Paragon NTFS & HFS for Linux 8.1

User Manual

Table of Contents

1. Introduction

1.1 About this document

Information provided in this manual applies to all products, unless otherwise noted:

Paragon NTFS & HFS for Linux Combo 8.1 Professional

Paragon NTFS & HFS for Linux Combo 8.1 Express

The first and the second products provide support for NTFS or HFS only, respectively, while the third one provides support for both NTFS and HFS in a single kernel module.

1.2 Historical review

Historically, different operating systems supported different file systems. Sharing files among different platforms was not an easy task. For instance, documents that were created in Windows and are stored on NTFS partitions may be inaccessible under Linux, because Linux does not include full support for NTFS. For example, open-source NTFS-3G NTFS driver does not support random write access to compressed files.

Paragon NTFS & HFS+ drivers for Linux solves these problems — now everyone can access NTFS and HFS+ partitions from Linux in a usual manner with maximum performance and reliability. The driver allows mounting NTFS and HFS+ partitions, so that programs may work transparently with these mounted partitions — browse contents, open documents, run applications, work with existing files (delete/copy/modify) and create new ones.

Paragon combined NTFS & HFS+ driver for Linux is commercial Linux driver for local access to NTFS and HFS+ volumes. It supports full read/write access. The driver is a Kernel module, which guarantees rapid and transparent access to supported file systems. Mount volumes manually or insert into **fstab**, and NTFS & HFS+ partitions will be available like any other directory tree.

Paragon NTFS & HFS+ Professional also includes Paragon LDM driver for Linux, that provides the ability to access all kinds of Microsoft Dynamic Disks (simple, mirror, spanned, stripe and RAID5) under Linux platforms, and useful additional utilities that provide the ability to check integrity, create/wipe/defrag NTFS volumes, perform many NTFS file system related tasks and copy (backup) files, saving all their attributes, between NTFS and native Linux file systems.

1.3 Paragon UFSD Technology

UFSD (Universal File System Driver) is an unique technology developed by Paragon Software to provide full access (read/write, format, etc.) to volumes of the popular file systems: NTFS, FAT, Ext2Fs, HFS, etc. under various platforms, including Windows, Linux, Mac OS X, etc. in case these file systems are not otherwise supported.

UFSD technology provides access directly to the physical devices that is why it can process partitions regardless of their support by the current OS. With UFSD it is possible to mount NTFS and HFS+ partitions under Linux, thus getting access to its contents, just the way it is implemented in the NTFS/HFS+ for Linux driver, and the technology also allows direct access via physical device addressing, the way it is implemented in the driver too.

Paragon UFSDs are designed to be readily integrated into any solution using our UFSD Software Development Kit (UFSD SDK), which includes all of the necessary tools to develop applications with the following main features:

- Access to un-mounted partitions (i.e. drive letter not assigned);
- Access to other file systems that normally would not be supported by the operating system;
- Platform-independent UFSD API.

Note: NTFS and HFS+ drivers for Linux as well as LDM driver and utilities were written using UFSD SDK.

1.4 How UFSD works on Linux

Modern operating systems are based on the concept of Installable File System drivers (IFS). User simply needs to provide an operating system with the proper file system driver to work with the file system in usual manner. Paragon NTFS & HFS+ for Linux includes NTFS, HFS+ and LDM drivers for Linux environment. Once appropriate components of Paragon NTFS & HFS+ for Linux are installed, the operating system can mount these file systems and work with directories/files stored on the file systems.

1.5 Key Features

Paragon NTFS & HFS+ for Linux Combo 8.1 is released in the Express and Professional Editions. All of the products share the following features:

- Transparent read-write access to NTFS and HFS volumes — single Kernel module provides support both NTFS and HFS+ file systems
- High performance (in some cases even better than Ext3 FS);
- Easy installation and unistallation (assistant scripts);
- Support for the latest Linux Kernels and distributions;
- Support for SMP kernels;
- File sharing over network via SAMBA;
- No system degradation during data transfers;
- Unlimited file and volume size (within NTFS/HFS+ and Kernel limitations).

What's new in Paragon NTFS & HFS+ for Linux 8.1:

- Several codepages for filename translation are supported simultaneously;
- Full interoperability with Mac OS X SAMBA clients;
- Full interoperability with P2P (BitTorrent) software;
- All known bugs are fixed.

NTFS-specific features:

- Full support for compressed files (random access for reading and writing with no limitations);
- Sparse files;
- Alternate data streams;

NTFS compatibility information:

File system version	Comments
NTFS version 1.2	Originates from Microsoft Windows NT 4.0
NTFS version 3.0	Originates from Microsoft Windows 2000
NTFS version 3.1	Originates from Microsoft Windows XP/2003 and Vista

Additional features of the Professional edition:

- Support for all kinds of Microsoft Dynamic Disks (simple, mirrored, spanned, striped and RAID5) — support for LDM (Logical Disk Manager);

- Support for encrypted files copying (cpntfs utility);

Additional features of the Professional Edition:

- Additional NTFS utilities:
  - **mkntfs** utility - format any partition as NTFS under Linux;
  - **chkntfs** utility - check NTFS partition integrity and fix errors;
  - **infntfs** utility - show detailed information about NTFS partitions;
  - **dfntfs** utility - defragment a NTFS volume;
  - **wipe** utility - fill with zeros free space on a NTFS/FAT volume;
  - **mftpack** utility - pack/truncate MFT (Master File Table) on a NTFS volume;
  - **hdlnk** utility - enumerate all hard links on a NTFS volume;
  - **junction** utility - show reparse points on a NTFS volume;
  - **fsutil** utility - perform many NTFS file system related tasks. Powerful utility;
  - **cpntfs** utility - create an archive of the NTFS volume or separate files/directories including all streams and attributes.
- Additional HFS+ utilities:
  - **mkhfs** utility - format any partition as HFS+ under Linux;
  - **chkhfs** utility - check HFS+ partition for integrity and fix errors;

2. System Requirements

Minimum hardware requirements:

- Processor: Intel Pentium 300 MHz and higher, or compatible;
- both 32- and 64-bit CPUs are supported.
- 16MB of RAM.

Due to unique technology our NTFS/HFS+ for Linux drivers have low system requirements. For example, it is enough for our driver to have 500KB of free RAM to work with NTFS partitions larger than 250 GB. Combined NTFS & HFS+ Kernel module itself occupies around 430 Kb of RAM.

Supported Linux kernels:

- Linux with kernel versions 2.4.x;
- Linux with kernel versions 2.6.x (NTFS/HFS+ drivers were tested with Kernels up to 2.6.32.22, LDM driver was tested with Kernels up to 2.6.31).

Linux distributions the products were tested with:

- Ubuntu 9.10, 10.04
- OpenSUSE 11.2
- Manrdiva Free 2010
- Debian 5.04
- LinuxMint 8
- Slackware 13
- Fedora Core/Fedora 3, 4
- CentOS 5.4

2.1 Development Environment

A development environment is required to compile Linux drivers and utilities. Please verify that these tools are all functional. The easiest way is to choose the developer toolkit when installing Linux.
What must be installed:

- Kernel source code (recommended) or Kernel header files (doesn't always work);

  ```
  #rpm -qa|grep kernel-source (for RPM based kernel-sources)
  ```
- GNU C compiler (GCC);

  ```
  #gcc --version
  ```
- GNU C++ compiler (g++) — for Professional version only;

  ```
  #g++ --version
  ```
- GNU Make;

  ```
  #make --version
  ```
- GNU ld (binutils);

  ```
  #ld --version
  ```
- Modutils (module-init tools);

  ```
  #insmod -V
  ```

2.2 Limitations

- GNU C compiler (gcc) version 2.95 or higher is required.
- The user should login as root to install the drivers and utilities.

- Correct operation is not guaranteed when using Linux with kernel versions 2.3.x and 2.5.x (which are known for their instability).
- Correct operation is not guaranteed for customized Linux kernels. Commercial porting service to customized Linux kernels is available from Paragon Software Group — for more information send e-mail to sales@paragon-software.com).

3. Installation

3.1 Shipment

The setup files for each product of the family are provided as the downloadable **TGZ** archives, which can be downloaded from the company site.

3.2 Components

The package includes the following components:

- The source files for the NTFS and HFS+ for Linux driver(s);
- The source files for additional utilities (for Professional edition only);
- The source files for the LDM driver (for Professional edition only);
- Assistant script files, which are purposed to simplify the installation and uninstallation routines.

Paragon NTFS and LDM Linux drivers and utilities must be compiled on the end user’s system for correct configuration. These modules are the open source code with libraries. Before installing the modules, one must build drivers and utilities by using the GNU development tools listed above.

3.3 Installing the Drivers

First, NTFS & HFS+ Combo driver must be built and installed. After that LDM driver may be installed (for Professional Edition only).

Steps to install the NTFS & HFS+ for Linux and LDM drivers are as follows:

1) Log in as root. This step is obligatory;
2) Build and install the NTFS & HFS Combo driver and LDM driver using install.sh script. Alternatively, driver binary module may be built manually using 'make' command.
3) Install the NTFS & HFS Combo driver and LDM driver (if needed) (this step will make the modules available for use);

4) Activating (loading) the driver. After building and installing, the NTFS & HFS Combo driver can be referenced as "used file system driver" when mounting NTFS and HFS+ partitions.

The steps 1-3 should be made only once while the step 4 is the standard way of using file system drivers in Linux environment.

NTFS & HFS for Linux as well as LDM driver include a set of assistant script files for the simplification of building, installing and uninstalling procedures. Note that these assistant scripts may fail to work in customized Linux configurations or unsupported Linux distributions.

Use **install.sh** and **uninstall.sh** script files to install and uninstall (correspondingly) NTFS & HFS+ combo and LDM drivers and utilities. The sections below describe the installation procedure in details.

3.3.1 Unpacking Setup Files

The setup files of the NTFS & HFS for Linux and LDM drivers are provided in the form of **TGZ** archives. The archives should be copied on a hard disk and decompressed. Unpack the archive files to directories using, for example, the following commands:

For the NTFS for Linux driver and utilities:

```
tar zxC /usr/tmp -f /mnt/cdrom/NtfsForLinux/ntfslin_drv.tgz
```

or

tar xzf /usr/tmp/ntfslin_drv.tgz – in case you have already copied the **TGZ** archive to the **/usr/tmp/** directory.

For the LDM driver:

```
tar zxC /usr/tmp -f /mnt/cdrom/LDM_drv.tgz
```

or

tar xzf /usr/tmp/LDM_drv.tgz – in case you have already copied the **TGZ** archive to the **/usr/tmp/** directory.

Next, change the current directory to the **/usr/tmp**:

```
cd /usr/tmp
```

Next actions are to build and install the NTFS for Linux, LDM drivers and additional utilities.

3.3.2 Using the INSTALL.SH Assistant Script

The assistant **install.sh** scripts provide easy and flexible way to build combined NTFS & HFS+ and LDM drivers, install them in the system and mount all or selected NTFS partitions as well as dynamic volumes which currently exist on local system. Additionally, the script configures all NTFS partitions to be mounted automatically at system startup.

However, **install.sh** script requires that development tools and kernel sources present on end-user system in their default locations.

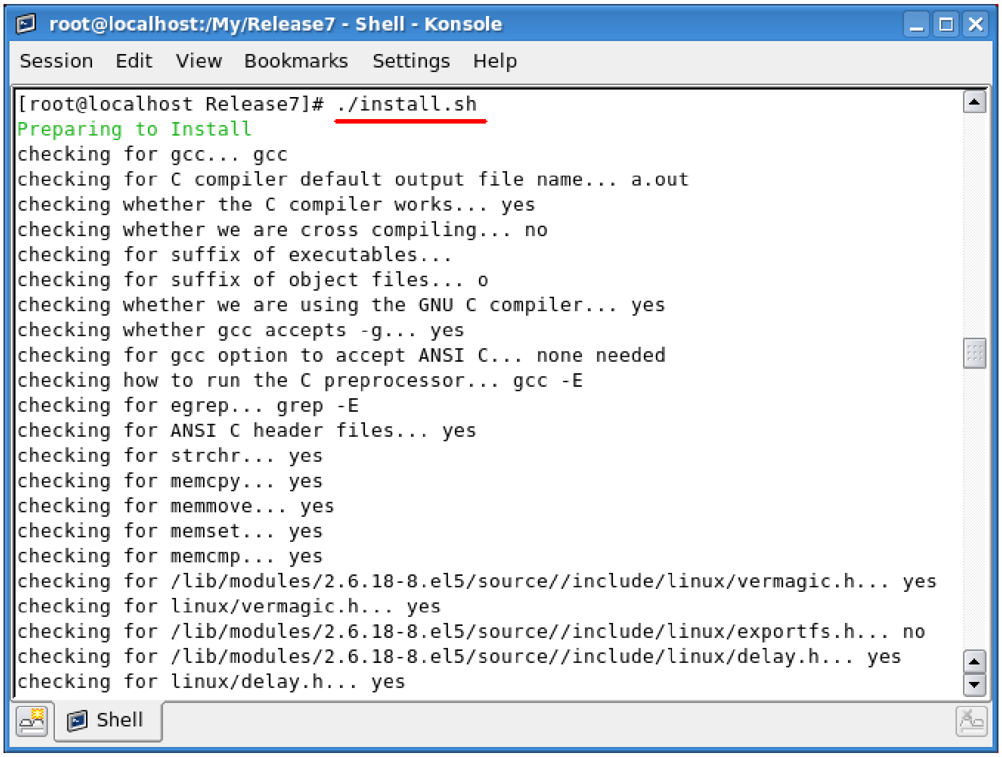
Installation

Just run the **install.sh** script:

./install.sh

The assistant script will automatically perform the following actions:

1) Detect the Linux type and kernel version;
2) Find kernel header files, kernel-config file and libraries needed for building the drivers;
3) Build driver and utilities as (binary modules);
4) Install driver and utilities;
5) Detect all NTFS and dynamic partitions on all local hard disks, mount all NTFS partition;
6) Reconfigure the file **/etc/fstab** to automatically mount NTFS and HFS+/HFSX partition at Linux startup;


root@localhost:/My/Release7 - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost Release7]# ./install.sh
Preparing to Install
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
checking how to run the C preprocessor... gcc -E
checking for egrep... grep -E
checking for ANSI C header files... yes
checking for strchr... yes
checking for memcpy... yes
checking for memmove... yes
checking for memset... yes
checking for memcmp... yes
checking for /lib/modules/2.6.18-8.el5/source//include/linux/vermagic.h... yes
checking for linux/vermagic.h... yes
checking for /lib/modules/2.6.18-8.el5/source//include/linux/exportfs.h... no
checking for /lib/modules/2.6.18-8.el5/source//include/linux/delay.h... yes
checking for linux/delay.h... yes
```

INSTALL.SH default mode for the NTFS/HFS+ for Linux driver

- The assistant script **install.sh** always names the NTFS/HFS+ for Linux driver module as **ufsd** (it is the abbreviation of the project name **Universal File System Driver**);
- The assistant script **install.sh** always mounts NTFS partitions to directories named like "/mnt/ntfs_0", "/mnt/hfsp_1", "/mnt/hfsx_1" etc.

INSTALL.SH default mode for the LDM driver

- The assistant script **install.sh** always names the LDM driver as **ufsdldm**;
- The assistant script **install.sh** always mounts dynamic partitions to directories named like "/mnt/dyn_0, /mnt/dyn_1" etc.

Now you can mount any NTFS/HFS+ partition: mount -t ufsd <device> <mount_point>.

Note: The /lib/modules/kernel_version/extra/ or /lib/modules/kernel_version/kernel/fs/ufsd directory will contain the ufsd.ko kernel binary module.

4. Uninstalling the Drivers

To completely remove the drivers and the utilities from the system, one should dismount all NTFS/HFS+ and dynamic partitions mounted with the driver, uninstall the drivers and optionally remove binary files.
NTFS/HFS+ for Linux provides tools for the drivers/utilities uninstall automation.

The assistant script **uninstall.sh** completely removes the drivers/utilities from the system, including unmounting all NTFS/HFS+ and dynamic partitions.

4.1 Using the UNINSTALL.SH Assistant Script

The assistant script **uninstall.sh** provides the extremely easy and flexible way to deactivate and remove the drivers and utilities from the system. The script performs the correct deactivation, uninstallation and the complete removing of the driver's and utilities' files.

4.1.1 Uninstalling

Just run the **uninstall.sh** script:

```
./uninstall.sh
```

The assistant script will automatically perform the following actions:

1. Unmount all currently mounted NTFS/HFS+ and dynamic partitions. Additionally, the script removes the appropriate mount-points and deletes reference to these partitions from the **fstab**. If some NTFS/HFS+ and dynamic partitions are in use, the script (for the NTFS/HFS+ or LDM driver) will not unmount these partitions. The further script execution is aborted in this case;
2. Deactivate the driver modules. If the drivers is still in use, the further script execution is aborted;
3. Uninstall the drivers;
4. Remove all binary and source files of the driver and utilities.

5. Using NTFS/HFS+ for Linux driver and LDM Driver

After building and installing the NTFS/HFS+ for Linux driver, it can be automatically loaded at the system startup. The driver allows to mount NTFS/HFS+ partitions and to get a plain access to their contents.

At loading the LDM driver scans and initialises mountable block devices in /dev/dm directory with names v0...vN. If **devfs** has not been compiled in, directory and device nodes should be created manually by calling (in /dev/dm) **mkdev vN b 212 N** with **N** in 0..max, where max is a reasonable limit but no more than 128.

/dev/dm/vN can be mounted in the usual way by using the mount command.

Devices containing dynamic disks and detected volumes are listed in system log during load.

After loading the driver you can mount and umount dynamic volumes by mount and umount commands.

5.1 Mounting NTFS/HFS+ Partitions

To gain access to a NTFS/HFS+ partition, use standard **mount** command with a file system type set to **ufsd**. For example:

