LINUX И NTFS

«...ntfs not supported by kernel»
Сообщение системы об ошибках.

СЕРГЕЙ ЯРЕМЧУК

До недавнего времени острой необходимости в доступе к разделам с файловой системой NTFS, в общем-то, и не было. Ругать разработчиков ядра Linux не за что, необходимые работы ведутся уже давно и отнюдь не безуспешно. Уже в 1995 году для ядер серии 2.0 был доступен патч для работы с этой файловой системой, а с версии 2.2 (если быть точнее 2.1.74) подержка NTFS в ядро была включена стандартно. Но все равно разработки велись несколько вяло. Причин тому несколько. Семейство пользовательских операционных систем Windows до Me включительно поддерживают исключительно FAT, а NTFS использовалась на ОС корпоративно-серверного уровня, где вряд ли кто-то додумается поставить одновременно две операционные системы на одном компьютере. Появление Windows 2000 практически ничего не изменило, на домашних компьютерах ее устанавливали довольно неохотно, и если посмотреть на форумах тех времен, то можно отметить относительное спокойствие. А вот Windows XP смог привлечь пользователя, этот момент и стоит считать началом действительного интереса разработчиков к проблеме. Многие (да почти все) производители включили поддержку NTFS в ядрах своих дистрибутивов. Проблем с доступом к разделам NTFS не обнаружат пользователи Mandrake, SUSE, ALTLinux, ASPLinux, Slackware, Debian и прочих популярных дистрибутивов. Только компания RedHat, очевидно, руководствуясь лицензионной чистотой своего дистрибутива, не включила поддержку данной ФС. Поэтому пользователи RedHat и Fedora увидят при попытке монтирования раздела сообщение, вынесенное в эпиграф.

На данный момент имеются два свободных проекта, по-своему решающие вопрос работы с разделами NTFS. Первый, проект Linux-NTFS, предлагает традиционный подход, т.е. написание драйвера, который позволит нормально работать с этой файловой системой. Второй, проект Captive NTFS, пробует решить все эмуляцией системы NT.

Проект Linux-NTFS

Сейчас для GNU/Linux фактически существуют два драйвера NTFS. Первый используется в ядрах серии 2.4.x и в 2.5.0-2.5.10, этот драйвер имеет ограниченные возможности по записи в раздел NTFS, и поэтому такой режим помечен как опасный, т.е. может привести к потере данных. Этот драйвер долго практически не развивался. Вторая версия драйвера была фактически переписана заново одним из разработчиков Антоном Алтапармаковым (Anton Altaparmakov), причем записи в раздел NTFS уделялось повышенное внимание, драйвер стал более легким, простым и быстрым, поддерживаются NTFS версий 1.2, 3.0 и 3.1, Unicod, сжатые файлы. Хотя в настоящее время имеются еще проблемы, например, драйвер не понимает зашифрованные файлы, квоты, игнорирует информацию безопасности и ограничен по записи. Проблемы эти, в общем-то, существуют не по вине разработчиков. Все дело в том, что Microsoft не выпустило никакой документации по внутреннему строению NTFS, и драйвера разрабатываются фактически с пустого листа, методом научного тыка и плясания с бубном вокруг компьютера. Строение же NTFS довольно сложное и внутри напоминает базу данных, когда изменение в одном месте требует замен и во многих других местах файловой системы, а иначе она попросту будет разрушена. Поэтому новый драйвер пока может только перезаписывать существующие файлы, но не может изменять их длину, добавлять новые и удалять старые файлы, не доступны операции с каталогами.

Теперь немного практики. В ядрах, начиная с 2.5.11 и, конечно же, в серии 2.6, используется новый драйвер, поэтому если у вас с поддержкой не сложилось, то просто перекомпилируйте ядро, включив нужные пункты. Так как в народе еще популярны ядра серии 2.4, то для их владельцев доступен патч.

Текущую версию драйвера узнать очень просто:

```
# dmesg | grep -i ntfs
NTFS driver v1.1.22 [Flags: R/O MODULE]
```

Или такой вариант:

```
# grep -i ntfs /var/log/messages
Jun  3 12:01:29 grinder kernel: NTFS driver v1.1.22 [Flags: R/O MODULE]
```

Как видите, ядро Slaksware 9.1, которое использовалось во время написания статьи, поддерживает первую версию драйверов, но это довольно легко исправить. Если вывод этих команд ничего не дал, то дополнительно проверить поддерживаемые ядром файловые системы можно, введя:

```
# cat /proc/filesystems
```

За драйверами, патчами, утилитами для работы с NTFS и документацией идем на сайт http://linux-ntfs.sourceforge.net.

