 1006 цилиндров из 1008 доступных на устройстве. Раздел идентифицирован как раздел Windows 95 FAT32. Некоторые программы используют этот идентификатор, ограничивая виды операций с диском, но чаще изменение идентификатора не влечет серьезных последствий. Однако ради демонстрации мы изменим его, чтобы показать, что это раздел Linux. Для этого нужно сначала узнать, какой идентификатор обозначает разделы Linux. В листинге, приведенном выше, мы видели, что существующий раздел имеет идентификатор (столбец Id) b. Чтобы увидеть список известных типов разделов, вернитесь к меню программы и обратите внимание на пункт:

l список известных типов разделов

Если ввести команду l, появится длинный список допустимых типов разделов. Среди них можно увидеть идентификатор b типа существующего раздела и идентификатор 83 для Linux. Вернемся обратно к меню программы, где можно увидеть команду изменения идентификатора раздела:

t изменить тип раздела

Введите t и затем новый идентификатор:

Command (m for help): **t**

Selected partition 1

Hex code (type L to list codes): **83**

Changed system type of partition 1 to 83 (Linux)

Это все изменения, которые нам нужно было сделать. До этого момента никаких изменений на самом устройстве не было произведено (все изменения пока просто зафиксированы в памяти программы, а не на физическом устройстве), поэтому теперь запишем измененную таблицу разделов на устройство и выйдем.

Для этого введите команду w:

Command (m for help): **w**

The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: If you have created or modified any DOS 6.x

partitions, please see the fdisk manual page for additional

information.

Syncing disks.

[me@linuxbox ~]$

Если бы мы решили оставить устройство в неизменном состоянии, то могли бы ввести команду q и покинуть программу без записи изменений на устройство. Преду­преждающее сообщение, выглядящее зловещим, можно просто игнорировать[7](http://10.207.106.8/os/pr/978549602303-20.xhtml#footnote-343439-7).

**Создание новой файловой системы с помощью mkfs**

Завершив редактирование разделов (довольно простое, хотя так бывает не всегда), мы создадим на флеш-диске новую файловую систему. Для этого воспользуемся программой mkfs (сокращенно от *make filesystem* — создать файловую систему), способной создавать разные файловые системы. Чтобы создать на устройстве файловую систему ext3, следует передать команде параметр -t и с типом файловой системы ext3, затем указать имя устройства и раздел, который требуется отформатировать:

[me@linuxbox ~]$ **sudo mkfs -t ext3 /dev/sdb1**

mke2fs 1.40.2 (12-Jul-2012)

Filesystem label=

OS type: Linux

Block size=1024 (log=0)

Fragment size=1024 (log=0)

3904 inodes, 15608 blocks

780 blocks (5.00%) reserved for the super user

First data block=1

Maximum filesystem blocks=15990784

2 block groups

8192 blocks per group, 8192 fragments per group

1952 inodes per group

Superblock backups stored on blocks:

        8193

Writing inode tables: done

Creating journal (1024 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.

[me@linuxbox ~]$

Когда выбирается тип файловой системы ext3, программа выводит массу информации. Чтобы восстановить на устройстве оригинальную файловую систему FAT32, следует указать тип файловой системы vfat:

[me@linuxbox ~]$ **sudo mkfs -t vfat /dev/sdb1**

Эту процедуру с редактированием разделов и форматированием можно повторять с любыми дополнительными устройствами хранения, подключаемыми к системе. Хотя в данном примере мы работали с маленьким флеш-диском, ту же процедуру можно применить и к внутренним жестким дискам, и к другим извлекаемым устройствам хранения, таким как жесткие USB-диски.

**Проверка и восстановление файловой системы**

Знакомясь с файлом */etc/fstab*, мы видели некие странные цифры в конце каждой строки. Каждый раз, когда система загружается, она проверяет целостность файловых систем перед их монтированием. Эту проверку выполняет программа fsck (сокращенно от *filesystem check* — проверка файловой системы). Последнее число в каждой записи в файле *fstab* определяет порядок проверки файловых систем. В примере, приведенном выше, видно, что корневая файловая система проверяется первой, вслед за ней проверяются файловые системы *home* и *boot*. Устройства с нулем в последнем поле не проверяются стандартными механизмами.

Программа fsck может не только проверить целостность, но и восстановить поврежденные файловые системы с той или иной степенью успеха в зависимости от масштаба повреждений. В Unix-подобных системах восстановленные фрагменты файлов помещаются в каталог *lost+found*, находящийся в корне каждой файловой системы.

Проверить наш флеш-диск (который предварительно необходимо размонтировать) можно с помощью следующей команды:

[me@linuxbox ~]$ **sudo fsck /dev/sdb1**

fsck 1.40.8 (13-Mar-2012)

e2fsck 1.40.8 (13-Mar-2012)

/dev/sdb1: clean, 11/3904 files, 1661/15608 blocks

По моему опыту, файловые системы повреждаются крайне редко, если нет никаких проблем с аппаратной частью, таких как выход из строя привода диска. В большинстве файловых систем обнаруженные на этапе загрузки повреждения вызывают остановку системы с выводом предложения запустить fsck перед продолжением.