```
mount –t ufsd /dev/hdb1 /mnt/ntfs
```

5.2 Mount Options for the NTFS/HFS+ for Linux Driver

SYNOPSYS
mount –t ufsd [-o options] <device> <mount point>

Option	NTFS	HFS+	Expected behavior

iocharset
or
nsl
or
codepage
	●	●	**`-o iocharset={NAME1}[,iocharset={NAME2}]`**
			`-o nls={NAME1}[,nls={NAME2}]`
			`-o codepage={NAME1}[,codepage={NAME2}]`
			The NTFS/HFS+ file systems store all file/directory names in Unicode format (UTF-16), which can represent any character from any language. In case none of these options is set, the default codepage will be used (**`CONFIG_NLS_DEFAULT`**). If none of the specified codepages exist on the system, the default codepage will be used again. This option informs the driver how to interpret path strings and translate them to Unicode and back. Up to 8 different code pages can be specified. The driver tries to use the codepages from specified list in order until it manages to translate all the characters in the string. If none of the specified codepages allows to translate all the characters, Kernel's default codepages is used[1].
			Note: Paragon driver uses extended UTF-8 for Unicode number U+10000 characters support when '**`=utf8`**' is specified.

nocase	●		**`-o nocase`** All file and directory operations (open, find, rename) are case insensitive. Casing is preserved in the names of existing files

[1] That is, codepage specified by **`CONFIG_NLS_DEFAULT`** Kernel configuration option.

Option	NTFS	HFS+	Expected behavior
			and directories.
showmeta	●	●	**-o showmeta** Use this parameter to show all meta-files (System Files) on a mounted NTFS/HFS+ partition. By default, all meta-files are hidden.
noatime	●	●	**-o noatime** All files and directories will not update their last access time attribute if a NTFS/HFS+ partition is mounted with this parameter. This option can speed up file system operation.
uid	●	●	**-o uid={USERID}** By default all files on a mounted NTFS/HFS+ volume are owned by root. By specifying the uid parameter you can set an owner of files. The userid can be any name from **/etc/passwd**, or any number representing a user id.
gid	●	●	**-o gid={GROUPID}** By default all files on a mounted NTFS/HFS+ volume are owned by group root. By specifying the gid parameter you can set a owner group of the files. The groupid can be any name from **/etc/group**, or any number representing a group id.
umask	●	●	**-o umask={VALUE}** The default permissions given to a mounted NTFS/HFS+ volume are **rwx------** (for security reasons). The **umask** option controls these permissions for files/directories created after the volume is mounted. **mount –t ufsd /dev/hda1 /mnt/ntfs_0 –o umask=0222**
fmask **dmask**	●	●	**-o fmask={VALUE}** **-o dmask={VALUE}** **umask** option changes the permissions for new created files and directories; **fmask** is applied to files; **dmask** to directories that already exist on a mounted volume. The effect of these options can be combined. To mount Samba, FTP or NFS shares the combination of **umask=000,fmask=000,dmask=000** is usually specified.

Option	NTFS	HFS+	Expected behavior
`ro`	●	●	To mount a NTFS/HFS+ volume in read-only mode.
`bestcompr`	●		Instructs the driver to use highest compression level when writing compressed files. High CPU-load.
`nobuf`	●	●	Disables buffered read/write operations for metadata and directories. Useful option for embedded device with little memory (<64MB).
`sparse`	●		Create new files as “sparse”. This feature allows creating holes inside new created files (avoids filling unwritten space with zeroes). This option is useful in case NTFS partition is used for BitTorrent downloads. For more information see Developer’s Reference Manual.
`force`	●	●	Forces the driver to mount partitions even if ‘dirty’ flag (volume dirty) is set. It is recommended to use Paragon or OS-specific file system checking utility before mounting ‘dirty’ partitions to reset the ’dirty’ flag.
`nohidden`	●		Files with the Windows-specific HIDDEN attribute will not be shown under Linux.
`sys_immutable`	●		Files with the Windows-specific SYSTEM attribute will be marked as system immutable files.
`clump`	●		`-o clump={size}` Driver will pre-allocate space up to "size" in Kbytes during file extension operation. Preallocated space will be aligned up to the cluster size. This space will be preallocated, but the file size information will only show the real written size. This speeds up file write operations if write function is called with small buffer size, but will enlarge files.

5.3 Mounting Partitions With ‘Dirty’ Flag Set

Both NTFS and HFS+ file system have special feature called ‘dirty flag’ that allows detecting incorrectly removed partitions that possibly contain errors or inconsistencies. UFSD driver refuses to mount such partitions before dirty flag is reset. **chkntfs/chkhfs** utility with -a -f command line options can be used to reset the dirty flag. Alternatively, there is ‘force’ mount option that forces the driver to ignore ‘dirty’ flag and mount the volume anyway.

5.4 Unmounting NTFS/HFS+ Partitions

To unmount a NTFS partition, use the standard command **umount**. For example:

```
umount /dev/hdb1
```

5.5 Unmounting Dynamic Partitions

To unmount a dynamic partition, use the standard command **umount**.

```
umount /mnt/dyn_1
```

or

```
umount /dev/dm/v1
```

5.6 Choosing the codepage/charset for NTFS/HFS+ Partitions

The format of filenames on NTFS/HFS+ partitions differs from text standard presentation used in Linux. To accommodate NTFS/HFS+ standards to Linux ones, character translation is required. The character translation uses **charset** or **codepage** information for correct translation non-English characters between NTFS/HFS+ and Linux.

Unfortunately Linux is unable to automatically detect NTFS/HFS+ codepage/charset settings. For this reason, the user must assign character set for filenames translation manually.

The standard Linux command **mount** allows choosing the character set that is used for the filenames translation, the **iocharset** parameter is used for this purpose.

iocharset parameter of **install.sh** script provides the ability to define the character set for all automatically mounted partitions. One should realize that character set assigned to the driver should conform to the actual locale settings in Linux. Otherwise, non-English filenames will remain unreadable.

Examples:

1. Mounting a partition:

```
mkdir /mnt/test
mount -t ufsd /dev/hda6 /mnt/test
```

2. Dismounting a partition:

```
umount /mnt/test
```

3. Mounting partition in **read-only** mode:

```
mount -t ufsd -o ro /dev/hda6 /mnt/test
```

4. Choosing the character set to be used with NTFS partitions, when installing Linux driver:

```
./install.sh --iocharset=utf8
```

5. Choosing character set to be used with NTFS/HFS+ when mounting partitions manually:

```
mount -t ufsd -o iocharset=koi8-r /dev/hdb1 /mnt/test
```

6. Troubleshooting

6.1 The install.sh script can't find kernel sources

1. Read system requirements section, make sure all tools are functional. For more information, please read kernel documentation.
2. Linux kernel must be configured correctly.
3. Make sure that you have kernel sources, for example, in the **/usr/src/linux-x.x.xx** directory, where **x.x.xx** is your kernel version (for example, 2.6.10). Type **uname-r** in the command line to know your current kernel version.
4. Create a symbolic link from the **/usr/src/linux-x.x.xx** directory to **/usr/src/linux**. To create the link type **ln –s /usr/src/linux-$(uname-r) /usr/src/linux .**
5. Make sure that you have the **config-x.x.xx** file, for the booted Linux kernel, in the **/boot** directory. If you haven't the **config-x.x.xx** file then type **ln –s /usr/src/linux-$(uname-r)/.config /boot/config-$(uname –r)** to create a symbolic link to the config file.

Note: There are cases when the kernel sources may be located in other directories. In these cases you should create a symbolic link to **/usr/src/linux**, for example, **ln –s /lib/modules/$(uname-r)/build /usr/src/linux .**

If you still have the same problem i.e. the **install.sh** script can't find the kernel sources it is better to rebuild your kernel or download and build a stable kernel from the www.kernel.org site.

6.2 Can't compile the NTFS/HFS+ for Linux driver

1. Read System requirements section, make sure all tools are functional. For more information, please read kernel documents.
2. Linux kernel must be configured correctly.
3. The **/boot** directory must contain the **config-(kernel version)** file. If the file is missing you should execute the following command: **ln –s /usr/src/linux-$(uname-r)/.config /boot/config-$(uname –r)**.

6.3 "Can't load module" message at the end of installation

1. Make sure that you use the same version of GCC compiler that was used for kernel compilation.
2. Make sure that the **Makefile** of the kernel (you can find the **Makefile** in the directory where the kernel sources are located) have the correct kernel version at the beginning of the file. For example: if your loaded kernel version is **2.6.11-6mdksmp** then the following lines must be found at the beginning of the **Makefile**:

```
VERSION = 2
PATCHLEVEL = 6
SUBLEVEL = 11
EXTRAVERSION = -6mdksmp
```

6.4 ufsd Module: kernel-module version mismatch

That means kernel version mismatch.

1. Check kernel source version in **/usr/src/linux/include/linux/version.h**
2. Check the currently running kernel version: **uname -r**
3. Both version must match.
4. If they don't match, please restore Kernel configuration or recompile kernel (advanced).

6.5 ufsd Module: create_module: operation is not permitted

That means you must have root privilege to load driver.

6.6 insmod: a module named as ufsd already exists

That means driver have been loaded. There is no need to load it again. Driver status can be found by using the following command: **lsmod | grep ufsd**

6.7 When I run the “insmod ufsd.o” command, there are some error messages

1. Make sure you are trying to install a module for this kernel.
2. Generally the same **ufsd** binary module works with both **smp** and **non-smp** kernels, but there are exceptions to the rule, please note this.
3. Please note that **ufsd.o** is for 2.4.x kernels, while **ufsd.ko** is for 2.6.x kernel.

6.8 I can’t mount NTFS/HFS+ volume

1. Make sure that the driver is activated (loaded into the Kernel): **lsmod | grep ufsd**
2. Make sure that the driver supports file system mounted partition is formatted with:

   cat /proc/fs/ufsd/version
3. The volume is dirty. Use chkntfs/chkhfs utility with –a –f command line options to reset ‘dirty’ flag. Alternatively, use ‘force’ mount options to make the driver ignore ‘dirty’ flag.

7. Limitations

7.1 NTFS limitations

1. Encrypted files not supported. Body of file will be copied encrypted with loss of decryption capability. To make a full archive **cpntfs** utility (see the cpntfs chapter) can be used (available in the Professional Edition).
2. When copying from NTFS to Linux FS: all additional streams will not be copied, along with compression flag and security attributes (use the cpntfs utility to preserve this information).
3. Hardlinks and symlinks: any link will be copied as a full file with its body, losing link information.

7.2 HFS limitations

1. Extended attributes are not yet supported;
2. NFS on HFS+ is not yet supported;
3. Alternate streams (forks) are not yet supported;

8. Legal question

8.1 NTFS Legal Questions

Paragon NTFS for Linux driver is absolutely legal. It does not violate any patents and/or intellectual property rights. It is well known that originally NTFS was very close to the HPFS file system developed by IBM. HPFS was much more OPEN in terms of documentation support, data structure and so on. It helped us to gain a better understanding of its nature, architecture and ideology. The knowledge about NTFS we also have got has already been used for years inside our best-seller product – **Paragon Partition Manager**. We have sold several million copies of **Paragon Partition Manager** all over the world. The stability of the products as far as NTFS related operations are concerned says for itself about the stability of the NTFS technology at all. Thus, having a pretty good idea about what the HPFS file system is, we may understand the way NTFS functions.

Applying to the other sources of information like Linux drivers for NTFS and debugging Windows applications, we've documented NTFS structures from within and finally created the Universal File System Driver.

While developing Paragon NTFS for Linux driver we always stuck to the following rules:

1) We never applied to any confidential Microsoft NTFS stuff (docs, codes, etc.) and the reverse engineering approach for MS code.

2) Open sources are the only thing we used. E.g. from **www.ntfs.com** we got the great part of our NTFS knowledge and understanding.

3) NTFS as a file system as well as on-disk layout is not patented and not documented.

8.1 HFS Legal Questions

Paragon HFS for Linux driver is absolutely legal. It does not violate any patents and/or intellectual property rights. HFS specifications are openly published by Apple Corporation on http://developer.apple.com/.

9. Additional Utilities

Additional utilities for Paragon NTFS/HFS+ for Linux provide the ability to check integrity and create NTFS/HFS+ volumes on block devices. Additional NTFS utilities allow to defragment, wipe, and perform many NTFS file system related tasks and copy (backup) files, saving all NTFS-specific data and attributes, between NTFS and native Linux file systems. Additional utilities for Paragon NTFS/HFS+ for Linux were developed with Paragon UFSD SDK.

9.1 NTFS utilities

There are 10 additional utilities for NTFS:

- **mkntfs** — format any partition as NTFS under Linux;
- **chkntfs** — check NTFS partition for integrity and (optionally) fix errors;
- **infntfs** — show detailed information about NTFS partitions;
- **dfntfs** — defragment a NTFS volume;
- **wipe** — fill with zeros free space on a NTFS/FAT volume;
- **mftpack** — pack/truncate MFT (Master File Table) on a NTFS volume;
- **hdlnk** — enumerate all hard links on a NTFS volume;
- **junction** — show reparse points on a NTFS volume;
- **fsutil** — perform many NTFS file system related tasks. Powerful utility;
- **cpntfs** — creates an archive of the NTFS volume or separate files/directories including all streams and attributes.

There are 2 additional utilities for HFS+:

- **chkhfs** — check HFS+ partition for integrity and (optionally) fix errors;
- **mkhtfs** — format any partition as HFS+ under Linux;

9.1.1 INFNTFS Utility - Show information about NTFS Volumes.

Name

infntfs – is intended for showing and changing common information about NTFS volumes.

Synopsis

infntfs [options] device

E.g.: infntfs --trace --verbose --label “New Volume” --dirty clear --serial AAAAAAAA-BBBBBBBB /dev/hdb1;

E.g.: infntfs /dev/hdb1.

Options

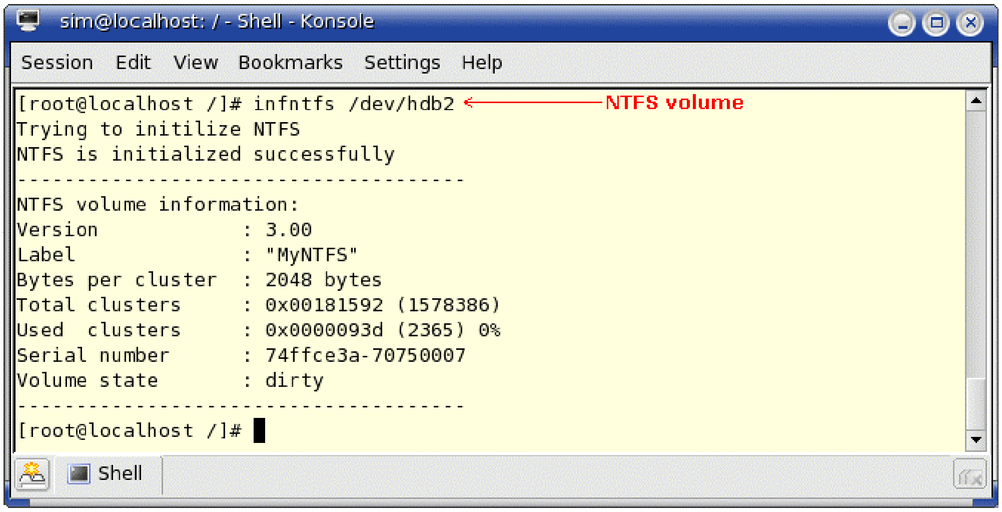
--label label	Set new volume label.
--dirty set	Set dirty flag.
--dirty clear	Clear dirty flag.
--serial lo-high	Set a new serial number (in hex).
--trace	Turn on UFSD trace.
--verbose	Explain what is being done.
--help	Display this help.

Description

infntfs shows NTFS volume label, used space, dirty flag, version, serial number and allows to change NTFS volume label, dirty flag and serial number.

Screenshots

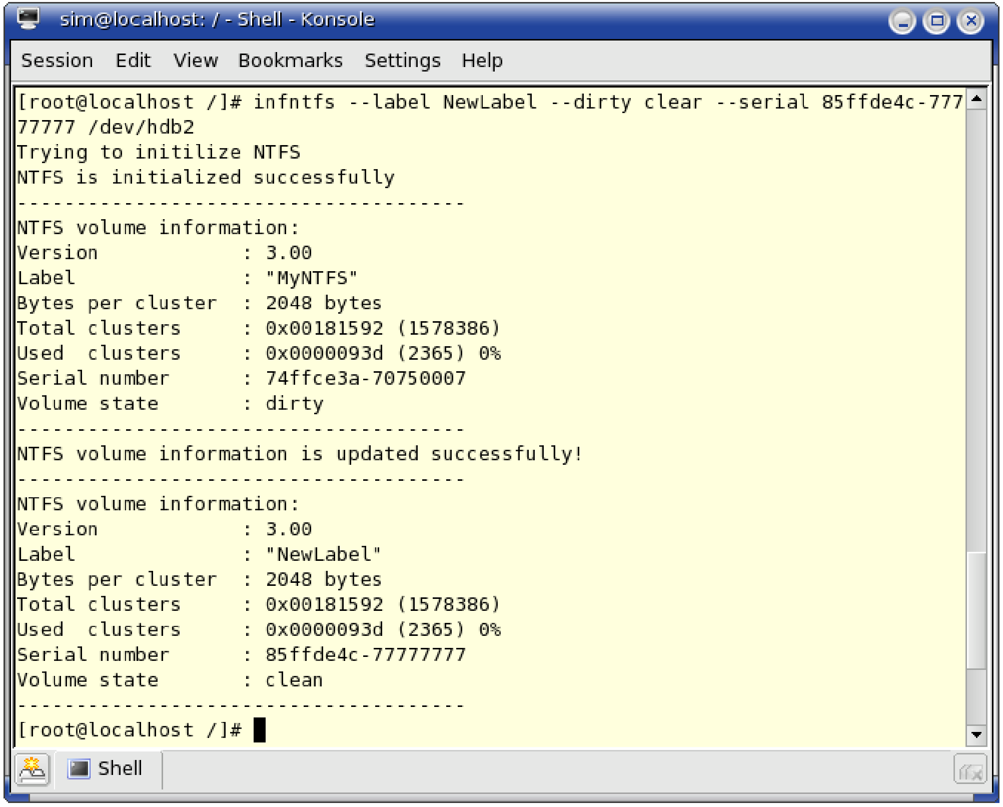
1. Showing common information about NTFS volume:



sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
NTFS volume
Shell

```
[root@localhost /]# infntfs /dev/hdb2
Trying to initilize NTFS
NTFS is initialized successfully
--------------------------------------
NTFS volume information:
Version            : 3.00
Label              : "MyNTFS"
Bytes per cluster  : 2048 bytes
Total clusters     : 0x00181592 (1578386)
Used  clusters     : 0x0000093d (2365) 0%
Serial number      : 74ffce3a-70750007
Volume state       : dirty
--------------------------------------
[root@localhost /]#
```

2. Changing common information about the NTFS volume:


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# infntfs --label NewLabel --dirty clear --serial 85ffde4c-777
77777 /dev/hdb2
Trying to initilize NTFS
NTFS is initialized successfully
--------------------------------------
NTFS volume information:
Version            : 3.00
Label              : "MyNTFS"
Bytes per cluster  : 2048 bytes
Total clusters     : 0x00181592 (1578386)
Used  clusters     : 0x0000093d (2365) 0%
Serial number      : 74ffce3a-70750007
Volume state       : dirty
--------------------------------------
NTFS volume information is updated successfully!
--------------------------------------
NTFS volume information:
Version            : 3.00
Label              : "NewLabel"
Bytes per cluster  : 2048 bytes
Total clusters     : 0x00181592 (1578386)
Used  clusters     : 0x0000093d (2365) 0%
Serial number      : 85ffde4c-77777777
Volume state       : clean
--------------------------------------
[root@localhost /]#
```

9.1.2 CHKNTFS Utility - Perform consistency checks on a NTFS volume.

Name

chkntfs - provide consistency checking of a NTFS volume and fixing errors.

Synopsis

chkntfs device [options]

E.g.: chkntfs /dev/hdb1

Options

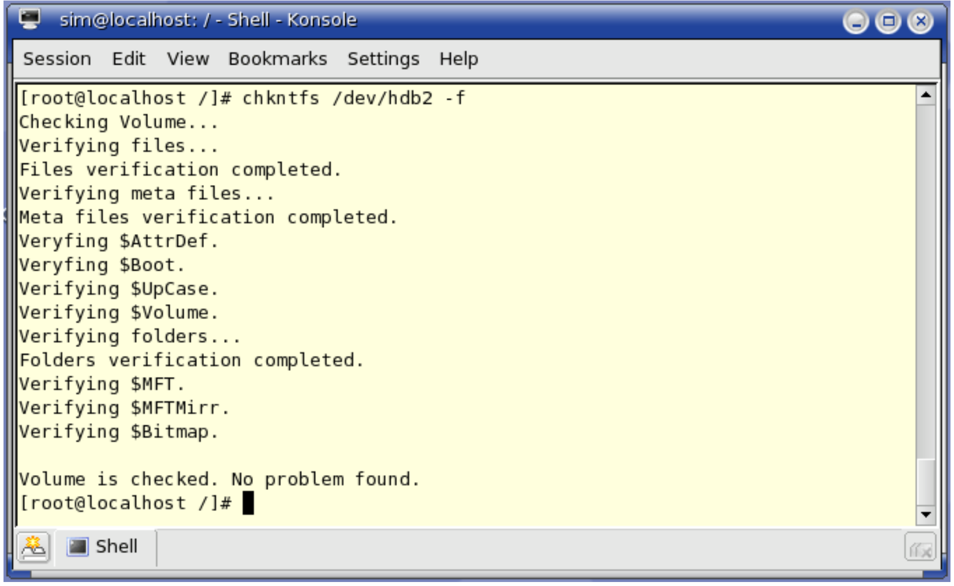
-f	Fix errors on the disk.
-a	Perform checks only if ‘dirty’ flag is set.
-h	Display this help.
--trace	Turn on UFSD trace.
--verbose	Explain what is being done.
--version	Show the version and exit.