Владельцам RedHat и Fedora можно идти сразу на страницу http://linux-ntfs.sourceforge.net/rpm/index.html, где доступны прекомпилированные rpm-пакеты с необходимыми модулями. Выбираем нужный и устанавливаем.

```
#rpm -ihv --noscripts kernel-ntfs-2.6.5-1.358.i586.rpm
#/sbin/depmod -a
```

Например, мы хотим пересобрать ядро 2.4.25 с новым драйвером.

Скачиваем по ссылке патч к используемому ядру, в моем случае это linux-2.4.25-ntfs-2.1.6a.patch.bz2, и распаковываем его.

```
# bunzip2 linux-2.4.25-ntfs-2.1.6a.patch.bz2
```

Теперь распаковываем ядро, взятое с www.kernel.org, можно использовать и имеющееся в вашем дистрибутиве, только тогда возьмите и патч с соответствующим номером.

```
#cd /usr/src
```

Накладываем патч:

```
#cd linux
#patch -p1 < ../linux-2.4.25-ntfs-2.1.6a.patch
```

В последнем случае не должно быть сообщений об ошибках, иначе проверьте, правильно ли заданы пути. И теперь приступаем к конфигурации ядра. Набираем make nenuconfig, в «File systems» необходимо включить пункты драйвера NTFS (рис. 1). Сохраняем конфигурацию и выходим. После чего идет стандартная компиляция ядра и настройка загрузчика.

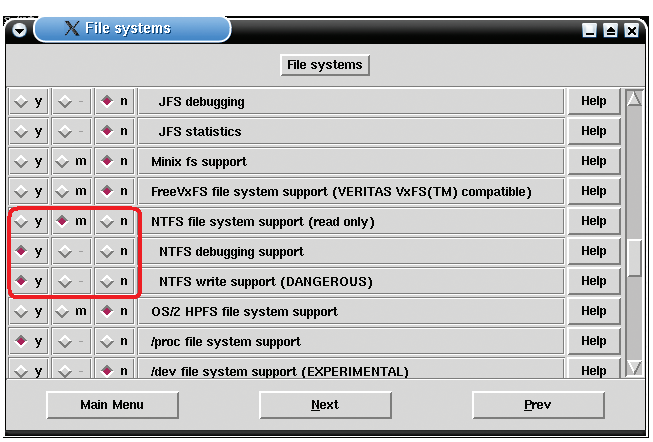

Рисунок 1

Выполнив все, перезагружаемся с новым ядром и проверяем:

```
#grep -i ntfs /var/log/messages
Jun  3 21:09:08 grinder kernel: NTFS driver 2.1.6a [Flags: R/W DEBUG MODULE].
Jun  3 21:09:08 grinder kernel: NTFS volume version 1.2.
```

Как видите, версия драйвера обновилась и соответствует установленной; пробуем примонтировать раздел, для того чтобы узнать, какой именно, вводим:

```
#   fdisk -l  |  grep -i ntfs
/dev/hda7        3522      3649    1028128+  7  HPFS/NTFS
```

Монтируем раздел.

```
# mount /dev/hda7 /mnt/temp/ -t ntfs
```

Обратите внимание, что ключ -t, указывающий на файловую систему, в данном случае обязательный, иначе система не сможет сама определить ее и выдаст примерно такое сообщение:

```
# mount /dev/hda7 /mnt/temp/
mount: wrong fs type, bad option, bad superblock on /dev/hda9,
       or too many mounted file systems
```

Для того чтобы просмотреть опции по умолчанию, с которыми монтируется раздел, вводим:

```
# cat /proc/mounts | grep -i ntfs
/dev/hda7 /mnt/temp ntfs rw,uid=0,gid=0,fmask=0177,dmask=077,
nls=koi8-r,errors=continue,mft_zone_multiplier=1 0 0
```