**что такое fsck?**

В культуре Unix слово «fsck» часто используется взамен распространенного ругательства, в котором три буквы совпадают с буквами в имени команды. Это показательно — вы почти наверняка будете произносить упомянутое слово, оказавшись в ситуации, вынуждающей запустить fsck.

**Форматирование гибких дисков**

Те из вас, кто пользуется компьютерами, настолько старыми, что они оборудованы приводами гибких дисков, также смогут управлять этими устройствами. Подготовка чистого гибкого диска выполняется в два этапа. Сначала нужно выполнить низкоуровневое форматирование диска, а затем создать файловую систему. Для форматирования в данном случае используется программа dformat, которой передается имя устройства привода гибких дисков (обычно */dev/fd0*):

[me@linuxbox ~]$ **sudo fdformat /dev/fd0**

Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.

Formatting ... done

Verifying ... done

Затем следует создать файловую систему FAT с помощью mkfs:

[me@linuxbox ~]$ **sudo mkfs -t msdos /dev/fd0**

Обратите внимание, что здесь использован тип файловой системы msdos, чтобы создать старую (и меньшую по размерам) таблицу размещения файлов. После подготовки диска он монтируется как любые другие устройства.

**Непосредственное перемещение данных между устройствами**

Обычно на компьютерах мы работаем с данными, организованными в файлы, однако точно так же можно работать с данными в «низкоуровневой» форме. Если взглянуть на содержимое диска, можно увидеть, что оно состоит из множества «блоков» данных, которые операционная система интерпретирует как файлы и каталоги. Если бы мы умели интерпретировать диски как простые коллекции блоков данных, мы смогли бы выполнять множество полезных задач, таких как клонирование дисков.

Эту задачу решает программа dd. Она копирует блоки данных из одного места в другое. По историческим причинам команда имеет уникальный синтаксис:

dd if=*входной\_файл* of=*выходной\_файл* [bs=*размер\_блока* [count=*число\_блоков*]]

Представьте, что у вас есть два флеш-диска USB одинакового размера и вам нужно создать точную копию первого диска на втором. Допустим, что после подключения к компьютеру им назначаются имена устройств */dev/sdb* и */dev/sdc* соответственно. В этом случае скопировать содержимое первого диска на второй можно следующей командой:

**dd if=/dev/sdb of=/dev/sdc**

Как вариант, если к компьютеру подключено только первое устройство, можно скопировать его содержимое в обычный файл, который впоследствии использовать для восстановления или копирования:

**dd if=/dev/sdb of=flash\_drive.img**

**ВНИМАНИЕ**

dd — очень мощная команда. Ее название происходит от *data definition* (определение данных), но иногда его расшифровывают как *destroy disk* (уничтожить диск), потому что пользователи часто допускают ошибки в параметрах if и of. Всегда дважды проверяйте их, прежде чем нажать **ENTER**!

**Создание образа компакт-диска**

Запись на компакт-диски (CD-R или CD-RW) выполняется в два этапа: сначала нужно создать файл образа ISO, являющийся точным образом файловой системы компакт-диска, а затем записать файл образа на носитель (то есть на сам компакт-диск).

**Создание образа-копии компакт-диска**

Чтобы создать ISO-образ имеющегося компакт-диска, необходимо с помощью dd прочитать все блоки с данными с этого компакт-диска и скопировать их в локальный файл. Например, допустим, что у нас есть компакт-диск с дистрибутивом Ubuntu, и мы хотим создать файл ISO-образа, который потом можно будет использовать для создания нескольких копий. Вставив компакт-диск в привод CD-ROM и определив имя устройства (пусть это будет */dev/cdrom*), мы сможем создать файл ISO-образа следующим способом:

**dd if=/dev/cdrom of=ubuntu.iso**

Этот прием также применим к дискам DVD с данными, но он не будет работать с аудиодисками, так как для хранения данных на них файловая система не используется. Если вы хотите скопировать аудиодиск, обратитесь к команде cdrdao.

**что в имени твоем…**

В руководствах по созданию и записи оптических дисков, таких как CDROM и DVD, которых в избытке на просторах Интернета, часто можно встретить упоминание двух программ, mkisofs и cdrecord. Эти программы были некогда частью популярного пакета cdrtools, созданного Йоргом Шиллингом (Jörg Schilling). Летом 2006-го мистер Шиллинг изменил лицензию в части, касающейся пакета cdrtools, из-за чего она, по мнению многих в сообществе пользователей Linux, стала несовместимой с GNU GPL. Как результат, на основе cdrtools был создан альтернативный проект, включающий программы wodim и genisoimage взамен cdrecord и mkisofs соответственно.