Description

chkntfs creates and displays a status report about a NTFS file system. **Chkntfs** also lists and corrects errors on the disk, if any (-f flag must be specified).

Screenshots

Verifying and fixing errors on the specified partition:



```
[root@localhost /]# chkntfs /dev/hdb2 -f
Checking Volume...
Verifying files...
Files verification completed.
Verifying meta files...
Meta files verification completed.
Veryfing $AttrDef.
Veryfing $Boot.
Verifying $UpCase.
Verifying $Volume.
Verifying folders...
Folders verification completed.
Verifying $MFT.
Verifying $MFTMirr.
Verifying $Bitmap.

Volume is checked. No problem found.
[root@localhost /]#
```

9.1.3 MKNTFS Utility — Create a NTFS volume on a partition.

Name

mkntfs - create a NTFS volume (1.2, 3.0, 3.1 (Windows NT 4.0/2000/XP/2003/Vista) file system) on a user specified (block) device under Linux OS.

Synopsis

mkntfs [options] device

E.g.: mkntfs /dev/hdb1

Options

-v:label	Specify the volume label.
-q	Perform a quick format.
-c	Files created on the new volume will be compressed by default.
-a:size	Override the default allocation unit size. Default settings are strongly recommended for general use. NTFS supports 512, 1024, 2048, 4096, 8192, 16K, 32K, 64K.

NTFS compression is not supported for allocation unit sizes above 4096.

-f Force the format without confirmation.

-s:start **S**pecify “hidden” sectors in the boot area.

-g:tracks:sectors Specify the disk geometry that should be written in the boot area.
“tracks” – Specify the number of tracks per disk side.
“sectors” – Specify the number of sectors per track.
The most known geometries are:
NORMAL: 63 sectors per track and 15(16) tracks per cylinder.
LBA: 63 sectors per track and 255 tracks per cylinder.
In general Windows uses the LBA geometry (-g:255:63)
If **–g** is not specified this program gets geometry from Linux.

--help Display this help.

--trace Turn on UFSD trace.

--verbose Explain what is being done.

--version Show the version and exit.

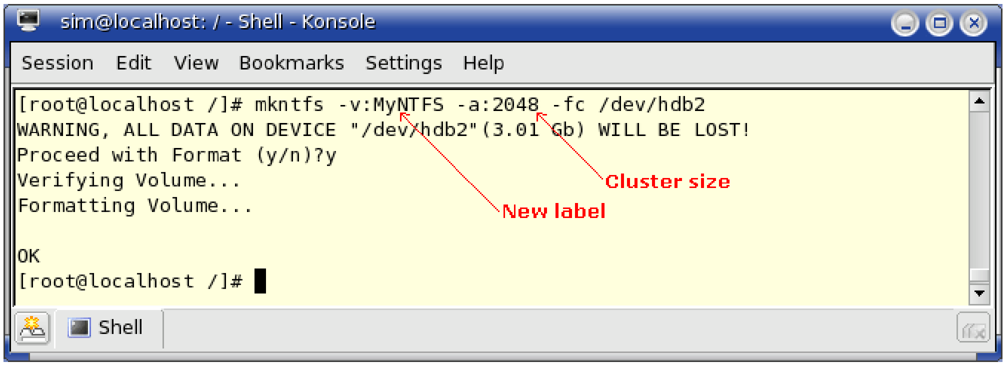
Description

mkntfs is a standalone utility that allows to format NTFS partitions under Linux. It is used to create a NTFS 1.2, 3.0, 3.1 (Windows NT 4.0/2000/XP/2003/Vista) file system on a device (usually a disk partition).

Note: mkntfs doesn’t change the MBR (Master Boot Record) when formatting a partition. It follows that most of Linux commands (like **fdisk -l**) will not define that the partition’s files system was changed to a NTFS one.

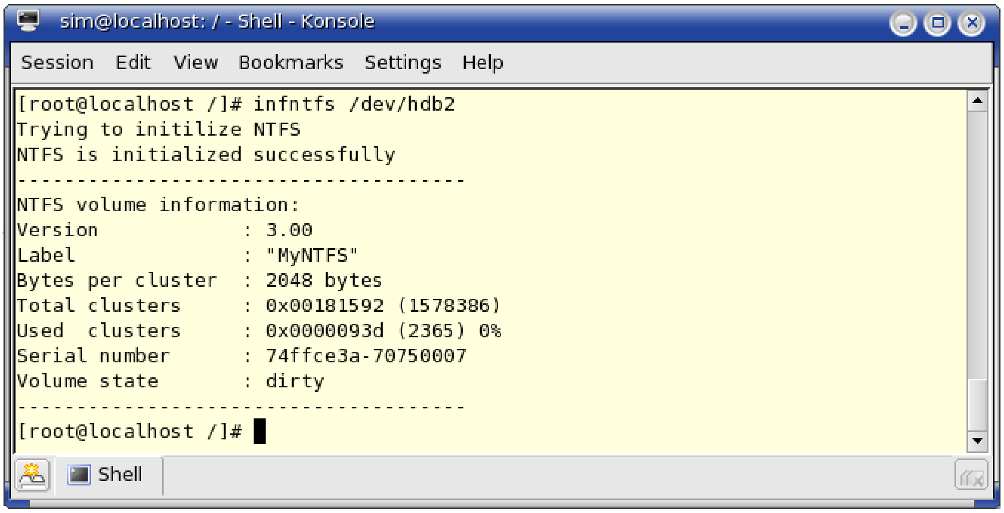
Screenshots

Making NTFS partition:


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Cluster size
New label
Shell

```
[root@localhost /]# mkntfs -v:MyNTFS -a:2048 -fc /dev/hdb2
WARNING, ALL DATA ON DEVICE "/dev/hdb2"(3.01 Gb) WILL BE LOST!
Proceed with Format (y/n)?y
Verifying Volume...
Formatting Volume...

OK
[root@localhost /]#
```

Result:


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# infntfs /dev/hdb2
Trying to initilize NTFS
NTFS is initialized successfully
--------------------------------------
NTFS volume information:
Version            : 3.00
Label              : "MyNTFS"
Bytes per cluster  : 2048 bytes
Total clusters     : 0x00181592 (1578386)
Used  clusters     : 0x0000093d (2365) 0%
Serial number      : 74ffce3a-70750007
Volume state       : dirty
--------------------------------------
[root@localhost /]#
```

9.1.4 DFNTFS Utility – Defragment a NTFS volume.

Name

dfntfs – defragment a NTFS volume (1.2, 3.0, 3.1 (Windows NT 4.0/2000/XP/2003/Vista) file system) on a user specified (block) device under Linux OS.

Synopsis

dfntfs [options] device

E.g.: dfntfs –s- /dev/hdb1

Options

-t+ Creation time increasing (sort files and directories according to their “creation time” attribute in ascending order);

-t- Creation time decreasing;

-s+ File size increasing (sort file and directories according to their “file size” attribute in ascending order);

-s- File size decreasing;

-d+ Directory first (place directories ahead files);

-d- Directory last;

-l+ Start cluster increasing (the order (according to the start cluster) of files and directories will be preserved and they will be placed continuously);

-l- Start cluster decreasing (files and directories will be placed in the reserved sequence order and continuously);

--help Display help;

--trace Turn on UFSD trace;

--verbose Explain what is being done;

--version Show version and exit.

Description

Defragmentation is the process of rewriting parts of a file to contiguous sectors on a hard disk to increase the speed of access and retrieval. When files are updated, the computer tends to save these updates on the largest continuous space on the hard disk, which is often on a different sector than the other parts of the file. When files are thus fragmented, the computer must search the hard disk each time the file is opened to find all of the file’s parts, which slows down response time.

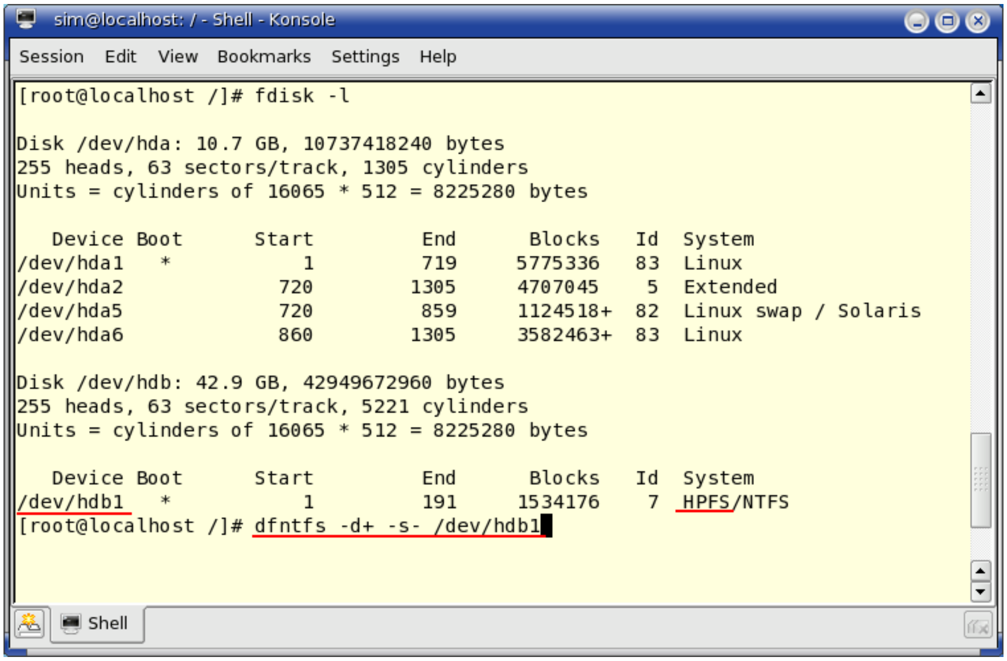
This dfntfs utility provides the necessary functionality for the defragmentation of NTFS partitions.

Screenshots

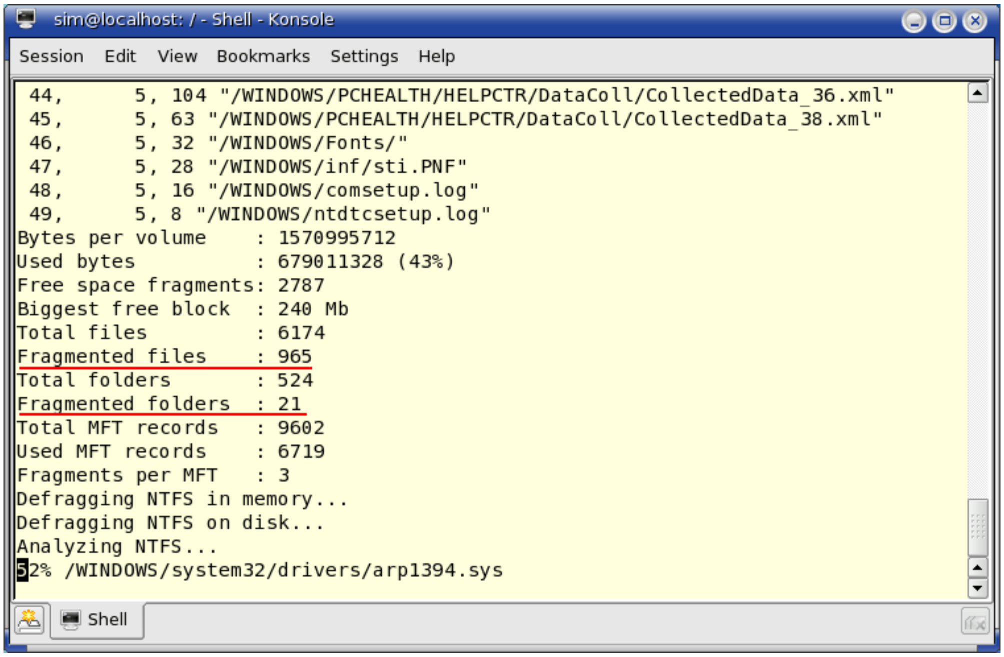
Let’s defragment a NTFS partition in the following way:

1. Place directories ahead files;

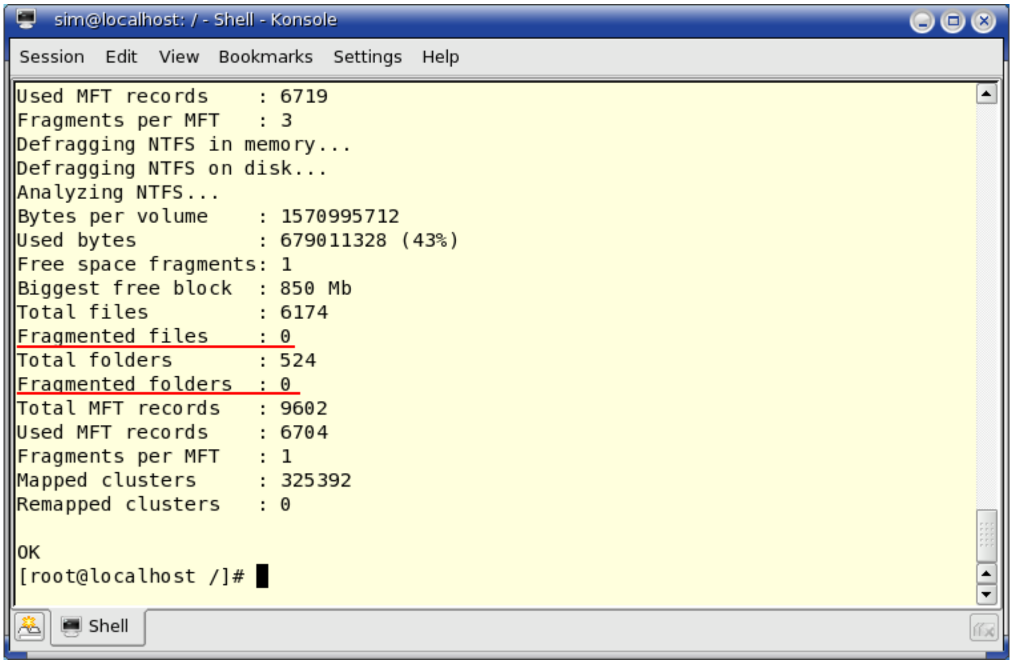
2. Sort file and directories according to their “file size” attribute in descending order.


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fdisk -l

Disk /dev/hda: 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/hda1   *           1         719     5775336   83  Linux
/dev/hda2             720        1305     4707045    5  Extended
/dev/hda5             720         859     1124518+  82  Linux swap / Solaris
/dev/hda6             860        1305     3582463+  83  Linux

Disk /dev/hdb: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hdb1   *           1         191     1534176    7  HPFS/NTFS
[root@localhost /]# dfntfs -d+ -s- /dev/hdb1
```


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
 44,      5, 104 "/WINDOWS/PCHEALTH/HELPCTR/DataColl/CollectedData_36.xml"
 45,      5, 63 "/WINDOWS/PCHEALTH/HELPCTR/DataColl/CollectedData_38.xml"
 46,      5, 32 "/WINDOWS/Fonts/"
 47,      5, 28 "/WINDOWS/inf/sti.PNF"
 48,      5, 16 "/WINDOWS/comsetup.log"
 49,      5, 8 "/WINDOWS/ntdtcsetup.log"
Bytes per volume     : 1570995712
Used bytes           : 679011328 (43%)
Free space fragments: 2787
Biggest free block  : 240 Mb
Total files          : 6174
Fragmented files     : 965
Total folders        : 524
Fragmented folders   : 21
Total MFT records    : 9602
Used MFT records     : 6719
Fragments per MFT    : 3
Defragging NTFS in memory...
Defragging NTFS on disk...
Analyzing NTFS...
52% /WINDOWS/system32/drivers/arp1394.sys
```


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
Used MFT records    : 6719
Fragments per MFT   : 3
Defragging NTFS in memory...
Defragging NTFS on disk...
Analyzing NTFS...
Bytes per volume    : 1570995712
Used bytes          : 679011328 (43%)
Free space fragments: 1
Biggest free block  : 850 Mb
Total files         : 6174
Fragmented files    : 0
Total folders       : 524
Fragmented folders  : 0
Total MFT records   : 9602
Used MFT records    : 6704
Fragments per MFT   : 1
Mapped clusters     : 325392
Remapped clusters   : 0

OK
[root@localhost /]#
```

9.1.5 WIPE Utility – Fill with zeros free space on a NTFS/FAT volume.

Name

wipe – zero free space (unused clusters and tails of files/directories) on NTFS/FAT volumes.

Synopsis

wipe [options] device

E.g.: wipe –c –t /dev/hdb1

Options

-c	Wipe unused clusters;
-t	Wipe tails of files/directories;
--help	Display this help;
--trace	Turn on UFSD trace;
--verbose	Explain what is being done;
--version	Show the version and exit.

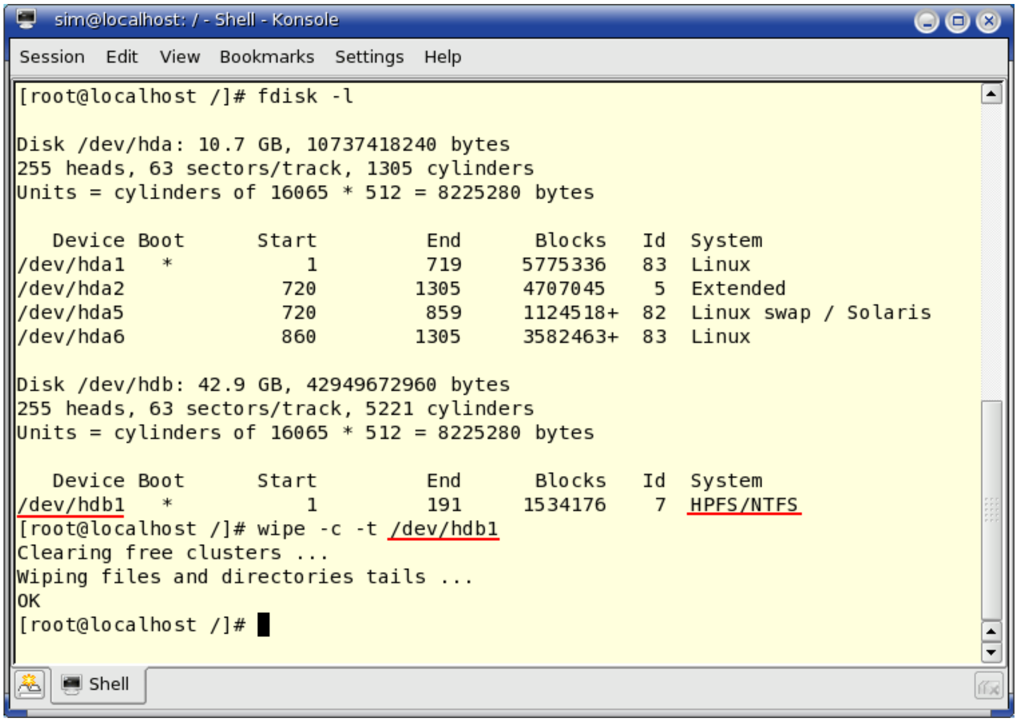
Description

Wipe Partition function allows irreversibly destroying all contents of a partition by overwriting all of its sectors with unused data (zeroes).