Как видите, раздел смонтирован в режиме «чтение-запись», это можно изменить на «только чтение», явно задав в командной строке опцию -r (или -o ro). Владельцем является root (uid=0, gid=o), опция errors указывает, как будет себя вести система при возникновении ошибок, поддерживается два варианта: continue (продолжает работу) и recover (пытается восстановить), в настоящее время поддерживается только замена boot-сектора резервным. Опции fmask и dmask задают параметры доступа к файлам и каталогам соответственно, возможно использование общей опции umask, задающей доступ к файлам и каталогам одновременно. NTFS хранит имена в Unicode, но драйвер переводит их в ASCII; чтобы указать на используемый язык, применяются две конструкции: -o iocharset= или в новом варианте -o nls=. Для отображения русских имен используются koi8-r, возможно задание utf8 (если ядро не поддерживает Unicod, то дополнительно используйте utf8=true). В моем случае nls был выбран автоматически, потому что при конфигурировании ядра эта кодировка была прописана по умолчанию (смотрите в File Systems – Native Language Support – Default NLS Option). Параметр mft_zone_multiplier указывает на размер зарезервированной в master file table части, которая содержит информацию о файле. Примечательно, что маленькие файлы полностью помещаются в MFT, что позволяет быстро его находить и избежать потерь дискового пространства присущих FAT (в FAT файл размером 1 байт займет на диске минимум 4 Кб). Первоначально задается при форматировании, но в процессе эксплуатации может изменяться на лету. Цифра 1 является значением по умолчанию и соответствует 12.5% зарезервированного объема, 2 – 25%, 3 – 37.5% и 4 – 50.0%. Полная опция монтирования может быть указана в таком виде:

```
#mount /dev/hda7 /mnt/temp/ -t ntfs -r -o nls=koi8-r -o ↵
   uid=500,gid=winuser,umask=0222
```

часть после второй -o, вообще говоря, не нужна и дана для примера.

После этого с чтением данных проблем быть не должно, все имена будут отображаться нормально.

Дополнительно проект Linux-NTFS предоставляет и ряд утилит для работы с NTFS из-под Linux, библиотеку libntfs, обеспечивающую доступ к функциям NTFS программ, в том числе и других разработчиков, а также libntfs-gnomevfs – модуль для Gnome VFS (virtual filesystem), обеспечивающий универсальный доступ ко всем файловым системам. Большая часть утилит ориентирована скорее на разработчиков, но годом раньше вообще ничего подобного не было, прогресс налицо. Все они доступны в пакете ntfsprogs, который

распространяется как в исходных кодах, так и имеются прекомпилированные пакеты. Установка из исходников проблем не вызывает, все те же стандартные ./configure; make; make install. В результате в системе появится еще 10 утилит:

- ntfsfix – для установки измененных драйвером разделов NTFS, нечто вроде scandisk, который должен использоваться после каждой записи (особенно со старым драйвером) во избежание возможной потери данных, для того чтобы привести файловую систему в непротиворечивое состояние.
- mkntfs – для создания NTFS 1.2 (поддерживается всеми Windows NT/2000/XP).
- ntfscat – является аналогом стандартной UNIX-утилиты cat, предназначен для чтения файлов в разделах NTFS.
- ntfsclone – предназначена для клонирования (копирования, сохранения, создания резервного образа и восстановления) раздела с файловой системой NTFS. Работает на уровне секторов диска и сохраняет только используемые данные, неиспользуемые заполняются нулями, что позволяет эффективно сжимать полученные образы. Полезна для создания точных копий раздела и восстановления системы. Примеры:
  - Создание копии раздела:

```
#ntfsclone --output system.img /dev/hda1
```

  - Восстановили раздел:

```
#ntfsclone --overwrite /dev/hda1 system.img
```

  - Так можно заглянуть внутрь созданного образа:

```
#mount -t ntfs -o loop system.img /mnt/ntfsclone
```