**Создание образа из коллекции файлов**

Создать файл ISO-образа, включающий содержимое некоего каталога, можно с помощью программы enisoimage. Для этого сначала создадим каталог со всеми необходимыми файлами для включения в образ и затем командой genisoimage создадим файл образа. Например, если предположить, что вы создали каталог *~/cd-rom-files* и наполнили его файлами для записи на компакт-диск, следующая команда создаст файл образа с именем *cd-rom.iso*:

**genisoimage -o cd-rom.iso -R -J ~/cd-rom-files**

Параметр -R требует добавить метаданные *расширений Rock Ridge*, позволяющих использовать длинные имена файлов и права доступа к файлам в стиле POSIX. Аналогично, параметр -J включает *расширения Joliet*, разрешающие использовать длинные имена файлов в Windows.

**Запись образа компакт-диска**

После подготовки файла образа его можно записать на оптический носитель. Большинство команд, обсуждаемых ниже, применимы и для записи на носители CD-ROM и DVD.

**Непосредственное монтирование файла ISO-образа**

Существует один трюк, позволяющий монтировать ISO-образы, хранящиеся на жестком диске, и работать с ними, как если бы это были оптические носители. Параметр -o loop, добавленный в команду mount (вместе с обязательным параметром -t iso9660, определяющим тип файловой системы), позволяет смонтировать файл образа в дерево файловой системы, как если бы это было обычное устройство:

**mkdir /mnt/iso\_image**

**mount -t iso9660 -o loop image.iso /mnt/iso\_image**

В примере, приведенном выше, мы создали точку монтирования с именем */mnt/iso\_image* и затем смонтировали в нее файл образа *image.iso*. После монтирования образа с ним можно работать как с настоящим диском CD-ROM или DVD. Не забудьте размонтировать образ, когда он станет не нужен.

**Очистка перезаписываемых компакт-дисков**

Перезаписываемые компакт-диски CD-RW нужно стирать, или очищать, перед повторным использованием. Для этого воспользуемся командой wodim, указав ей имя устройства пишущего привода компакт-дисков и тип очистки. Программа wodim предлагает несколько типов очистки. Для минимальной (и самой быстрой) очистки следует указать тип fast:

**wodim dev=/dev/cdrw blank=fast**

**Запись образа**

Записать образ можно с помощью все той же программы wodim, указав ей имя устройства пишущего привода компакт-дисков и имя файла образа:

**wodim dev=/dev/cdrw image.iso**

Помимо имени устройства и файла образа программа wodim поддерживает массу дополнительных параметров. Чаще других используются параметры -v (обеспечивает вывод подробной информации в ходе записи) и -dao (выполняет запись на диск в режиме *disc-at-once* — диск целиком). Режим «диск целиком» следует использовать, если вы собираетесь воспроизводить диски в коммерческих целях. По умолчанию wodim использует режим *track-at-once* (по одной дорожке), который хорошо подходит для записи музыкальных треков.

**Дополнительные сведения**

Часто бывает полезно проверить целостность ISO-образа, загруженного из Интернета. В большинстве случаев распространители ISO-образов сопровождают их *файлами с контрольными суммами*. Контрольная сумма — это результат экзотических математических вычислений в виде числа, представляющего содержимое целевого файла. Если содержимое файла образа изменится хотя бы в одном бите, его контрольная сумма будет отличаться от указанной распространителем. Для вычисления контрольной суммы чаще всего используется программа md5sum, возвращающая уникальное шестнадцатеричное число:

**md5sum image.iso**

34e354760f9bb7fbf85c96f6a3f94ece   image.iso

Загрузив образ, запустите md5sum для него и сравните результат работы md5sum со значением, указанным распространителем.

Помимо проверки целостности загруженного файла, программу md5sum можно использовать для проверки вновь записанного оптического носителя. Для этого сначала вычислите контрольную сумму для файла образа, а затем — для носителя. Вся хитрость проверки носителя заключается в том, чтобы ограничить вычисления частью оптического носителя, содержащей образ. Для этого определите число 2048-байтных блоков в образе (запись на оптические носители всегда выполняется блоками по 2048 байт) и прочитайте с носителя ровно столько блоков. Для некоторых типов носителей это не обязательно. Например, компакт-диск, записанный в режиме disc-at-once, можно проверить так:

**md5sum /dev/cdrom**

34e354760f9bb7fbf85c96f6a3f94ece /dev/cdrom

Многие типы носителей, такие как DVD, требуют точного вычисления числа блоков. Следующий пример демонстрирует проверку целостности файла образа *dvd-image.iso* и диска в устройстве */dev/dvd* привода DVD. Вам понятно, как работает эта команда?

**md5sum dvd-image.iso; dd if=/dev/dvd bs=2048 count=$(( $(stat -c "%s" dvd-image.iso) / 2048 )) | md5sum**

[7](http://10.207.106.8/os/pr/978549602303-20.xhtml#footnote-343439-7-backlink) Оно гласит: «ВНИМАНИЕ: если вы создали или изменили разделы DOS 6.x, прочитайте дополнительную информацию на странице справочного руководства для команды fdisk». — *Примеч. пер.*