This function can be used, if a user intends:

- destroying on-partition data without an ability of restoration any of their parts;
- reselling or renting a workable hard disk;
- surely exclude any traces of old data on a newly formatted partition;
- destroying non-standard protection/registration/deactivation hidden marks made by some software.

Screenshots


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fdisk -l

Disk /dev/hda: 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/hda1   *           1         719     5775336   83  Linux
/dev/hda2             720        1305     4707045    5  Extended
/dev/hda5             720         859     1124518+  82  Linux swap / Solaris
/dev/hda6             860        1305     3582463+  83  Linux

Disk /dev/hdb: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hdb1   *           1         191     1534176    7  HPFS/NTFS
[root@localhost /]# wipe -c -t /dev/hdb1
Clearing free clusters ...
Wiping files and directories tails ...
OK
[root@localhost /]#
```

9.1.6 MFTPACK Utility – Pack/truncate MFT (Master File Table) on a NTFS volume.

Name

mftpack – pack MFT records and/or truncate MFT on NTFS volumes.

Synopsis

mftpack [options] device

E.g.: mftpack –c –t /dev/hdb1

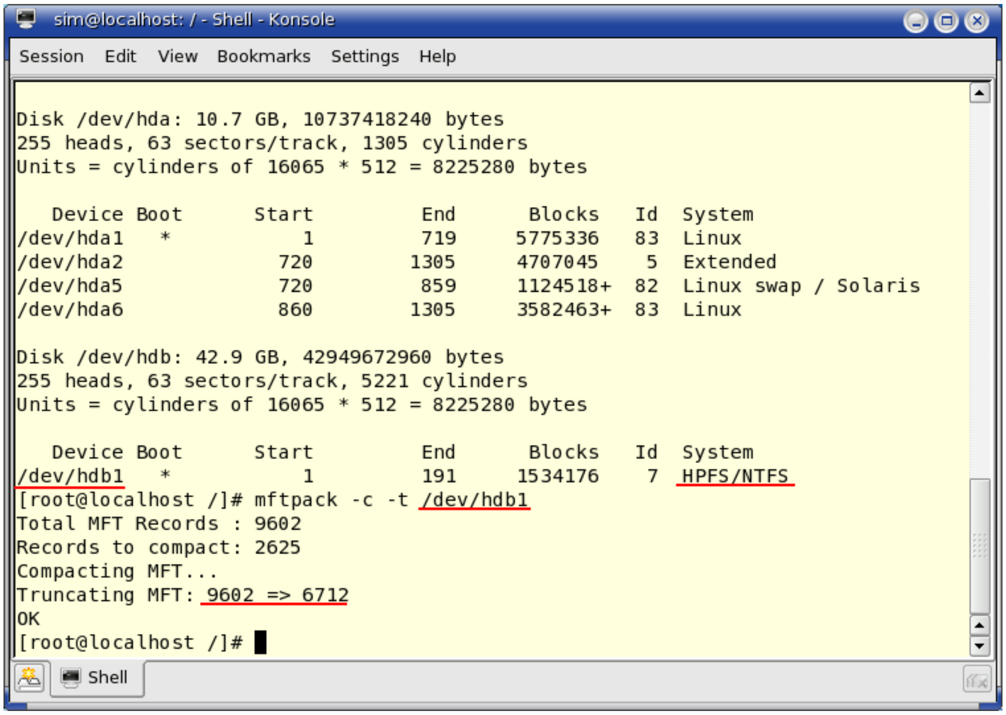
Options

-c	Compact MFT records (move tail records to the head of $MFT);
-t	Truncate MFT (remove unused tail records);
--help	Display this help;
--trace	Turn on UFSD trace;
--verbose	Explain what is being done;
--version	Show the version and exit.

Description

Master File Table (MFT) is a relational database that consists of rows of file records and columns of file attributes (size, time and date stamps, permissions, data contents and so forth). It contains at least one entry for every file on an NTFS volume, including the MFT itself. MFT is similar to a FAT table in a FAT file system. In the course of time the MFT file can also be fragmented, bulky and inefficiently take up too much disk space, thus slowing down the speed at which data is accessed. The mftpack utility provides with all necessary functionality to defragment MFT. Please note this utility may release considerable disk space that MFT inefficiently takes up.

Screenshots



```
Disk /dev/hda: 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/hda1   *           1         719     5775336   83  Linux
/dev/hda2             720        1305     4707045    5  Extended
/dev/hda5             720         859     1124518+  82  Linux swap / Solaris
/dev/hda6             860        1305     3582463+  83  Linux

Disk /dev/hdb: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hdb1   *           1         191     1534176    7  HPFS/NTFS
[root@localhost /]# mftpack -c -t /dev/hdb1
Total MFT Records : 9602
Records to compact: 2625
Compacting MFT...
Truncating MFT: 9602 => 6712
OK
[root@localhost /]#
```

9.1.7 HDLNK Utility – Enumerate all hard links on NTFS volume.

Name

hdlnk – enumerate all hard links on NTFS volumes and display to stdout (standard output).

Synopsis

hdlnk device [options]

E.g.: hdlnk /dev/hdb1 –o report.txt

Options

-o	A file name should be specified (where all hard links must be enumerated). Stdout is by default;
-v	Explain what is being done;
-h	Display this help;
--trace	Turn on UFSD trace;
--version	Show the version and exit.

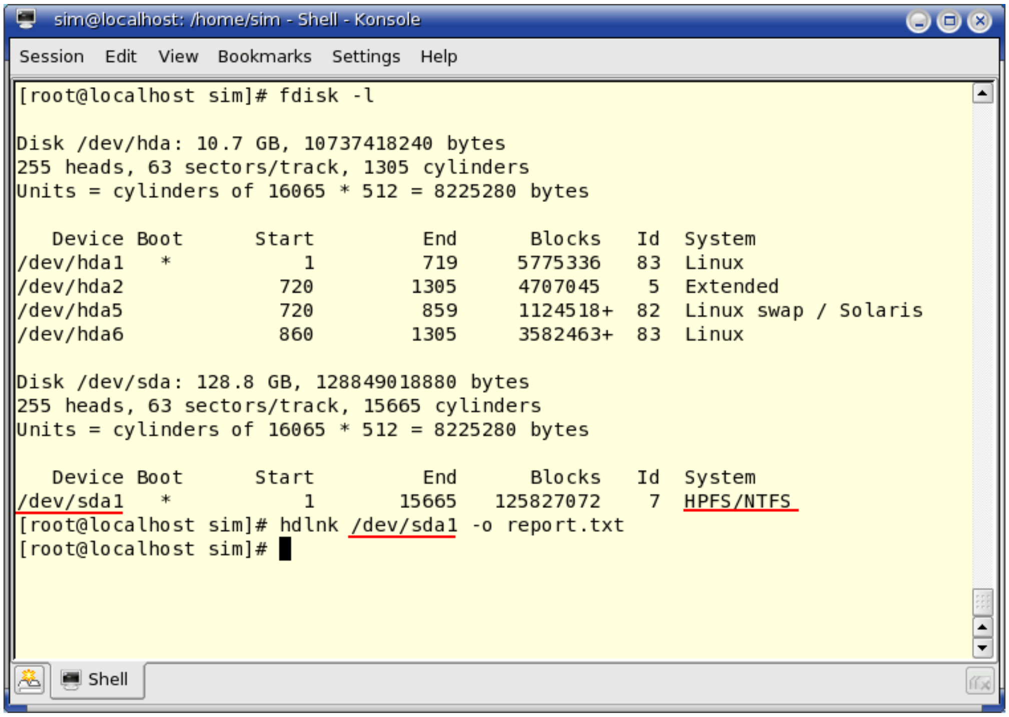
Description

A hard link is a directory entry for a file. Every file can be considered to have at least one hard link. On NTFS volumes, each file can have multiple hard links, and thus a single file can appear in many directories (or even in the same directory with different names). Because all of the links reference the same file, programs can open any of the links and modify the file. A file is deleted from the file system only after all links to it have been deleted. After you create a hard link, programs can use it like any other file name.

All actual data on disk that have more than one hard link will be enumerated using the hdlnk utility.

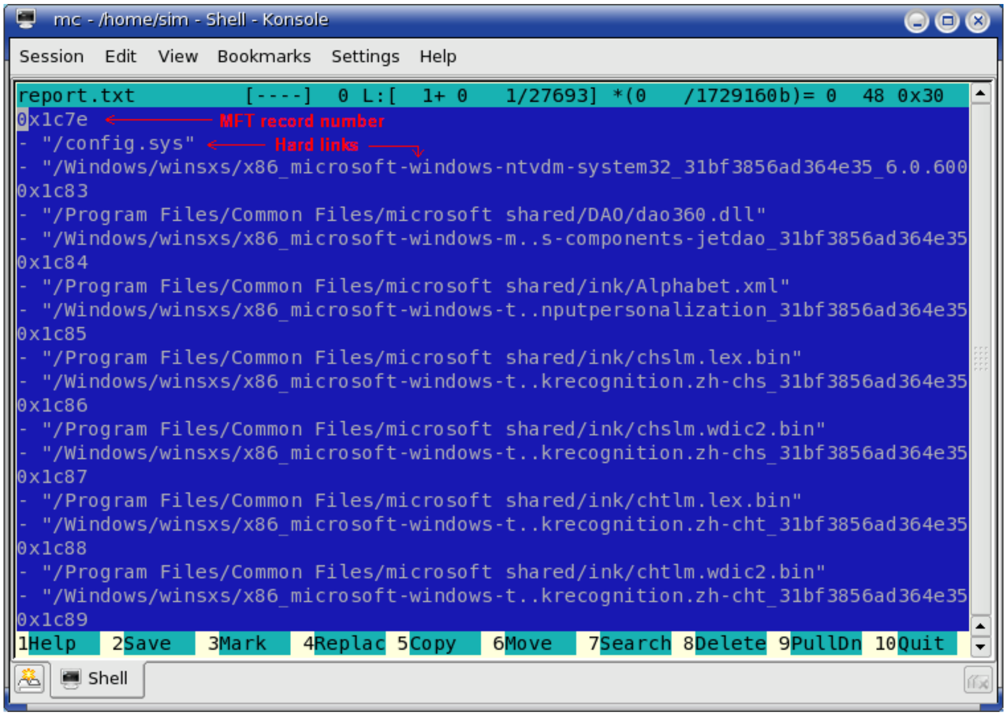
Screenshots

Let’s enumerate all hard links on a Vista NTFS partition. The list of hard links must be written to a report.txt file (the file doesn’t exist).



```
[root@localhost sim]# fdisk -l

Disk /dev/hda: 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/hda1   *           1         719     5775336   83  Linux
/dev/hda2             720        1305     4707045    5  Extended
/dev/hda5             720         859     1124518+  82  Linux swap / Solaris
/dev/hda6             860        1305     3582463+  83  Linux

Disk /dev/sda: 128.8 GB, 128849018880 bytes
255 heads, 63 sectors/track, 15665 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1       15665   125827072    7  HPFS/NTFS
[root@localhost sim]# hdlnk /dev/sda1 -o report.txt
[root@localhost sim]#
```

The report.txt file:



```
report.txt           [----]  0 L:[  1+ 0   1/27693] *(0   /1729160b)= 0   48 0x30
0x1c7e
- "/config.sys"
- "/Windows/winsxs/x86_microsoft-windows-ntvdm-system32_31bf3856ad364e35_6.0.600
0x1c83
- "/Program Files/Common Files/microsoft shared/DAO/dao360.dll"
- "/Windows/winsxs/x86_microsoft-windows-m..s-components-jetdao_31bf3856ad364e35
0x1c84
- "/Program Files/Common Files/microsoft shared/ink/Alphabet.xml"
- "/Windows/winsxs/x86_microsoft-windows-t..nputpersonalization_31bf3856ad364e35
0x1c85
- "/Program Files/Common Files/microsoft shared/ink/chslm.lex.bin"
- "/Windows/winsxs/x86_microsoft-windows-t..krecognition.zh-chs_31bf3856ad364e35
0x1c86
- "/Program Files/Common Files/microsoft shared/ink/chslm.wdic2.bin"
- "/Windows/winsxs/x86_microsoft-windows-t..krecognition.zh-chs_31bf3856ad364e35
0x1c87
- "/Program Files/Common Files/microsoft shared/ink/chtlm.lex.bin"
- "/Windows/winsxs/x86_microsoft-windows-t..krecognition.zh-cht_31bf3856ad364e35
0x1c88
- "/Program Files/Common Files/microsoft shared/ink/chtlm.wdic2.bin"
- "/Windows/winsxs/x86_microsoft-windows-t..krecognition.zh-cht_31bf3856ad364e35
0x1c89
1Help   2Save   3Mark   4Replac 5Copy   6Move   7Search 8Delete 9PullDn 10Quit
```

9.1.8 JUNCTION Utility – Reparse point viewer on a NTFS volume.

Name

junction – display reparse point information.

Synopsis

junction device [options]

E.g.: junction /dev/hdb1 –o report.txt

Options

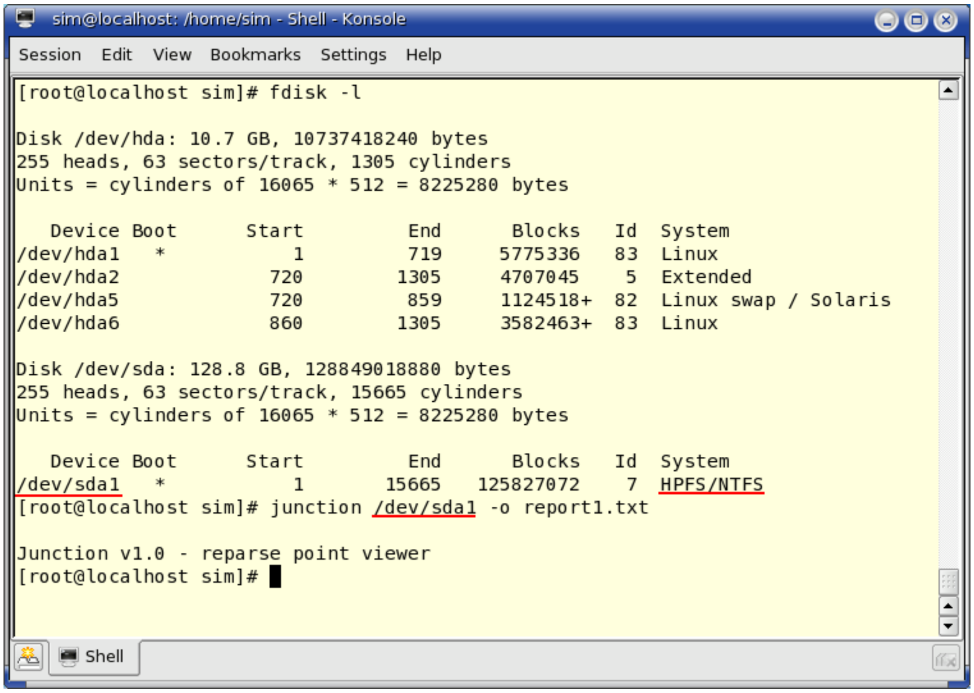
-o	A file name should be specified (where all reparse points must be enumerated). Stdout is by default;
-v	Explain what is being done;
--help	Display this help;
--trace	Turn on UFSD trace;
--version	Show the version and exit.

Description

Windows 2000 and higher supports junctions - directory symbolic links, where a directory used as a symbolic link to another directory on the computer. For example, if the directory “D:\symlink” specifies “C:\winnt\system32” as its target, then when an application accesses “D:\symlink\drivers”, it actually accesses “C:\winnt\system32\drivers”. Linux doesn’t have any tools to manage junctions and we therefore decided to write this junction utility. It allows you to see if files or directories are actually reparse points. Reparse points are the mechanism on which NTFS junctions are based, and they are used by Windows’ Remote Storage Service (RSS), as well as volume mount points.

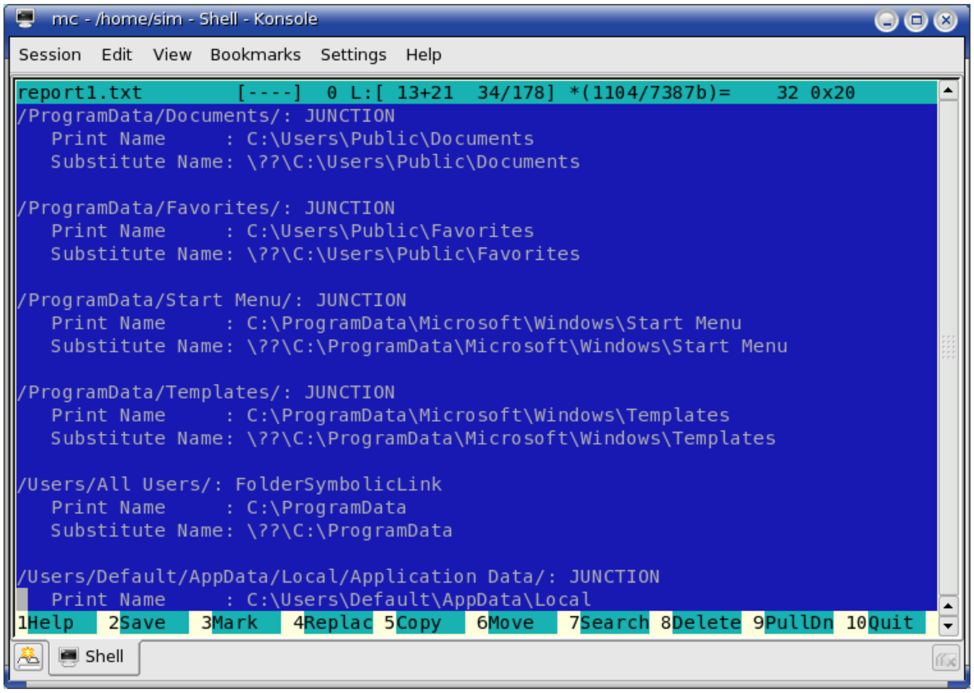
Screenshots

Let’s enumerate all reparse points on a Vista NTFS partition. The list of reparse points must be written to a report1.txt file (the file doesn’t exist).


sim@localhost: /home/sim - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost sim]# fdisk -l

Disk /dev/hda: 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/hda1   *           1         719     5775336   83  Linux
/dev/hda2             720        1305     4707045    5  Extended
/dev/hda5             720         859     1124518+  82  Linux swap / Solaris
/dev/hda6             860        1305     3582463+  83  Linux

Disk /dev/sda: 128.8 GB, 128849018880 bytes
255 heads, 63 sectors/track, 15665 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1       15665   125827072    7  HPFS/NTFS
[root@localhost sim]# junction /dev/sda1 -o report1.txt

Junction v1.0 - reparse point viewer
[root@localhost sim]#
```

report1.txt file:


mc - /home/sim - Shell - Konsole
Session Edit View Bookmarks Settings Help
1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
Shell

```
report1.txt         [----]  0 L:[ 13+21  34/178] *(1104/7387b)=    32 0x20
/ProgramData/Documents/: JUNCTION
   Print Name      : C:\Users\Public\Documents
   Substitute Name: \??\C:\Users\Public\Documents

/ProgramData/Favorites/: JUNCTION
   Print Name      : C:\Users\Public\Favorites
   Substitute Name: \??\C:\Users\Public\Favorites

/ProgramData/Start Menu/: JUNCTION
   Print Name      : C:\ProgramData\Microsoft\Windows\Start Menu
   Substitute Name: \??\C:\ProgramData\Microsoft\Windows\Start Menu

/ProgramData/Templates/: JUNCTION
   Print Name      : C:\ProgramData\Microsoft\Windows\Templates
   Substitute Name: \??\C:\ProgramData\Microsoft\Windows\Templates

/Users/All Users/: FolderSymbolicLink
   Print Name      : C:\ProgramData
   Substitute Name: \??\C:\ProgramData

/Users/Default/AppData/Local/Application Data/: JUNCTION
   Print Name      : C:\Users\Default\AppData\Local
```

9.1.9 FSUTIL Utility – Powerful Utility to Perform NTFS File System Related Tasks

Name

fsutil – NTFS file system utility for advanced users (Windows XP fsutil analogue).