- ntfscluster – идентификация файлов в указанном разделе или области NTFS. Работает в трех режимах:
  - info (режим по умолчанию) – покажет общую информацию об области NTFS;
  - sector – покажет список файлов в заданном диапазоне секторов;
  - cluster – то же, что и предыдущий, только выводит список файлов в группе.

```
#./ntfscluster /dev/hda7
```

```
Bytes per sector        : 512
bytes per cluster       : 4096
sectors per cluster     : 8
bytes per volume        : 2212564992
sectors per volume      : 540177
clusters per volume     : 67522
initialized mft records : 56
mft records in use      : 40
mft records percentage  : 71
bytes of free space     : 2207653888
sectors of free space   : 4311824
clusters of free space  : 538978
percentage free space   : 99
bytes of user data      : 151552
sectors of user data    : 296
clusters of user data   : 37
percentage user data    : 0
bytes of metadata       : 4759552
sectors of metadata     : 9296
clusters of metadata    : 1162
percentage metadata     : 0
```

- ntfsinfo – выводит атрибуты по номеру inode или имени файла.
- ntfslabel – выведет или установит метку файловой системы NTFS (метка до 128 Unicode-знаков).

```
# ./ntfslabel -v /dev/hda7
```

```
Using locale 'ru_RU.KOI8R'.
```

- ntfsls – аналог UNIX-утилиты ls (Windows – dir) – выводит список файлов в разделе NTFS (монтировать необязательно).

```
# ./ntfsls -v -d /dev/hda7
```

```
Using locale 'ru_RU.KOI8R'.
RECYCLER
System Volume Information
тест.zip
тест.htm
```

- ntfsresize – а вот это действительно полезная утилита, предназначена для изменения размера файловой системы NTFS без потерь данных. Примечание: утилита не манипулирует размерами разделов, для этого необходимо воспользоваться утилитой fdisk. Как работать с ней, поговорим дальше.
- ntfsundelete – восстановление удаленных файлов на разделе NTFS. Имеет три режима работы:
  - scan (режим по умолчанию) – просмотр файловой системы на предмет наличия удаленных файлов, при нахождении выводит список таких файлов;
  - undelete – пытается, насколько это возможно, восстановить утраченные данные (кроме сжатых и зашифрованных файлов), файл сохраняется в другой раздел;
  - copy – полезен большей частью при отладке, сохраняет данные MFT в файл.

```
#./ntfsundelete /dev/hda7
```

```
Volume is dirty.
Run chkdsk and try again, or use the –force option.
```

Сообщение «Volume is dirty» может возникнуть после записи в раздел, изменения размера раздела и других возможных операций, связанных с изменением данных, после каждой такой операции во избежание несоответствия рекомендуется проверка раздела средствами Windows.

```
#./ntfsundelete /dev/hda7  --force
```

```
Volume is dirty.
Forced to continue.
Inode   Flags  %age  Date         Size  Filename
-----------------------------------------------
16      F..!   0%  1970-01-01       0  <none>
17      FN..   100%  2004-06-02    2318  bootex.log
18      F..!   0%  1970-01-01       0  <none>
19      F..!   0%  1970-01-01       0  <none>
20      F..!   0%  1970-01-01       0  <none>
21      F..!   0%  1970-01-01       0  <none>
22      F..!   0%  1970-01-01       0  <none>
23      F..!   0%  1970-01-01       0  <none>
46      FN..   100%  2004-04-18    6011  home.jpg
47      FN..   100%  2004-04-18   12928  index_r1_c1.jpg
48      FN..   100%  2004-04-18   25171  index_r1_c2.jpg
49      FN..   100%  2004-04-18    2392  index_r2_c1.jpg
50      FN..   100%  2004-04-18   10352  index_r2_c2.jpg
51      FN..   100%  2004-04-18    5878  map.jpg
52      FR..   100%  2004-04-18      53  poloska.gif
53      FN..   100%  2004-04-18   47616  Thumbs.db

Files with potentially recoverable content: 9
```

Пробуем восстановить один из найденных файлов:

```
#./ntfsundelete /dev/hda7 -s -m home.jpg --force
Volume is dirty.
Forced to continue.
Inode   Flags  %age  Date         Size  Filename
----------------------------------------------------
46      FN..   100%  2004-04-18   6011  home.jpg

Files with potentially recoverable content: 1
```

В настоящее время начата разработка еще нескольких утилит, некоторые из них находятся в стадии альфы. Это ntfswipe – позволит зачистить нулями свободные части диска, ntfsdefrag – дефрагментатор файлов, каталогов и MFT, для проверки диска будет использоваться ntfsck, nttools позволит просмотреть/создать/изменить/копировать/найти требуемые значения (эквивалентны командам ntfscp, ntfsgrep, ntfstouch, ntfsrm, ntfsrmdir, ntfsmkdir). И пока в планах еще одна утилита ntfsdiskedit, которая позволит работать с дисковыми структурами NTFS.