Description

Fsutil is a Linux utility that you can use to perform many NTFS file system related tasks, such as managing file system information, compression, hardlinks and etc. Because **fsutil** is quite powerful, it should only be used by advanced users who have a thorough knowledge of NTFS file system.

Note: To view help for the available subcommands, type **fsutil**, type the subcommand, and then type **help** (that is, **fsutil** *subcommand* **help**).

Synopsis

fsutil <subcommand>

Subcommands

behavior	Control file system behavior.
dirty	Manage volume dirty bit.
file	File specific commands.
fsinfo	File system information.
hardlink	Hardlink namagement.
objectid	Object ID management.
compress	Manage compression.
s**treams**	Streams management.
sparse	Sparse file control.

Fsutil: behavior

Controls file system behavior. Queries, changes, enables, or disables the settings for generating 8.3 character-length file names and the amount of disk space reserved of the MFT Zone. Queries how many bytes of RAM NTFS for Linux library (UFSD) uses.

Syntax

fsutil behavior query <*volume*> <*option*> - Query the file system behavior parameters.

E.g.: fsutil behavior query /mnt/vol1/ memoryusage

fsutil behavior set <*volume*> <*option*> <*value*> - Change the file system behavior parameters.

E.g.: fsutil behavior set /mnt/vol1 disable8dot3 1

Options

disable8dot3 {**1**|**0**}

Disables creation of 8.3 character-length file names on NTFS-formatted volumes.

mftzone *value*

The master file table (MFT) Zone is a reserved area that enables the MFT to expand as needed, in order to prevent MFT fragmentation. Set the *value* from 1 (default) to 4 (maximum). The *value* is in 8ths of the disk.

memoryusage

Shows memory usage of NTFS for Linux library (UFSD) in bytes.

TotalBytes – total amount of bytes.

BytesPerDir – how many bytes the library uses for opened directories.

BytesPerFile - how many bytes the library uses for opened files.

Remarks

- Using **disable8dot3** {**1**|**0**}

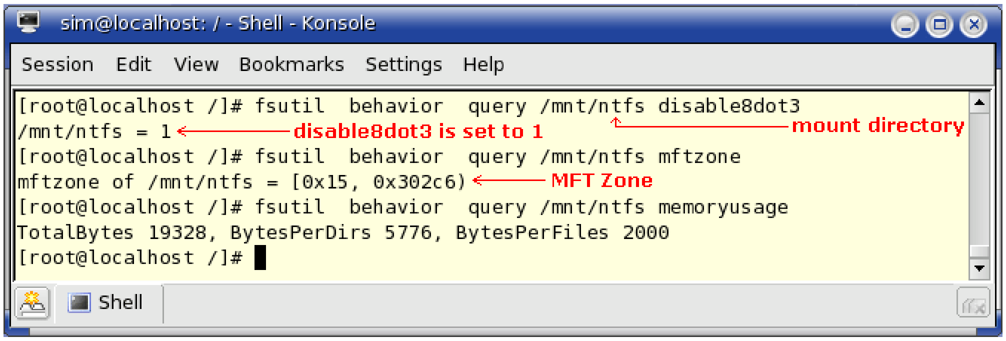
  When **disable8dot3** is set to **0**, every time you create a file with a long file name, NTFS creates a second file entry that has a 8.3 character-length file name. When NTFS creates files in a folder, it must find the 8.3 character-length file names associated with the long file names.

- Using **mftzone** *value*

  The *value* is an approximation of the initial size of the MFT plus the MFT Zone for a new volume. It is set when mounting for each file system. As space on the volume is used, NTFS adjusts the space reserved for future MFT growth. If the MFT Zone is already large, the full MFT Zone size is not reserved again. MFT Zone shrinks as the space is used.

  The file system does not redetermine the MFT Zone location until the current MFT Zone is completely used.

Screenshots


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil  behavior  query /mnt/ntfs disable8dot3
/mnt/ntfs = 1
[root@localhost /]# fsutil  behavior  query /mnt/ntfs mftzone
mftzone of /mnt/ntfs = [0x15, 0x302c6)
[root@localhost /]# fsutil  behavior  query /mnt/ntfs memoryusage
TotalBytes 19328, BytesPerDirs 5776, BytesPerFiles 2000
[root@localhost /]#
```

disable8dot3 is set to 1
mount directory
MFT Zone

Fsutil: dirty

Queries to see whether a volume’s dirty bit is set. Sets a volume's dirty bit. When a volume’s dirty bit is set, **autochk** (for Windows OS only) automatically checks the volume for errors the next time the computer is restarted.

Syntax

fsutil dirty query <*volume or device*> - Query the dirty bit.

E.g.: fsutil dirty query /mnt/vol1

fsutil dirty set <*volume or device*> - Set the dirty bit.

E.g.: fsutil dirty set /mnt/vol1

fsutil dirty clear <*volume or device*> <*option*> <*value*> - Clear the dirty bit.

E.g.: fsutil dirty clear /mnt/vol1

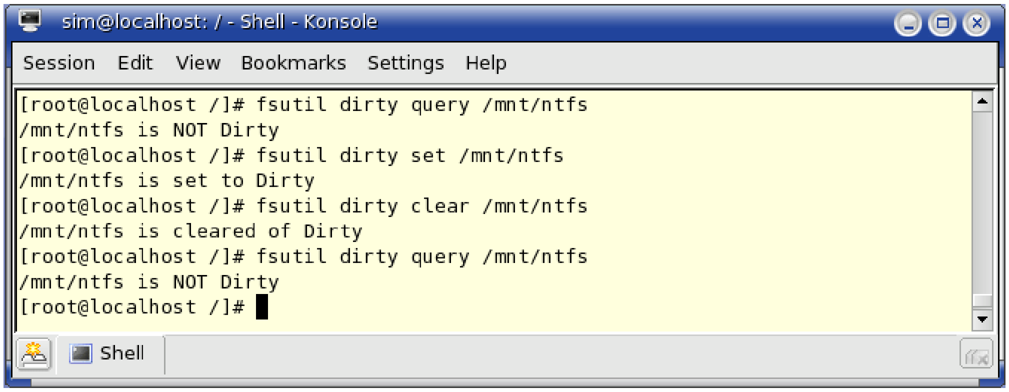
<volume or device>

You can specify the volume (mount point) in case the partition is mounted or you can specify the device name (/dev/hda1) in case the partition in not mounted.

Remarks

- If a volume’s dirty bit is set, this indicates that the file system may be in an inconsistent state. The dirty bit can be set because the volume is online and has outstanding changes, because changes were made to the volume and the computer shutdown before the changes were committed to disk, or because corruption was detected on the volume. If the dirty bit is set when the computer restarts, **chkdsk** (Windows utility) runs to verify the consistency of the volume.

Screenshots



sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil dirty query /mnt/ntfs
/mnt/ntfs is NOT Dirty
[root@localhost /]# fsutil dirty set /mnt/ntfs
/mnt/ntfs is set to Dirty
[root@localhost /]# fsutil dirty clear /mnt/ntfs
/mnt/ntfs is cleared of Dirty
[root@localhost /]# fsutil dirty query /mnt/ntfs
/mnt/ntfs is NOT Dirty
[root@localhost /]#
```

Fsutil: file

Typically used by support professionals. Queries allocated ranges for a file, sets a file's short name, sets a file's valid data length, sets zero data for a file and etc.

Syntax

fsutil file <queryallocranges> <*filename*> - Query the allocated ranges for a file.

E.g.: fsutil file queryallocranges /mnt/vol1/hello.txt

fsutil file <setshortname> <*filename*> <*shortname*> - Set the short name for a file.

E.g.: fsutil file setshortname /mnt/vol1/hello.txt short.txt

fsutil file <getsizes> <*filename*> - Get the sizes for a file.

E.g.: fsutil file getsizes /mnt/vol1/hello.txt

fsutil file <setvaliddata> <*filename*> <*datalength*> - Set the valid data length for a file.

E.g.: fsutil file setvaliddata /mnt/vol1/hello.txt 4096

fsutil file <setzerodata> offset=<*offset*> **length**=<*length*> <*filename*> - Set the zero data for a file.

E.g.: fsutil file setzerodata offset=100 length=150 /mnt/vol1/hello.txt

fsutil file <dumprecord> <*filename*> - Dumps raw file/directories record.

E.g.: fsutil file dumprecord /mnt/vol1/hello.txt

fsutil file <dumprecordnum> <*volume*> <*record_num*>- Dump raw record by its number.

E.g.: fsutil file dumprecordnum /mnt/vol1/ 1234

E.g.: fsutil file dumprecordnum /mnt/vol1/ 0x1234

Options

queryallocranges

Queries the allocated ranges for a file on an NTFS volume. Useful for determining whether a file has sparse regions.

setshortname

Sets the short name (8.3 character-length file name) for a file on a NTFS volume.

shortname

Specifies the file's shortname.

getsizes

Shows three Windows sizes: Allocated, Data, Valid.

setvaliddata

Sets the valid data length for a file on an NTFS volume.

datalength

Specifies the length of the file in bytes.

setzerodata

Sets a range (specified by *offset* and *length*) of the file to zeroes, which empties the file. If the file is a sparse file, the underlying allocation units are decommitted.

offset=*offset*

Specifies the file offset, the start of the range to set to zeroes.

length=*length*

Specifies the length of the range to set to zero.

dumprecord

Shows all MFT records for the specified file.

dumprecordnum

Shows the specified MFT record for the specified file.

record_num

Specified the number of MFT record to show.

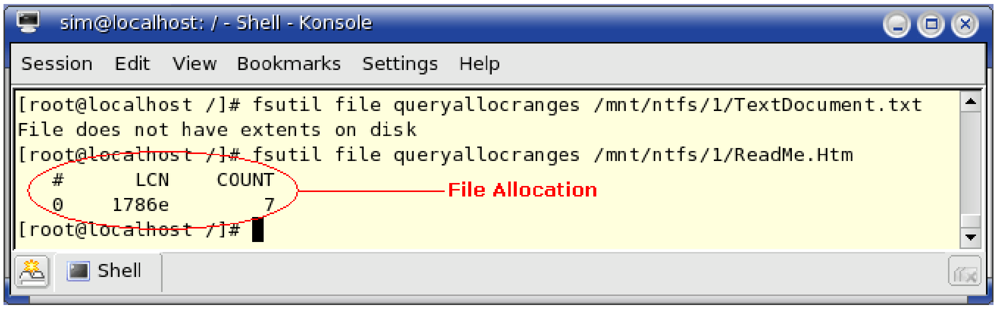
Remarks

- Using **setvaliddata**

There are two important concepts of file length in NTFS: the End of File (EOF) marker and the Valid Data Length (VDL). The EOF indicates the actual length of the file. The VDL identifies the length of valid data on disk. Any reads between VDL and EOF automatically return 0.

Screenshots

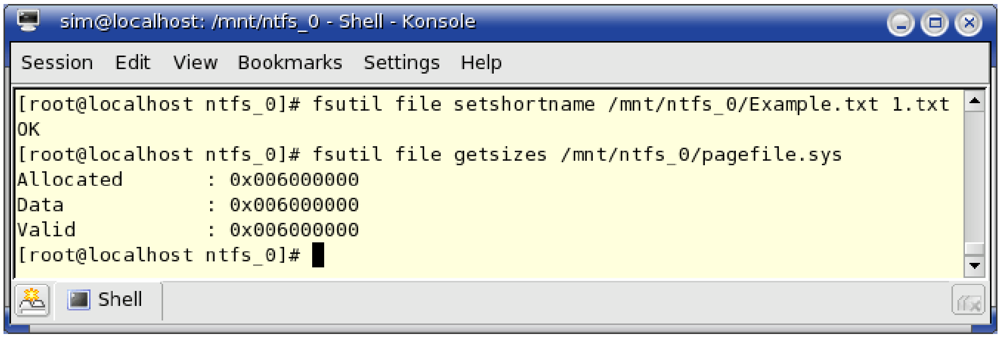
1. The **queryallocranges** option.


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell
File Allocation

```
[root@localhost /]# fsutil file queryallocranges /mnt/ntfs/1/TextDocument.txt
File does not have extents on disk
[root@localhost /]# fsutil file queryallocranges /mnt/ntfs/1/ReadMe.Htm
   #       LCN     COUNT
   0     1786e         7
[root@localhost /]#
```

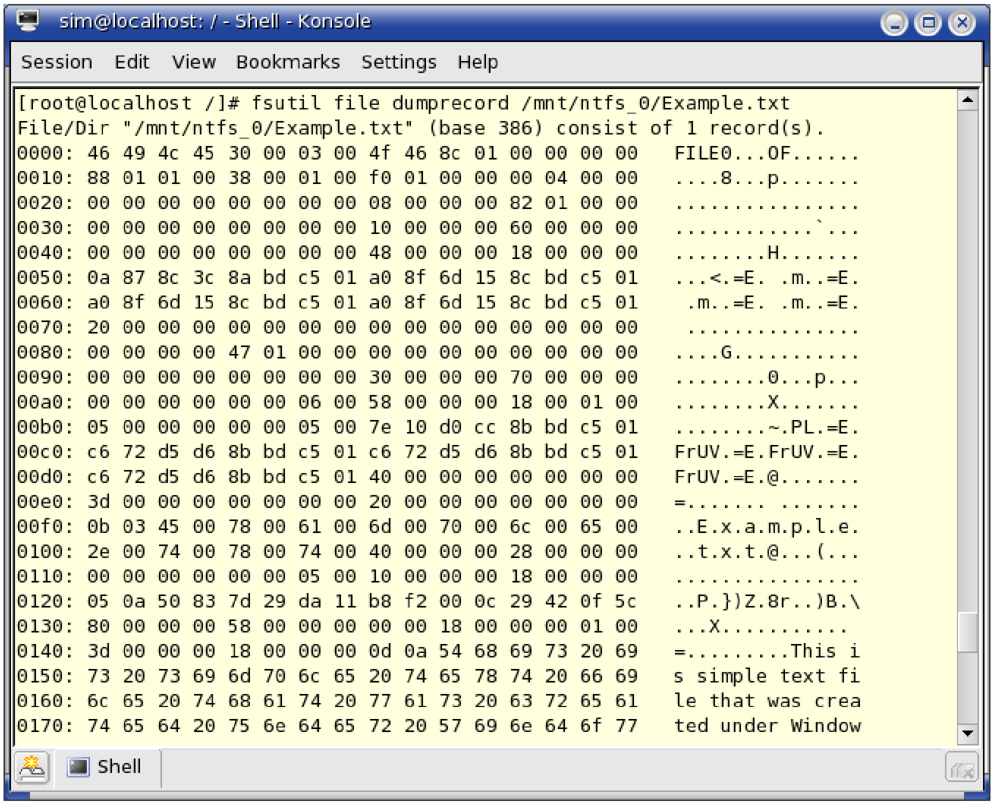
The /mnt/1/TextDocument.txt file lies in the MFT Zone that is why the file doesn't have extents on the disk. **LCN** – Logical Cluster Number.

2. **Setshortname** and **getsizes** options.


sim@localhost: /mnt/ntfs_0 - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost ntfs_0]# fsutil file setshortname /mnt/ntfs_0/Example.txt 1.txt
OK
[root@localhost ntfs_0]# fsutil file getsizes /mnt/ntfs_0/pagefile.sys
Allocated       : 0x006000000
Data            : 0x006000000
Valid           : 0x006000000
[root@localhost ntfs_0]#
```

3. The **dumprecord** option:



sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil file dumprecord /mnt/ntfs_0/Example.txt
File/Dir "/mnt/ntfs_0/Example.txt" (base 386) consist of 1 record(s).
0000: 46 49 4c 45 30 00 03 00 4f 46 8c 01 00 00 00 00   FILE0...OF......
0010: 88 01 01 00 38 00 01 00 f0 01 00 00 00 04 00 00   ....8...p.......
0020: 00 00 00 00 00 00 00 00 08 00 00 00 82 01 00 00   ................
0030: 00 00 00 00 00 00 00 00 10 00 00 00 60 00 00 00   ............`...
0040: 00 00 00 00 00 00 00 00 48 00 00 00 18 00 00 00   ........H.......
0050: 0a 87 8c 3c 8a bd c5 01 a0 8f 6d 15 8c bd c5 01   ...<.=E. .m..=E.
0060: a0 8f 6d 15 8c bd c5 01 a0 8f 6d 15 8c bd c5 01    .m..=E. .m..=E.
0070: 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ...............
0080: 00 00 00 00 47 01 00 00 00 00 00 00 00 00 00 00   ....G...........
0090: 00 00 00 00 00 00 00 00 30 00 00 00 70 00 00 00   ........0...p...
00a0: 00 00 00 00 00 00 06 00 58 00 00 00 18 00 01 00   ........X.......
00b0: 05 00 00 00 00 00 05 00 7e 10 d0 cc 8b bd c5 01   ........~.PL.=E.
00c0: c6 72 d5 d6 8b bd c5 01 c6 72 d5 d6 8b bd c5 01   FrUV.=E.FrUV.=E.
00d0: c6 72 d5 d6 8b bd c5 01 40 00 00 00 00 00 00 00   FrUV.=E.@.......
00e0: 3d 00 00 00 00 00 00 00 20 00 00 00 00 00 00 00   =....... .......
00f0: 0b 03 45 00 78 00 61 00 6d 00 70 00 6c 00 65 00   ..E.x.a.m.p.l.e.
0100: 2e 00 74 00 78 00 74 00 40 00 00 00 28 00 00 00   ..t.x.t.@...(...
0110: 00 00 00 00 00 00 05 00 10 00 00 00 18 00 00 00   ................
0120: 05 0a 50 83 7d 29 da 11 b8 f2 00 0c 29 42 0f 5c   ..P.})Z.8r..)B.\
0130: 80 00 00 00 58 00 00 00 00 00 18 00 00 00 01 00   ...X...........
0140: 3d 00 00 00 18 00 00 00 0d 0a 54 68 69 73 20 69   =.........This i
0150: 73 20 73 69 6d 70 6c 65 20 74 65 78 74 20 66 69   s simple text fi
0160: 6c 65 20 74 68 61 74 20 77 61 73 20 63 72 65 61   le that was crea
0170: 74 65 64 20 75 6e 64 65 72 20 57 69 6e 64 6f 77   ted under Window
```

Fsutil: fsinfo

Typically used by support professionals. Queries the drive type, queries volume information, queries NTFS-specific volume information, or queries file system statistics.

Syntax

fsutil fsinfo <volumeinfo> <*volume pathname*> – Query volume information.

E.g.: fsutil fsinfo volumeinfo /mnt/vol1

fsutil fsinfo <ntfsinfo> <*volume pathname*> – Query NTFS specific volume information.

E.g.: fsutil fsinfo ntfsinfo /mnt/vol1

fsutil fsinfo <**statistics>** <*volume pathname*> - Query file system statistics.

E.g.: fsutil fsinfo statistics /mnt/vol1

Options

volumeinfo

Lists information for the specified volume, such as the file system, and whether the volume supports case-sensitive file names, unicode in file names, or disk quotas.

ntfsinfo

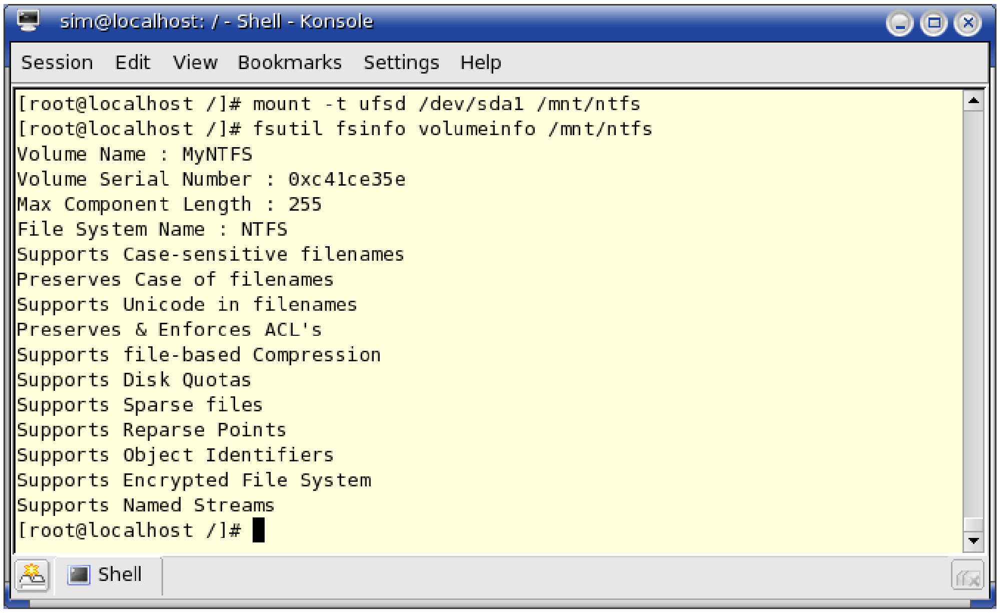
Lists NTFS specific volume information for the specified volume, such as the number of sectors, total clusters, free clusters, and the start and end of the MFT Zone.

statistics

Lists file system statistics for the specified volume, such as metadata, log file, and MFT reads and writes.

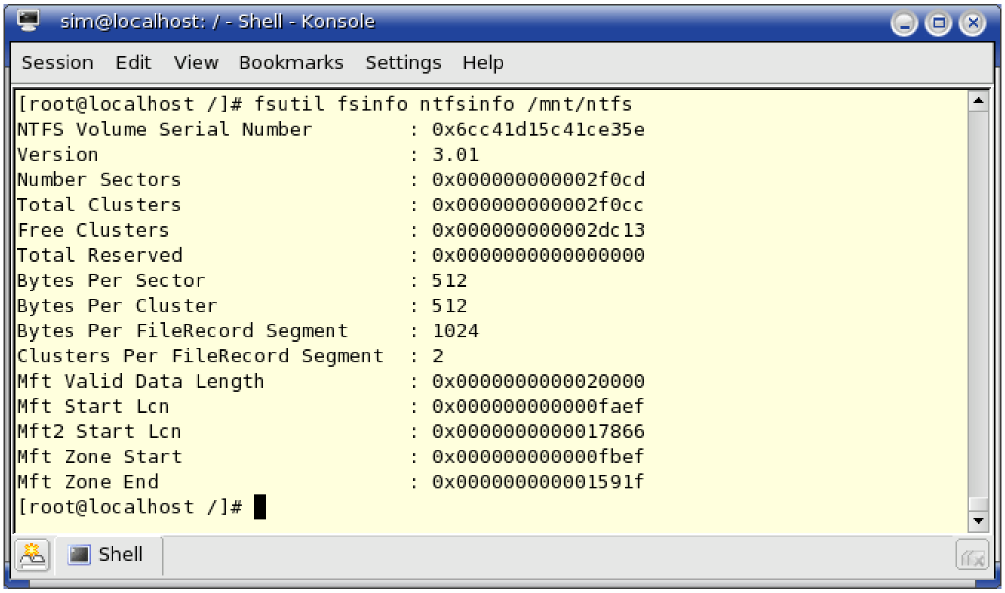
Screenshots

1. The **voluminfo** option.


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# mount -t ufsd /dev/sda1 /mnt/ntfs
[root@localhost /]# fsutil fsinfo volumeinfo /mnt/ntfs
Volume Name : MyNTFS
Volume Serial Number : 0xc41ce35e
Max Component Length : 255
File System Name : NTFS
Supports Case-sensitive filenames
Preserves Case of filenames
Supports Unicode in filenames
Preserves & Enforces ACL's
Supports file-based Compression
Supports Disk Quotas
Supports Sparse files
Supports Reparse Points
Supports Object Identifiers
Supports Encrypted File System
Supports Named Streams
[root@localhost /]#
```

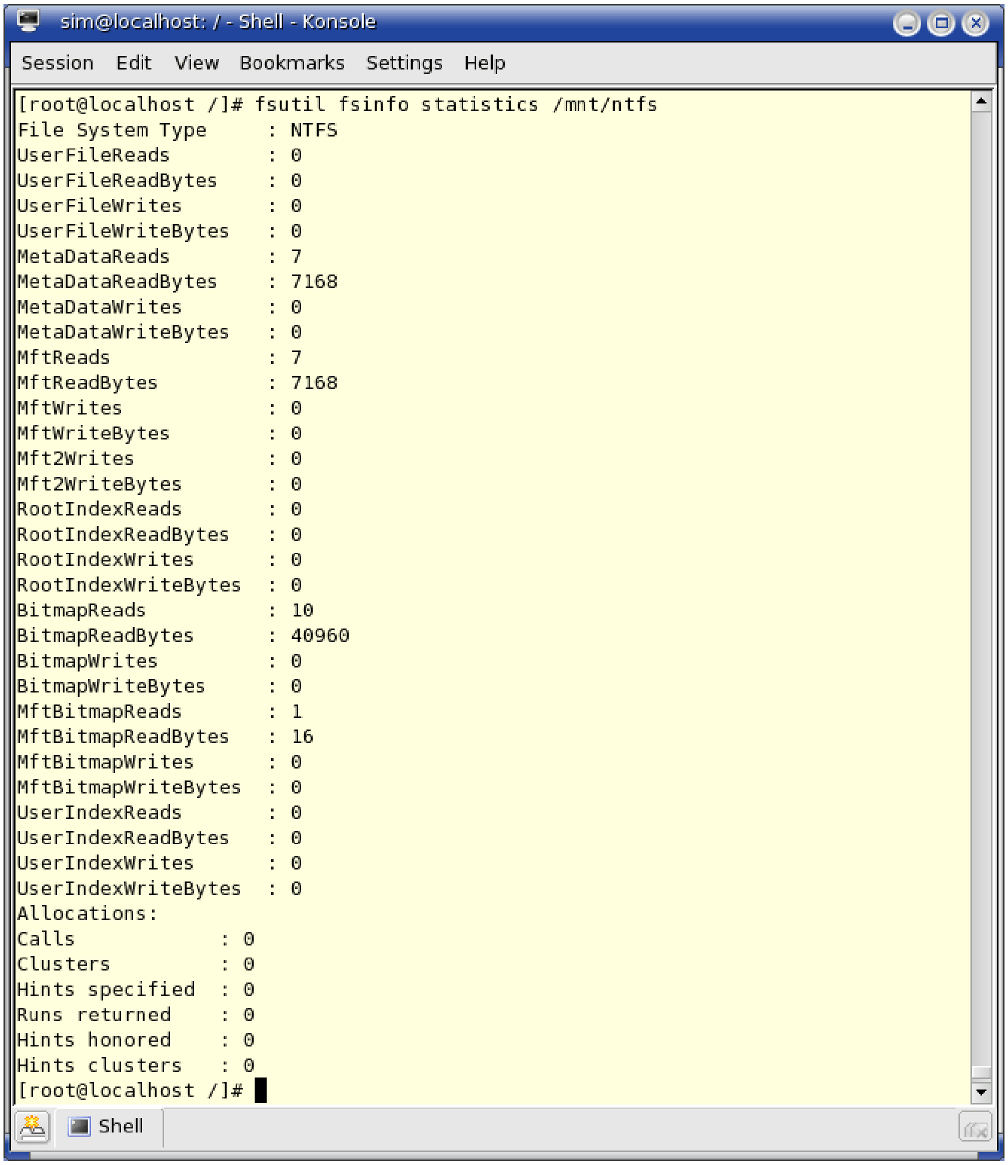
2. The **ntfsinfo** option.



sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil fsinfo ntfsinfo /mnt/ntfs
NTFS Volume Serial Number        : 0x6cc41d15c41ce35e
Version                          : 3.01
Number Sectors                   : 0x000000000002f0cd
Total Clusters                   : 0x000000000002f0cc
Free Clusters                    : 0x000000000002dc13
Total Reserved                   : 0x0000000000000000
Bytes Per Sector                 : 512
Bytes Per Cluster                : 512
Bytes Per FileRecord Segment     : 1024
Clusters Per FileRecord Segment  : 2
Mft Valid Data Length            : 0x0000000000020000
Mft Start Lcn                    : 0x000000000000faef
Mft2 Start Lcn                   : 0x0000000000017866
Mft Zone Start                   : 0x000000000000fbef
Mft Zone End                     : 0x000000000001591f
[root@localhost /]#
```

3. The **statistics** option.



sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil fsinfo statistics /mnt/ntfs
File System Type      : NTFS
UserFileReads         : 0
UserFileReadBytes     : 0
UserFileWrites        : 0
UserFileWriteBytes    : 0
MetaDataReads         : 7
MetaDataReadBytes     : 7168
MetaDataWrites        : 0
MetaDataWriteBytes    : 0
MftReads              : 7
MftReadBytes          : 7168
MftWrites             : 0
MftWriteBytes         : 0
Mft2Writes            : 0
Mft2WriteBytes        : 0
RootIndexReads        : 0
RootIndexReadBytes    : 0
RootIndexWrites       : 0
RootIndexWriteBytes   : 0
BitmapReads           : 10
BitmapReadBytes       : 40960
BitmapWrites          : 0
BitmapWriteBytes      : 0
MftBitmapReads        : 1
MftBitmapReadBytes    : 16
MftBitmapWrites       : 0
MftBitmapWriteBytes   : 0
UserIndexReads        : 0
UserIndexReadBytes    : 0
UserIndexWrites       : 0
UserIndexWriteBytes   : 0
Allocations:
Calls             : 0
Clusters          : 0
Hints specified   : 0
Runs returned     : 0
Hints honored     : 0
Hints clusters    : 0
[root@localhost /]#
```

Fsutil: hardlink

A hard link is a directory entry for a file. Every file can be considered to have at least one hard link. On NTFS volumes, each file can have multiple hard links, and thus a single file can appear in many directories (or even in the same directory with different names). Because all of the links reference the same file, programs can open any of the links and modify the file. A file is deleted from the file system only after all links to it have been deleted. After you create a hard link, programs can use it like any other file name.

Syntax

fsutil hardlink <create> *<new_filename>* *<existing_filename>* – Create a hardlink.

E.g.: fsutil hardlink create /mnt/vol1/hi.txt /hello.txt

Options

create

Establishes an NTFS hard link between an existing file and a new file. An NTFS hard link is similar to a POSIX hard link.

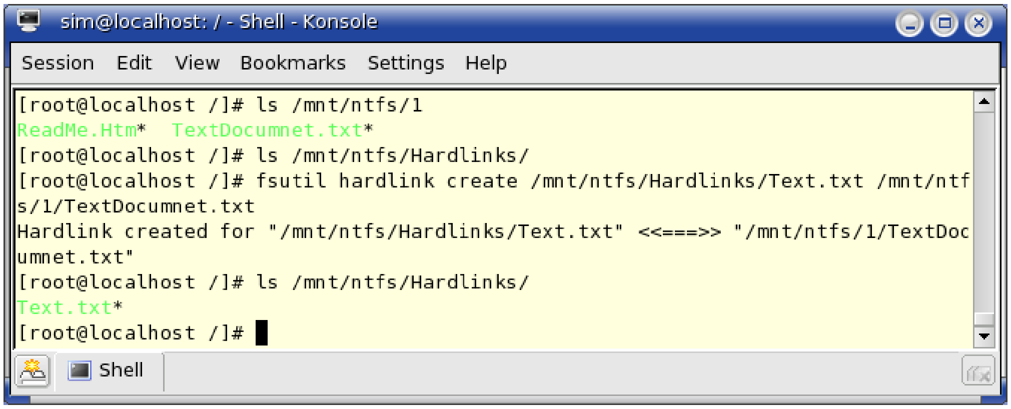
new_filename

Specifies the file to which you want to create a hardlink.

existing_filename

Specifies the file from which you want to create a hardlink.

Screenshots


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# ls /mnt/ntfs/1
ReadMe.Htm*  TextDocumnet.txt*
[root@localhost /]# ls /mnt/ntfs/Hardlinks/
[root@localhost /]# fsutil hardlink create /mnt/ntfs/Hardlinks/Text.txt /mnt/ntf
s/1/TextDocumnet.txt
Hardlink created for "/mnt/ntfs/Hardlinks/Text.txt" <<===>> "/mnt/ntfs/1/TextDoc
umnet.txt"
[root@localhost /]# ls /mnt/ntfs/Hardlinks/
Text.txt*
[root@localhost /]#
```

Fsutil: objectid

Typically used by professionals. Manages object identifiers, which are internal objects used by the Distributed Link Tracking (DLT) Client service and File Replication Service (FRS) to track other objects such as files, directories, and links. Object identifiers are invisible to most programs and should never be modified.

Syntax

fsutil objectid <query> - Query the object identifier.

E.g.: fsutil objectid query /mnt/vol1/hello.txt

fsutil objectid <set> *<ObjectId>* *<BirthVolumeId>* *<BirthObjectId>* *<DomainId>* *<filename>* - Change the object identifier.

```
E.g.: fsutil objected set 7adcc02fc9b4d4118f120090273fa9fc
                          dc6ad6      865fe8d21183913008c409d19e
                          d2dff02fc9b4d4118f120090273fa9d2
                           00000000000000000000000000000000 /mnt/vol1/hello.txt
```

fsutil objectid <delete> *<filename>* - Delete the object identifier.

E.g.: fsutil objected delete /mnt/vol1/hello.txt

fsutil objectid <create> - Create the object identifier.

E.g.: fsutil objected create /mnt/vol1/hello.txt

Options

query

Queries the object identifier.

set

Changes the object identifier.

delete

Deletes the object identifier.

create

Creates the object identifier if the file does not already have one, otherwise equivalent to query.

ObjectID

A file-specific 16 byte hexadecimal identifier that is guaranteed to be unique within a volume. It is used by the Distributed Link Tracking (DLT) Client service and the File Replication Service (FRS) to identify files. Any file that has an *ObjectID*, also has a *BirthVolumeID*, a *BirthObjectID*, and a *DomainID*. When you move a file, the *ObjectID* may change, but *BirthVolumeID* and *BirthObjectID* remain the same, which enables Windows to always find a file, no matter where it has been moved.

BirthVolumeID

A 16 byte hexadecimal identifier indicates the volume on which the file was located when it first obtained an *ObjectID*. This value is used by DLT Client service.

BirthObjectID

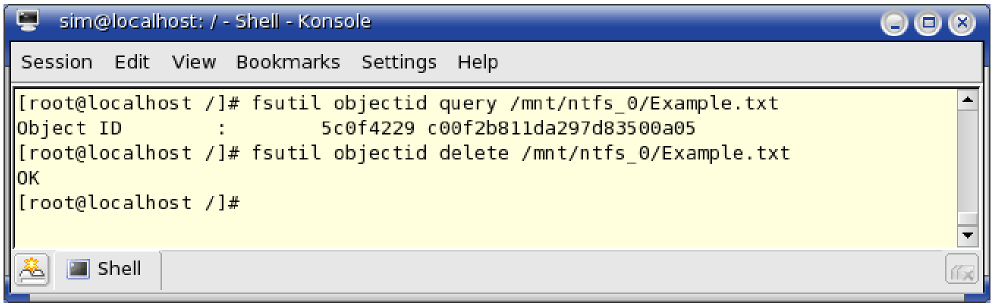
A 16 byte hexadecimal identifier indicates the file's original *ObjectID*. This value is used by DLT Client service.

DomainID

16 byte hexadecimal domain identifier. This value is not currently used and must be set to all zeros.

Note: All values must be in Hex of the form 8a0cf02fc9b4d4118f120090273fa91a.

Screenshots


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil objectid query /mnt/ntfs_0/Example.txt
Object ID         :         5c0f4229 c00f2b811da297d83500a05
[root@localhost /]# fsutil objectid delete /mnt/ntfs_0/Example.txt
OK
[root@localhost /]#
```

Fsutil: compress

Compressing files decreases their size and reduces the amount of space they use on your drives or removable storage media.

Syntax

fsutil compress queryflag <*filename*> – Query compression flag.

E.g.: fsutil compress queryflag /mnt/vol1/hello.txt

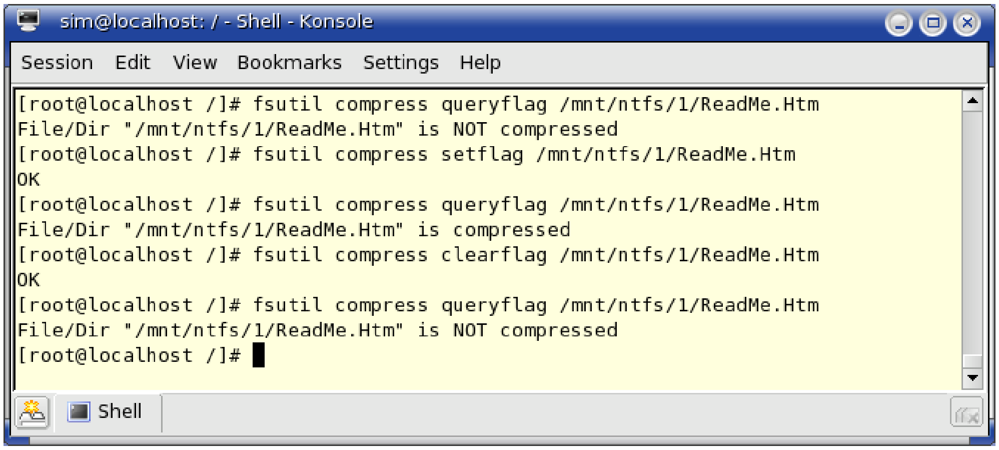
fsutil compress setflag <*filename*> (<-r>) – Set compression flag.

E.g.: fsutil compress setflag /mnt/vol1/hello.txt –r

fsutil compress clearflag <*filename*> (<-r>) – Clear compression flag.

E.g.: fsutil compress clearflag /mnt/vol1/hello.txt –r

Screenshots


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil compress queryflag /mnt/ntfs/1/ReadMe.Htm
File/Dir "/mnt/ntfs/1/ReadMe.Htm" is NOT compressed
[root@localhost /]# fsutil compress setflag /mnt/ntfs/1/ReadMe.Htm
OK
[root@localhost /]# fsutil compress queryflag /mnt/ntfs/1/ReadMe.Htm
File/Dir "/mnt/ntfs/1/ReadMe.Htm" is compressed
[root@localhost /]# fsutil compress clearflag /mnt/ntfs/1/ReadMe.Htm
OK
[root@localhost /]# fsutil compress queryflag /mnt/ntfs/1/ReadMe.Htm
File/Dir "/mnt/ntfs/1/ReadMe.Htm" is NOT compressed
[root@localhost /]# 
```

Fsutil: streams

This subcommand is intended for querying and dumping streams of a file. It shows Type, Id, Size and Name of all streams of the specified file. It also can show the raw data of the specified stream of a file as a dump.