Начиная с Windows 2000, файловая система получила новое понятие – динамический диск, который пришел на смену стилю разделов, принятых еще в MS-DOS (Basic Disks), и позволил снять все ограничения файловой системы, в том числе и создавать многодисковые тома. Информация хранится в журналируемой базе данных Logical Disk Manager (LDM) в последнем 1 Мб диска. Начиная с версии ядра 2.5.29, в него включаются драйвера для работы с LDM, для версий 2.4.19 и 2.4.20-pre1-ac1 доступен патч. Отдельно в пакете linux-ldm доступны две утилиты для работы с LDM. Первая ldminfo выводит детальную информацию о LDM-базе, ldmutil – предназначена для восстановления, резервирования и изменения LDM-баз. Планируется в ближайшее время начать работу над библиотекой ldmlib, в которую будет вынесена часть кода, и еще над несколькими утилитами, которые позволят работать с отдельными частями раздела LDM и записывать/читать информацию с базы. Как видите, хотя работа над новыми драйверами и утилитами идет полным ходом, но, увы, на данный момент они не могут обеспечить пользователя требуемой (полной) функциональностью, и, как говорится, нормальные герои всегда идут в обход, что и сделали разработчики в проекте, о котором пойдет речь дальше.

Проект Captive NTFS

Тема эмуляции работы операционных систем, и в частности Windows, довольно популярна в среде Linux/UNIX, достаточно вспомнить проекты вроде Wine или Crossover Office, поэтому не удивительно, что нашлись разработчики, попытавшиеся заставить работать Linux с разделами NTFS средствами самой Windows.

Свободный проект, реализующий возможность записи/чтения данных с разделов NTFS путем эмуляции всех необходимых уровней Windows, называется Captive и находится по адресу: http://www.jankratochvil.net/project/captive. Для работы captive потребуется драйвер ntfs.sys и системный модуль ядра NT – ntoskrnl.exe. Если под рукой имеются компьютеры с установленными Windows XP, то их можно взять в C:\WINDOWS\system32\ntoskrnl.exe и C:\WINDOWS\system32\drivers\ntfs.sys), но captive нормально работает не со всеми версиями драйверов Windows (полный список проверенных в работе доступен на сайте), во всяком случае со взятыми с русифицированной версии Windows XP с первым сервис-паком работа у меня не пошла. В этом случае придется тащить файлы с сайта Microsoft, где они находятся по адресу: http://www.microsoft.com/WindowsXP/pro/downloads/servicepacks/sp1/checkedbuild.asp, с которыми captive работает отменно.

Правда, ситуация с их использованием двоякая, с одной стороны, они лежат свободно, с другой – предназначены для пользователей Windows, во всяком случае во всех дистрибутивах, имеющих captive, их приходится добывать самостоятельно. Для работы их следует положить в каталог /var/lib/captive. Так как драйверы Windows требуют особых привилегий вроде прямого доступа к железу, то полную 100% эмуляцию реализовать не получится, т.к. UNIX-системы, естественно, будут защищаться от процессов, которые лезут не в свое дело, поэтому эмулируемая среда отделена от остальной части UNIX и любой код, выполняющийся в этой среде, не должен привести к краху системы. Для защиты от возможного краха используются несколько технологий. Так, для работы потребуется модуль ядра LUFS (Linux Userland File System), который, если будете собирать из исходников, нужно взять с http://lufs.sourceforge.net/lufs. Во время установки создается новый пользователь и группа captive, от имени которых и будут работать процессы, а сам процесс по умолчанию запускается в изолированой CORBA sandbox среде и chroot-окружении. К сожалению, эти ограничения сказались на невозможности работы одновременно сразу с несколькими разделами. Драйверы же на сайте проекта captive доступны как в исходных кодах, так и в прекомпилированном виде со статической линковкой, работают они одинаково, проблема может быть только с модулем ядра lufs.o, который должен быть собран под определенную версию ядра. После установки для первоначального конфигурирования необходимо запустить утилиту captive-install-acquire (рис. 2), которая проверит наличие необходимых библиотек и при необходимости закачает все нужное.

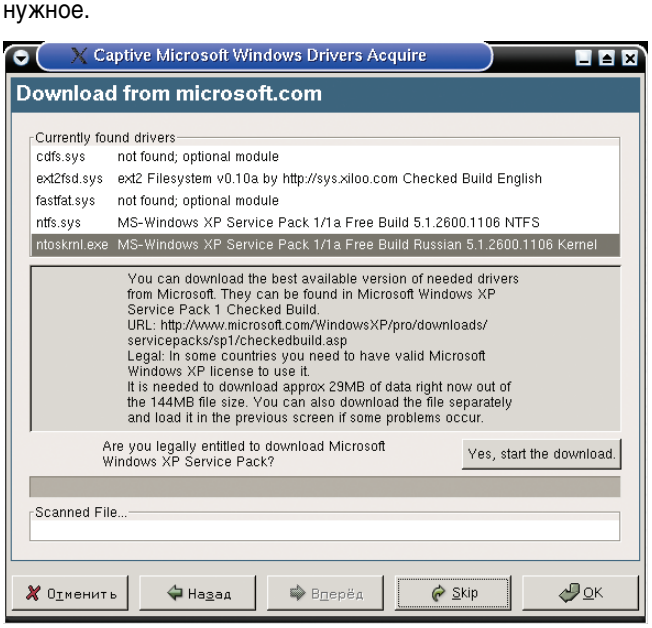

Рисунок 2

Если все есть, то можно монтировать раздел:

```
# mount -t captive-ntfs /dev/hda7 /mnt/utils/
```

В случае ошибок вся информация доступна /var/log/messages.

```
#cat /var/log/messages | grep captive-lufs
```

Если требуется автоматическое монтирование при загрузке системы, используйте скрипт captive-install-fstab с параметром -add, который автоматически добавит в файл /etc/fstab используемый раздел.

Еще пару слов хочу сказать о коммерческом драйвере от Paragon Software Group, обеспечивающем прозрачный доступ к разделам с файловой системой NTFS. Доступен драйвер в двух версиях: персональной и профессиональной. Как и у предыдущих проектов, поддерживаются все версии файловой системы NTFS, сжатые файлы и каталоги, размеры дисков до 127 Гб, в персональной версии работа возможна только в режиме чтения, а в професиональной возможна запись (что, в общем, и не должно вызывать удивления; учитывая опыт работы этой компании на подобном поприще). Демо-версию драйвера, которую производитель разрешает использовать без регистрации в течение 30 дней и поддерживающую только чтение, можно скачать с http://www.ntfs-linux.com.

Для установки потребуются исходники ядра. Установка проста до безобразия: после распаковки архива запускаем скрипт install.sh (возможен запуск в интерактивном режиме ./install.sh –interactive и при помощи –iocharset=koi8-r возможно установить кодировку по умолчанию для раздела NTFS).

После этого, если не выбрано автоматическое монтирование при загрузке, можно примонтировать раздел вручную:

```
#mount -t ufsd /dev/hda7 /mnt/test_ntfs
# mount -t ufsd -o iocharset=koi8-r /dev/hda7 /mnt/test_ntfs
```

Изменение размеров NTFS из-под GNU/Linux

Теперь давайте попробуем разобраться с вопросом, можно ли изменить размер раздела с файловой системой NTFS прямо из-под GNU/Linux, не прибегая к посторонним утилитам вроде Partition Magic. По заверению разработчиков проекта Linux-NTFS, утилита ntfsresize, доступная пользователям с июля 2002, позволит изменить размер раздела, не разрушив при этом данных, причем нормально поддерживаются все версии NTFS и работает нормально со всеми версиями системы (Windows XP/2000/NT4, Windows Server 2003 и даже беты Longhorn). Разработчиками сделано все, чтобы свести риск потери данных к минимуму, для подстраховки проводятся всевозможные проверки, включая проверку на непротиворечивость данных, и, найдя проблемы и при возникновении подозрений, утилита отказывается производить изменение размера. Давайте проверим. Хотя уже появились графические средства, использующие ntfsresize, но для начала разберемся с первоисточником, т.к. в некоторых дистрибутивах вам позволят изменить раздел только с командной строки (например, забравшись во вторую консоль [Ctrl]-[Alt]-[F2]). Для работы ntfsresize драйвер поддержки NTFS в ядре не нужен, утилита обращается напрямую к диску. Также если не требуются все утилиты или хотите использовать ее в спасательной дискете, то можно взять статически слинкованную версию ntfsresize по адресу: http://mlf.linux.rulez.org/mlf/ezaz/ntfsresize-static-1.9.1.tgz.

Теперь посмотрим, что у нас есть:

```
# ./ntfsresize -i /dev/hda7
```

```
ntfsresize v1.9.1
NTFS volume version: 1.2
Cluster size      : 1024 bytes
Current volume size: 1052803584 bytes (1053 MB)
Current device size: 1052803584 bytes (1053 MB)
Checking filesystem consistency ...
100.00 percent completed
Accounting clusters ...
Space in use      : 5 MB (0,4%)
Estimating smallest shrunken size supported ...
File feature         Last used at      By inode
$MFT              :         1 MB            0
You might resize at 4526080 bytes or 5 MB (freeing 1048 MB).
Please make a test run using both the -n and -s options before real resizing!
```

Команда выдала все о разделе NTFS и сообщила, что можем уменьшить раздел вплоть до 5 Мб. Но перед реальным изменением желательно прогнать тест.

```
# ./ntfsresize --no-action --size  500M /dev/hda7
```

Если в результате получим сообщение «The read-only test run ended successfully.», то можно смело приступать к изменению размера. Если же команда выдала «ERROR:», то лучше для начала исправить ошибки, спешка в данном случае ни к чему хорошему не приведет.

```
# ./ntfsresize -s 500M /dev/hda7
```

```
ntfsresize v1.9.1
NTFS volume version: 1.2
Cluster size      : 1024 bytes
Current volume size: 1052803584 bytes (1053 MB)
Current device size: 1052803584 bytes (1053 MB)
New volume size   : 499999232 bytes (500 MB)
Checking filesystem consistency ...
100.00 percent completed
Accounting clusters ...
Space in use      : 5 MB (0,4%)
Needed relocations : 4394 (5 MB)
WARNING: Every sanity check passed and only the DANGEROUS operations left.
Please make sure all your important data had been backed up in case of an
unexpected failure!
Are you sure you want to proceed (y/[n])? Y
```

Если все данные сохранены, то соглашаемся:

```
Are you sure you want to proceed (y/[n])? y
Schedule chkdsk for NTFS consistency check at Windows boot time ...
Resetting $LogFile ... (this might take a while)
Relocating needed data ...
100.00 percent completed
Updating $BadClust file ...
Updating $Bitmap file ...
Updating Boot record ...
Syncing device ...
Successfully resized NTFS on device '/dev/hda9'.
You can go on to shrink the device e.g. with 'fdisk'.
IMPORTANT: When recreating the partition, make sure you
  1)  create it with the same starting disk cylinder
  2)  create it with the same partition type (usually 7, HPFS/NTFS)
  3)  do not make it smaller than the new NTFS filesystem size
  4)  set the bootable flag for the partition if it existed before
Otherwise you may lose your data or can't boot your computer from the disk!
```

Проверить работу можно, введя:

```
#./ntfsresize --info --force /dev/hda7
```

Как видите, утилита свою работу проделала, но изменила только размер самой файловой системы NTFS, размер же дискового раздела остался неизменным (что можно легко проверить, введя: fdisk -l | grep -i ntfs), и далее, следуя инструкции, необходимо проделать еще несколько шагов.

Неплохо бы для начала на всякий случай сохранить MBR.

```
# dd if=/dev/hda of=hda.mbr bs=512 count=1
```

Запускаем fdisk, конечно, было бы нагляднее воспользоваться cfdisk, но если указать ему новое значение раздела в мегабайтах, то он округлит до ближайшего значения, которое, вполне возможно, будет меньше требуемой величины, что приведет к невозможности работать с разделом, fdisk же корректно обрабатывает ситуацию.

```
# fdisk /dev/hda
```

```
Command (m for help): p

Disk /dev/hda: 30.0 GB, 30020272128 bytes
255 heads, 63 sectors/track, 3649 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1   *           1         242     1943833+  83  Linux
....
/dev/hda7            3522        3649     1028128+   7  HPFS/NTFS

Partition table entries are not in disk order
```

Удаляем раздел, на котором размещается NTFS, в моем случае это 7.

```
Command (m for help): d
Partition number (1-7): 7
```

Создаем на его месте раздел с новым размером 500 Мб, при этом начальный цилиндр обязательно должен совпадать, т.е. судя по выводу выше – 3522.

```
Command (m for help): n
First cylinder (3522-3649, default 3522): 3522
Using default value 3522
Last cylinder or +size or +sizeM or +sizeK (3522-3649, default 3649): +500M

Command (m for help): t
Partition number (1-7): 7
```

Устанавливаем тип раздела для NTFS – 7.

```
Hex code (type L to list codes): 7

Changed system type of partition 7 to 7 (HPFS/NTFS)
```

Записываем изменения и выходим.

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

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

Если все выше проделывалось с LiveCD или дискетного дистрибутива, то сразу же можно просмотреть информацию о разделе.

```
# ./ntfsresize -i -f /dev/hda7
```

```
...
Current volume size: 499999232 bytes (500 MB)
Current device size: 509935104 bytes (510 MB)
```

Если же вся работа производилась в работающей с жесткого диска системе, необходимо заново перечитать данные о новом разбиении, для чего придется перезагрузиться.

Но для нормальной работы обязательно нужно проверить файловую систему средствами Windows.

Уже появились и графические утилиты, в том числе и инсталляторы, позволяющие изменить раздел в наглядном и понятном любому пользователю виде, которые используют код ntfsresize. К таким приложениям относятся DiskDrake от MandrakeSoft, который может встретиться и в других производных от Mandrake дистрибутивах, YaST от SUSE доступный и после установки. Некоторые дистрибутивы используют коммерческие утилиты для разбиения разделов ASPLinux – PartitionExpert, Xandros использует PQDisk. Но самой известной, распространенной и, главное, свободной графической утилитой, предназначенной для разбиения диска, является QTParted (http://qtparted.sourceforge.net, рис. 3). Те, кто пользовался PartitionMagic, трудностей с освоением не встретят. Выбираем нужный раздел, далее правый клик – Resize (рис. 4), полозком или цифрами выставляем новый размер и подтверждаем изменения File → Commit.

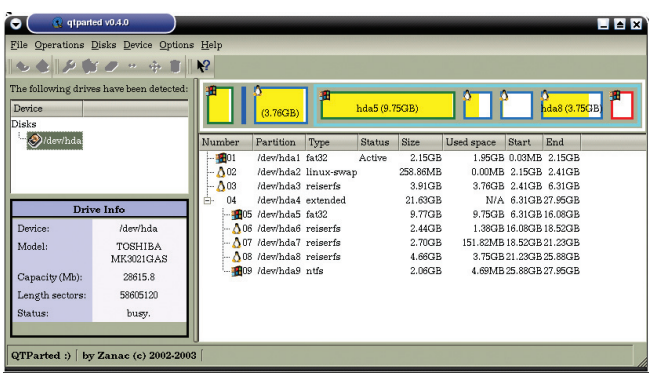

Рисунок 3

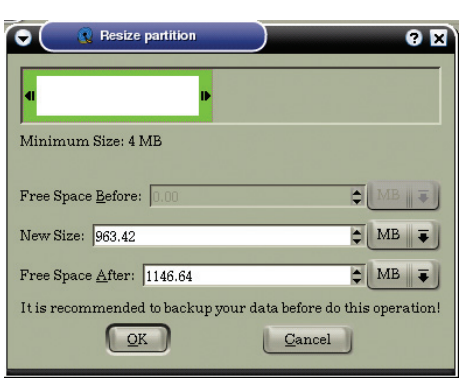

Рисунок 4

К сожалению, можно сделать вывод, что работа GNU/Linux с файловой системой NTFS еще далека от идеала, а учитывая трудности написания вслепую драйвера и проблемы с полной эмуляцией, когда будет окончательное решение, сказать пока трудно. Но как видите, по сравнению с предыдущими годами, сдвиги есть, и работа ведется усиленными темпами. Так, сайт Linux-NTFS уже давно не обновлялся, потому что разработчики сосредоточились на конечном результате. Будем надеяться, что он не заставит себя долго ждать.