Syntax

fsutil streams query <*filename*> – Query the list of streams.

E.g.: fsutil streams query /mnt/vol1/hello.txt

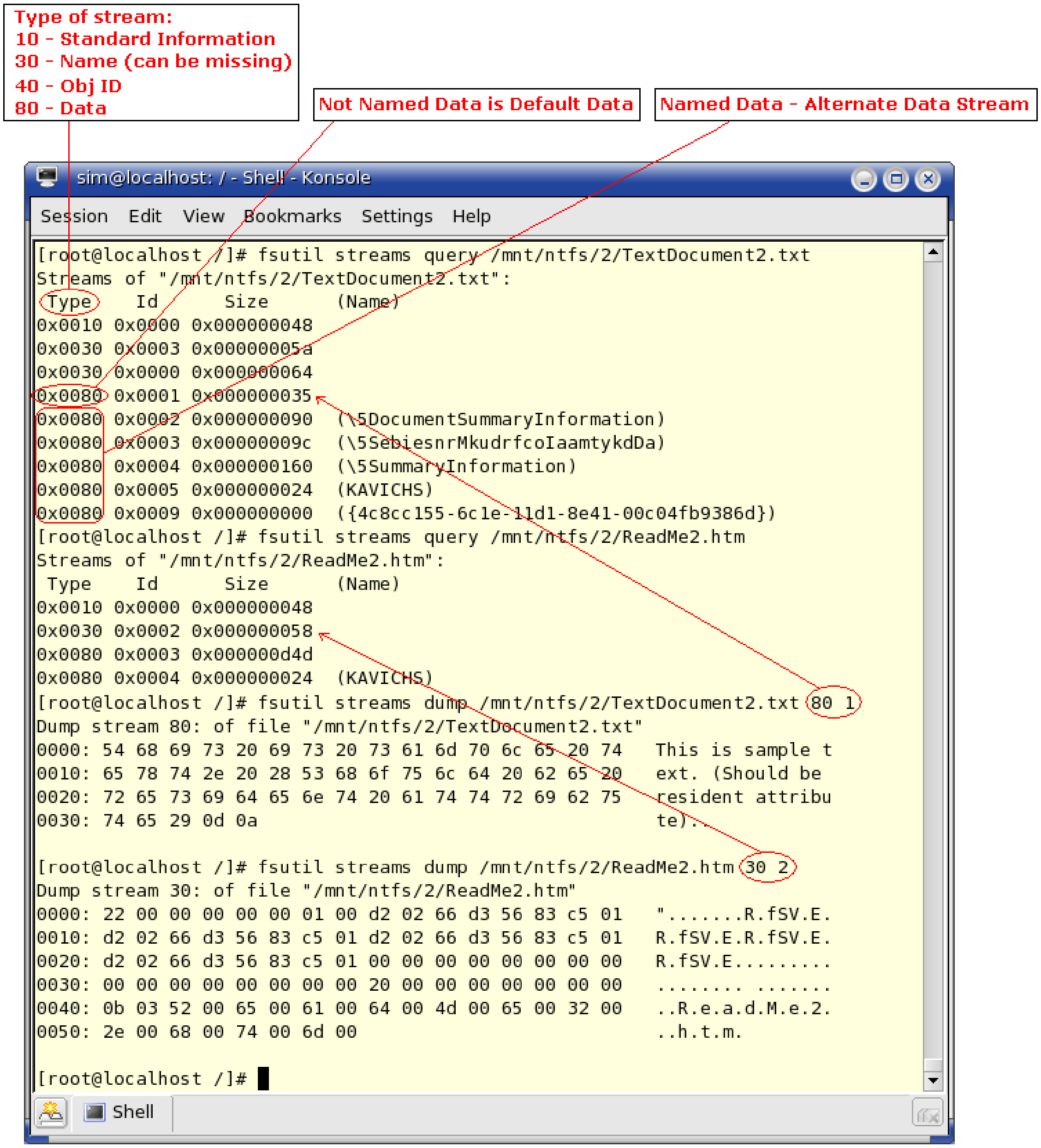
fsutil streams dump <*filename*> <*type(:name)*> (<*Id*>) - Dump the contents of stream.

E.g.: fsutil file streams dump /mnt/vol1/ hello.txt 10

E.g.: fsutil file streams dump /mnt/vol1/ hello.txt 30 1

E.g.: fsutil file streams dump /mnt/vol1/ hello.txt 90:$I30 2

Screenshots


Type of stream:
10 - Standard Information
30 - Name (can be missing)
40 - Obj ID
80 - Data
Not Named Data is Default Data
Named Data - Alternate Data Stream
sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# fsutil streams query /mnt/ntfs/2/TextDocument2.txt
Streams of "/mnt/ntfs/2/TextDocument2.txt":
 Type     Id       Size       (Name)
0x0010 0x0000 0x000000048
0x0030 0x0003 0x00000005a
0x0030 0x0000 0x000000064
0x0080 0x0001 0x000000035
0x0080 0x0002 0x000000090  (\5DocumentSummaryInformation)
0x0080 0x0003 0x00000009c  (\5SebiesnrMkudrfcoIaamtykdDa)
0x0080 0x0004 0x000000160  (\5SummaryInformation)
0x0080 0x0005 0x000000024  (KAVICHS)
0x0080 0x0009 0x000000000  ({4c8cc155-6c1e-11d1-8e41-00c04fb9386d})
[root@localhost /]# fsutil streams query /mnt/ntfs/2/ReadMe2.htm
Streams of "/mnt/ntfs/2/ReadMe2.htm":
 Type     Id       Size       (Name)
0x0010 0x0000 0x000000048
0x0030 0x0002 0x000000058
0x0080 0x0003 0x000000d4d
0x0080 0x0004 0x000000024  (KAVICHS)
[root@localhost /]# fsutil streams dump /mnt/ntfs/2/TextDocument2.txt 80 1
Dump stream 80: of file "/mnt/ntfs/2/TextDocument2.txt"
0000: 54 68 69 73 20 69 73 20 73 61 6d 70 6c 65 20 74   This is sample t
0010: 65 78 74 2e 20 28 53 68 6f 75 6c 64 20 62 65 20   ext. (Should be
0020: 72 65 73 69 64 65 6e 74 20 61 74 74 72 69 62 75   resident attribu
0030: 74 65 29 0d 0a                                    te)..

[root@localhost /]# fsutil streams dump /mnt/ntfs/2/ReadMe2.htm 30 2
Dump stream 30: of file "/mnt/ntfs/2/ReadMe2.htm"
0000: 22 00 00 00 00 00 01 00 d2 02 66 d3 56 83 c5 01   ".......R.fSV.E.
0010: d2 02 66 d3 56 83 c5 01 d2 02 66 d3 56 83 c5 01   R.fSV.E.R.fSV.E.
0020: d2 02 66 d3 56 83 c5 01 00 00 00 00 00 00 00 00   R.fSV.E.........
0030: 00 00 00 00 00 00 00 00 20 00 00 00 00 00 00 00   ........ .......
0040: 0b 03 52 00 65 00 61 00 64 00 4d 00 65 00 32 00   ..R.e.a.d.M.e.2.
0050: 2e 00 68 00 74 00 6d 00                           ..h.t.m.

[root@localhost /]#
```

Note: Every stream has a unique pair of **Type** and **Id**.

Fsutil: sparse

This subcommand manages sparse file. A sparse file is a file that is handled in a way that requires much less disk space than would otherwise be needed. Sparse support allows an application to create very large files without committing disk space for regions of the file that only contain zeros. For example, you can use sparse support to work with a 10GB file in which you need to write data only to the first 64 KB (the rest of the file is zeroed). In other words, all meaningful or nonzero data is allocated, whereas all not meaningful data (large strings of data composed of zeros) is not allocated. When a sparse file is read, allocated data is returned as stored and unallocated data is returned, by default, as zeros.

Syntax

fsutil sparse <setflag> <*filename*> (<-*r*>) – Set sparse.

E.g.: fsutil sparse setflag /mnt/vol1/hello.txt -r

fsutil sparse <queryflag> <*filename*> – Query sparse.

E.g.: fsutil sparse queryflag /mnt/vol1/hello.txt

fsutil sparse <queryrange> <*filename*> – Query range.

E.g.: fsutil sparse queryrange /mnt/vol1/hello.txt

fsutil sparse <setrange> <*filename*> <*beginning offset*> <*length*> – Set sparse range.

E.g.: fsutil sparse setrange /mnt/vol1/hello.txt 65536 131072

Options

setflag

Marks the indicated file as sparse.

queryflag

Queries sparse.

queryrange

Scans a file looking for ranges that may contain nonzero data.

setrange

Fills a specified range of a file with zeroes.

beginning offset

Offset within the file to mark as sparse.

length

Length of the region in the file to be marked as sparse, in bytes.

Remarks

- In a sparse file, large ranges of zeroes may not require disk allocation. Space for nonzero data will be allocated as needed as the file is written.
- Only compressed or sparse files can have zeroed ranges known to the operating system.
- If the file is sparse or compressed, NTFS may deallocate disk space within the file. This sets the range of bytes to zeroes without extending the file size.

9.1.10 CPNTFS Utility – Files and directories backup/restore utility.

Name

cpntfs – copy files/directories to/from/between NTFS volume(s).

Synopsis

cpntfs [options] <*source*> <*destination*>

Options

-e:ext	directory extension, default is “.nto”.
-p:size	the size of portion to read /write, default is 1M.
-i	interactive mode.
-a	abort the operation if error and “-i” is not set.
-r	copy directories recursively.
-s	copy pagefile.sys and hiberfil.sys.
-v	explain what is being done
-h	display this help.
--noatime	do not copy times (create, access, modification).
--trace	turn on UFSD trace.
--version	show the version and exit.

Description

Overview

The CPNTFS utility – is a standalone simple and fast Paragon NTFS for Linux backup/copy utility that allows to copy and restore files, directories to/from/between NTFS partitions under Linux OS. In other words, this utility provides the files/directories save/restore operations supporting all NTFS attributes (resident and non-resident). The CPNTFS utility can be useful to make backups of NTFS file system and restore it later on the same or any newly formatted NTFS partition.

Paragon NTFS for Linux driver (without the CPNTFS utility) enables to read files (retrieve data), but while rewriting it can happen to lose the NTFS attributes not supported by the other file systems. Some widely accepted attributes are transformed while writing according to the certain rules (default data, basic attributes (name, date, archive, hidden, system),…).

The CPNTFS utility provides the following main functionality:

- Copying NTFS files and directories including its attributes as an attached structure;
- Restoring previously saved files and folders to the NTFS partition providing full or partial recovery of the attributes with corresponding Meta Files correction.

NTFS File Attributes

The NTFS file system considers each file or folder as a set of attributes. Elements such as the file's name, its security information, and even its data, are all attributes. Each attribute is identified by an attribute type code or, optionally, an attribute name.

When file's attributes fit within the MFT file record, they are called resident attributes. For instance, information such as filename and timestamp are always included in the MFT file record. When file's attributes can't fit within the MFT file record, some of its attributes are non-resident. The non-resident attributes are allocated one or more clusters of disk space elsewhere in the volume. NTFS creates the Attribute List to describe the location of all of the attribute records.

The table below lists all file attributes currently used by the NTFS file system. This list is expandable, meaning that other file attributes can be defined in the future.

Attribute Type	Description
Standard Information	Includes standard information such as timestamp, link count and DOS File Permission (see below).
Attribute List	Lists the location of all attribute records do not fit in the MFT file record.
File Name	A repeatable attribute for both long and short file names. The long name of the file can be up to 255 Unicode characters. The short name is the 8.3 and

	case-insensitive. Additional names, or hard links can be included as additional file name attributes.
Security Descriptor	Describes who owns the file and who can access it.
Data	Contains file data. NTFS allows multiple data attributes per file. Each file typically has one unnamed data attribute. A file can also have one or more named data attributes (Alternate Data Streams (ADS)).
Object ID	A volume unique file identifier. Not all files have object identifiers.
Reparse Point	Used for volume mount points. They are also used by Installable File System (IFS) filter drivers to mark certain files as special to that driver.
Index Root, Index Allocation and Bitmap	Used to implement folders and other indexes.
Volume Information	Used only in the $Volume meta file. Contains the volume version.
Volume Name	Used only in the $Volume meta file. Contains the volume label.

DOS File Permissions

Flag	**Description**
0x0001	Read-Only
0x0002	Hidden
0x0004	System
0x0020	Archive
0x0040	Device
0x0080	Normal
0x0100	Temporary
0x0200	Sparse File
0x0400	Reparse Point
0x0800	Compressed
0x1000	Offline
0x2000	Not Content Indexed
0x4000	Encrypted

NTFS Multiple Data Streams

NTFS supports multiple data streams (Alternate Data Streams), where the stream name identifies a new data attribute for the file. It follows that a data stream is a unique set of file attributes.

The multiple data streams feature enables you to manage data as a single unit. The following is an example of an alternate data stream:

```
        myfile.txt:mystream1
```

A library of files may exist where the files are defined as alternate streams, for example:

```
library:file1
        :file2
        :file3
```

A file can be associated with more than one application at a time. For example, a file structure like the following shows file association, but not multiple files:

```
program:source_file
       :doc_file
       :excel_file
       :executable_file
```

To create an alternate data stream, you can type the following command at the command prompt:

```
echo any text>example.txt:stream_file
more<example.txt:stream_file
```

Conclusion

The CPNTFS utility copies files and directories to any supported file system under Linux (Ext2fs, Ext3fs, Reiser, FAT,…) and NTFS one, storing all their attributes that are not supported by the file system where you copy to (Security, Encryption, Compression, alternate data streams,…) as an attached structure.

In case you copy files and directories using the CPNTFS utility to NTFS files system (from any supported one) all attributes excepting Security attributes will be inherited by the NTFS file system where you copy to. The Security attributes will be copied as is. It follows that the object’s owner, primary group and administrator will get access only.

What is backing up files or directories?

Large servers have a different set of problems. Users frequently delete files by accident then ask for them to be restored. A file level backup (rather than partition level backup) makes this sort of request easier to handle because you backup files, that store data, with their attributes without free space of the disk. Using this backup image you are able to restore both the whole file system and single files or directories. Moreover, a complete file level backup of the operating system image may be useful if you run a large server, because you can get an exact replacement for the damaged machine in the fastest way.

There are two situations when you can find data that isn’t stored in the file system. The first is when data only exists in memory allocated to a running program (or process). This stuff simply can’t be backed up without taking extraordinary measures - but it’s virtually never used for anything other than transient working data (such as a password for accessing encrypted files). The other situation is data in a raw disk partition (without a file system). A few applications – in particularly databases such as Oracle — prefer to read and write disk blocks directly, rather than storing data in files. In this case, there are tools that let a Linux system administrator dump the entire partition to tape or another disk, but for more selective backup situations you’ll need to use the application’s own tools.

9.2 HFS+ utilities

There are 2 additional utilities for HFS+:

- **mkhfs** — format any partition as HFS+ under Linux;
- **chkhfs** — check HFS+ partition for integrity and (optionally) fix errors;

9.2.1 MKHFS Utility - Create an HFS volume on a partition.

Name

mkhfs — create an HFS+ volume on specified (block) device under Linux OS.

Synopsis

mkhfs [options] device

E.g.: mkhfs /dev/hdb1

Options

-q	Perform a quick format.
-v:label	Specify the volume label.
-a:size	Override the default allocation unit size. Default settings are strongly recommended for general use. 512, 1024, 2048, 4096, 8192, 16K, 32K and 64K are supported.
-f	Force the format without confirmation.

-j	make volume journaled.
-c	make volume **c**ase-sensitive.
--help	Display this help.
--trace	Turn on UFSD trace.
--verbose	Explain what is being done.
--version	Show the version and exit.

Description

mkhfs is a standalone utility that allows to format HFS+ partitions under Linux. It is used to create an HFS+ file system on a device (usually a disk partition).

9.2.2 CHKHFS Utility - Perform consistency checks on an HFS+ volume.

Name

chkhfs — provide consistency checking of a HFS volume and fix errors.

Synopsis

chkhfs device [options]

E.g.: chkhfs /dev/hdb1

Options

-f	Fix errors on the disk.
-a	Perform checks only if ‘dirty’ flag is set.
-h	Display this help.
--trace	Turn on UFSD trace.
--verbose	Explain what is being done.
--version	Show the version and exit.

Description

chkhfs creates and displays a status report about a HFS+ file system. **Chkhfs** also lists and corrects errors on the disk, if any (-f flag must be specified).

10. CPNTFS – Backup/restore Utility Workshop.

10.1 Overview

This chapter provides step-by-step instructions on using **CPNTFS** utility to backup and restore any NTFS volume or files and directories from NTFS volumes under Linux.

There are three kinds of the source and destination volumes for the **CPNTFS** utility:

1) NTFS volume is accessed via UFSD;
2) NTFS volume is accessed via the NTFS for Linux driver;
3) Native Linux Volume (Ext2/3, Linux Swap, Reiser, FAT(16,32)).

It is possible to use any source and destination volume combinations, except when the source and destination volumes are Native Linux ones.

10.2 The Issue

1. You need to create a backup copy of whole system disk with installed Windows OS.
2. You need to restore a backup copy of whole system disk that was created by the **CPNTFS** utility.
3. You need to create backup copy of files or directories that are located on NTFS volumes under Linux saving all their attributes and streams.
4. You need to restore files or directories, which were copied by the **CPNTFS** utility, to any NTFS volume.

10.3 The Solution

Restoring and backing up whole NTFS system disks, files and directories to/from NTFS volumes, saving all their attributes and streams, can be managed using the **CPNTFS** utility. This solution is recommended in the event you want to backup, restore NTFS volumes, files or directories under Linux.

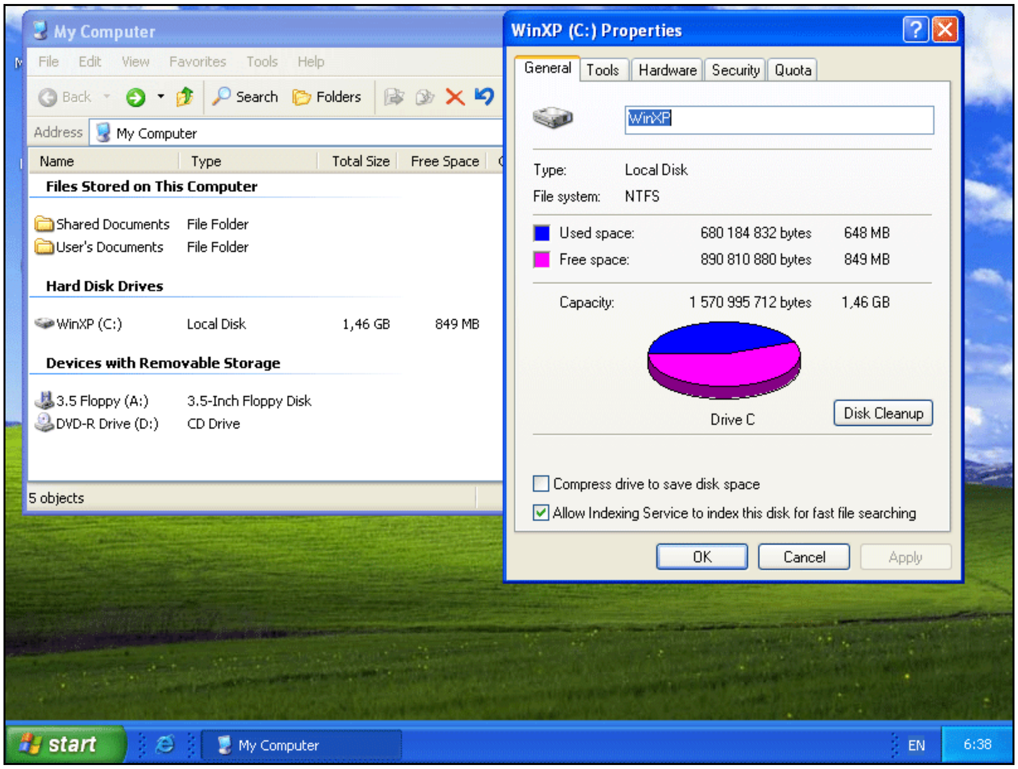
Note: The **CPNTFS** utility saves all attributes and streams of files and directories! It follows that after restoring to the same NTFS partition you will have exactly the same files or directories as before backing up.

10.4 Workshop – Whole NTFS System Disk Backup and Restore

This workshop describes how to create and restore a system disk (with installed Windows XP) under Linux using the **CPNTFS** utility.

In this workshop, there are two operating systems on a single PC – Windows and Linux. Each operation system uses its native file system: Windows – NTFS, Linux – Ext3.

Step№1. Windows is installed on a NTFS volume



Address: My Computer

Name	Type	Total Size	Free Space
Files Stored on This Computer			
Shared Documents	File Folder		
User's Documents	File Folder		
Hard Disk Drives			
WinXP (C:)	Local Disk	1,46 GB	849 MB
Devices with Removable Storage			
3.5 Floppy (A:)	3.5-Inch Floppy Disk		
DVD-R Drive (D:)	CD Drive		

5 objects

WinXP (C:) Properties

WinXP

Type: Local Disk

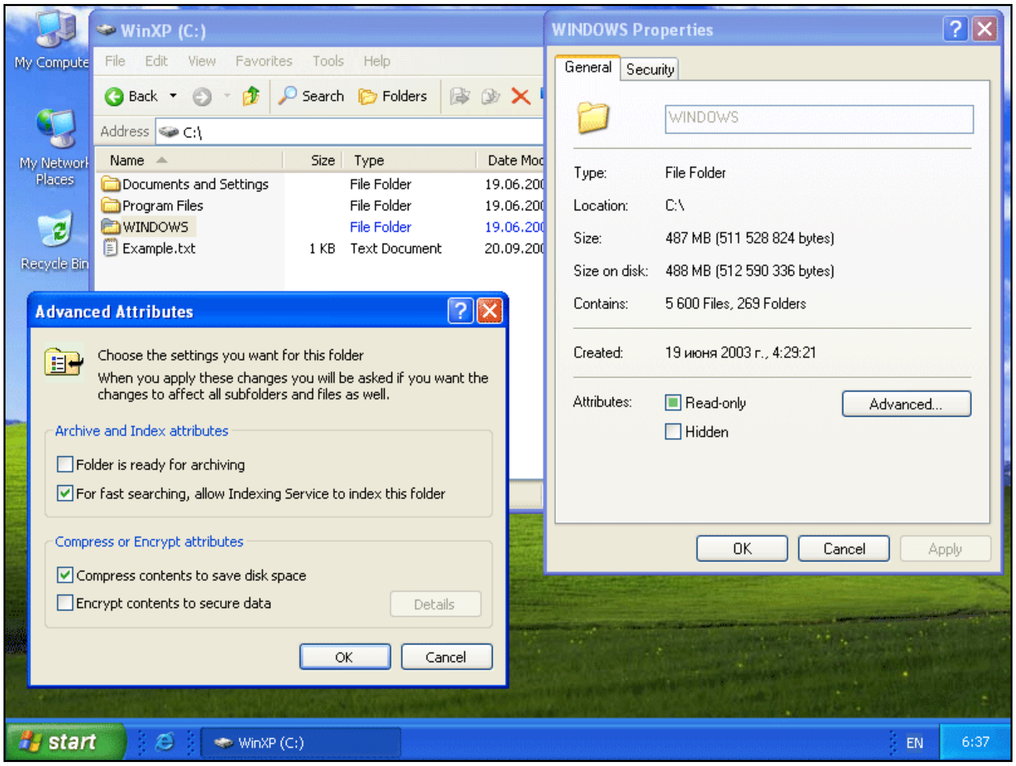
File system: NTFS

🟦 Used space:	680 184 832 bytes	648 MB
🟪 Free space:	890 810 880 bytes	849 MB
Capacity:	1 570 995 712 bytes	1,46 GB

□ Compress drive to save disk space

☑ Allow Indexing Service to index this disk for fast file searching

Windows folder:



Name	Size	Type	Date Mod
Documents and Settings		File Folder	19.06.200
Program Files		File Folder	19.06.200
WINDOWS		File Folder	19.06.200
Example.txt	1 KB	Text Document	20.09.200

WINDOWS Properties

General | Security

WINDOWS

Type: File Folder

Location: C:\

Size: 487 MB (511 528 824 bytes)

Size on disk: 488 MB (512 590 336 bytes)

Contains: 5 600 Files, 269 Folders

Created: 19 июня 2003 г., 4:29:21

Attributes: ■ Read-only □ Hidden   Advanced...

OK Cancel Apply

Advanced Attributes

Choose the settings you want for this folder

When you apply these changes you will be asked if you want the changes to affect all subfolders and files as well.

Archive and Index attributes

□ Folder is ready for archiving

☑ For fast searching, allow Indexing Service to index this folder

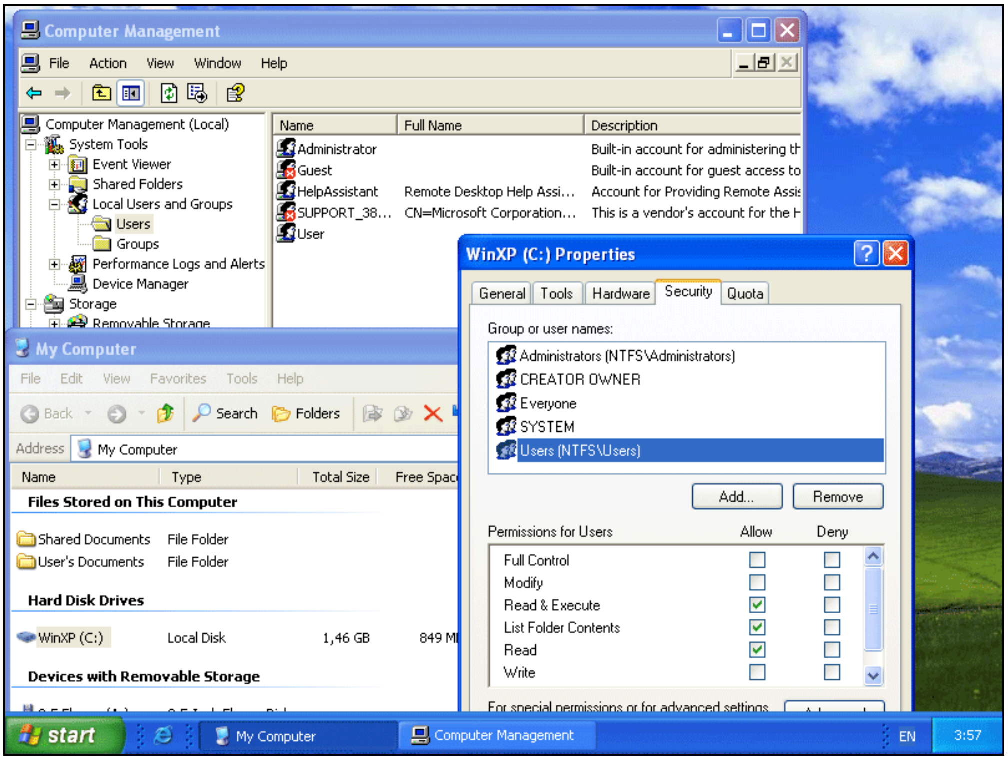
Compress or Encrypt attributes

☑ Compress contents to save disk space

□ Encrypt contents to secure data   Details

OK Cancel

Users and Security permissions:



Computer Management (Local)
- System Tools
  - Event Viewer
  - Shared Folders
  - Local Users and Groups
    - Users
    - Groups
  - Performance Logs and Alerts
  - Device Manager
- Storage
  - Removable Storage

Name	Full Name	Description
Administrator		Built-in account for administering th
Guest		Built-in account for guest access to
HelpAssistant	Remote Desktop Help Assi...	Account for Providing Remote Assis
SUPPORT_38...	CN=Microsoft Corporation...	This is a vendor's account for the H
User		

Name	Type	Total Size	Free Spac
Files Stored on This Computer			
Shared Documents	File Folder		
User's Documents	File Folder		
Hard Disk Drives			
WinXP (C:)	Local Disk	1,46 GB	849 MI
Devices with Removable Storage			

WinXP (C:) Properties

General | Tools | Hardware | Security | Quota

Group or user names:

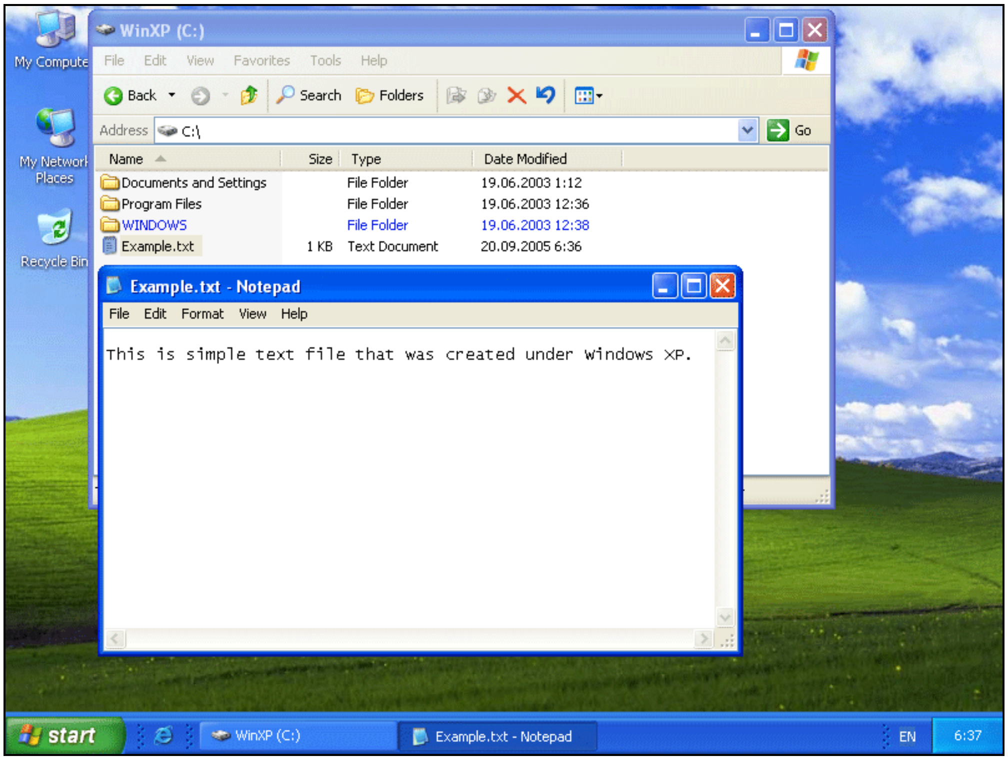
- Administrators (NTFS\Administrators)
- CREATOR OWNER
- Everyone
- SYSTEM
- Users (NTFS\Users)

Add... Remove

Permissions for Users	Allow	Deny
Full Control	□	□
Modify	□	□
Read & Execute	☑	□
List Folder Contents	☑	□
Read	☑	□
Write	□	□

For special permissions or for advanced settings,

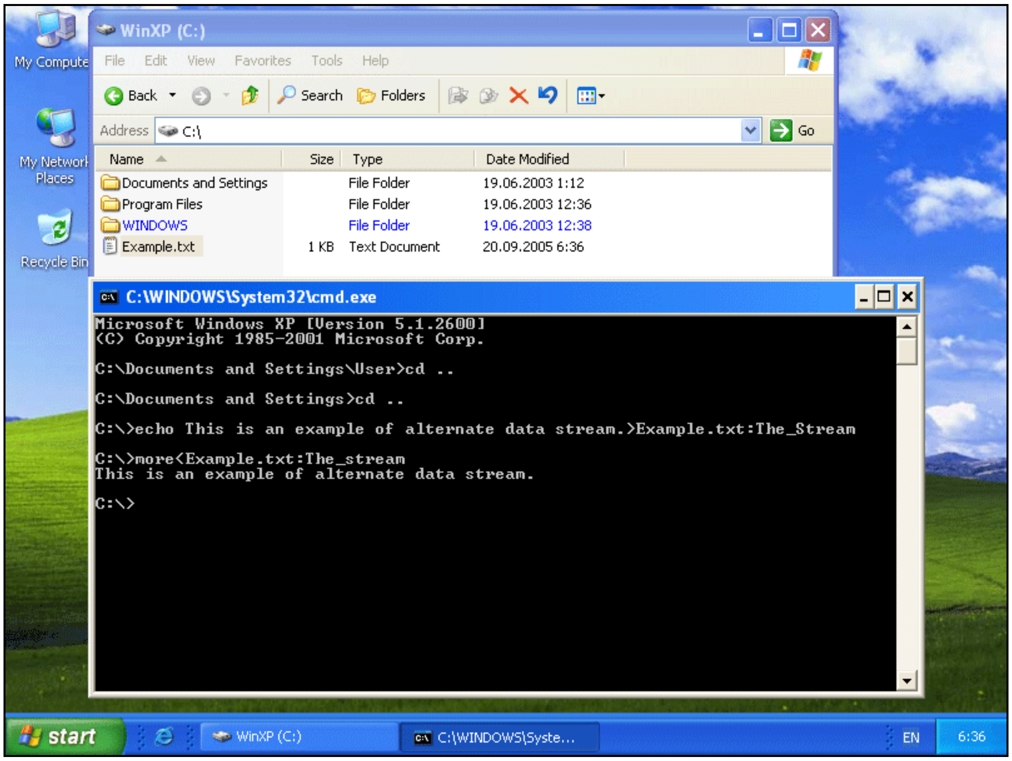
A text file:



Name	Size	Type	Date Modified
Documents and Settings		File Folder	19.06.2003 1:12
Program Files		File Folder	19.06.2003 12:36
WINDOWS		File Folder	19.06.2003 12:38
Example.txt	1 KB	Text Document	20.09.2005 6:36

```
This is simple text file that was created under Windows XP.
```

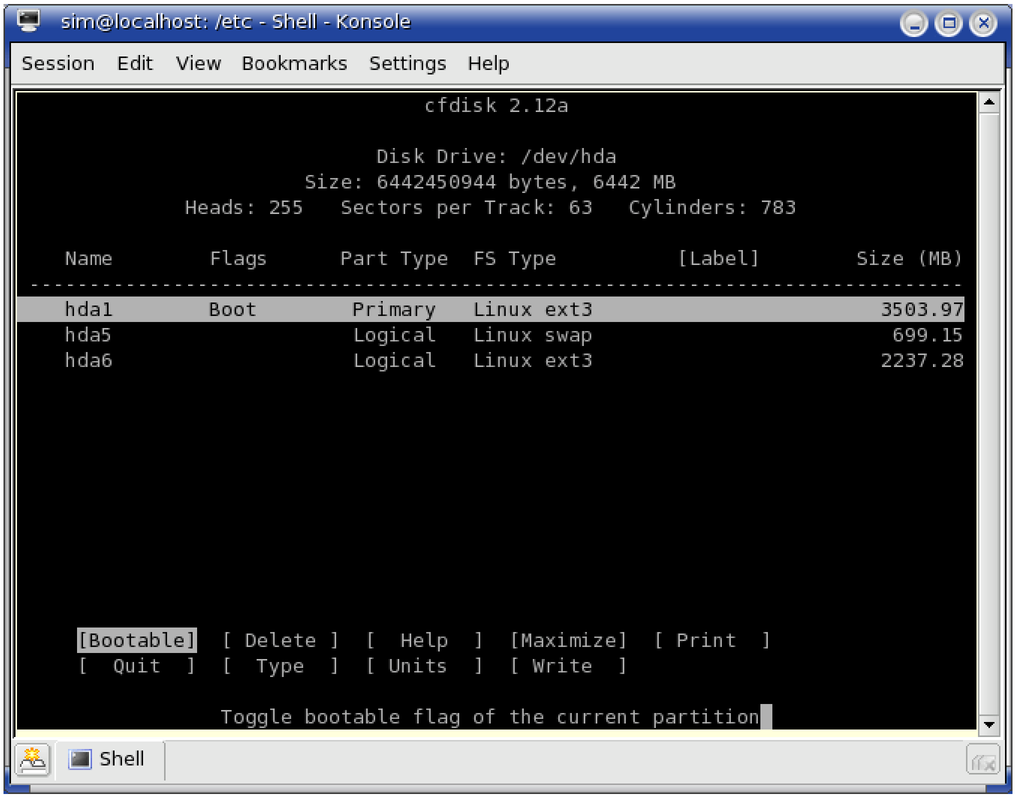
An alternate data stream:



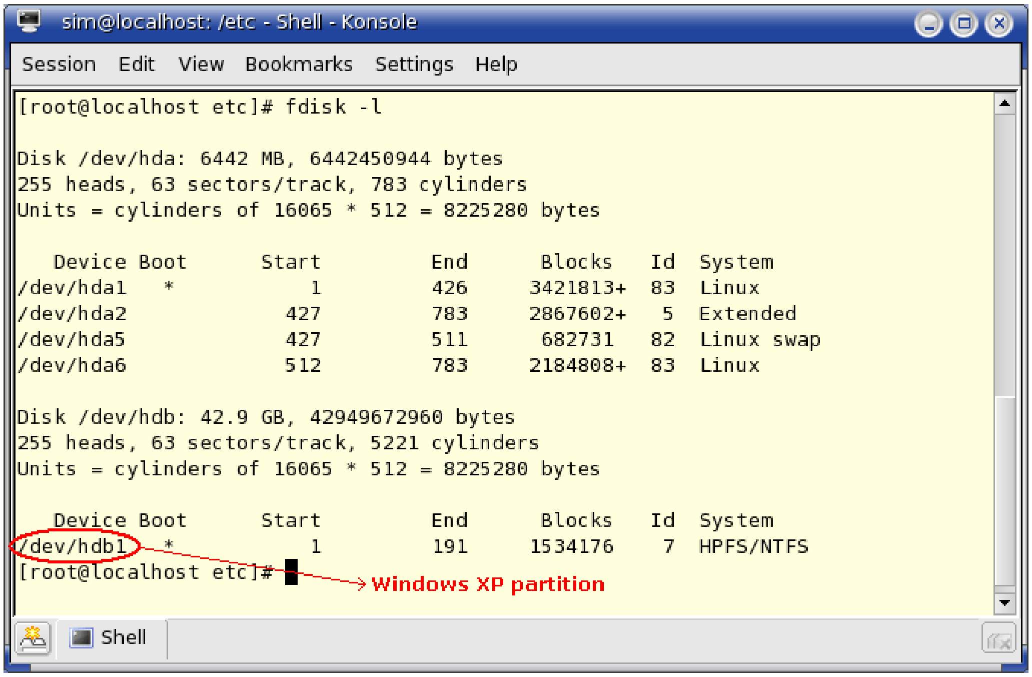
Name	Size	Type	Date Modified
Documents and Settings		File Folder	19.06.2003 1:12
Program Files		File Folder	19.06.2003 12:36
WINDOWS		File Folder	19.06.2003 12:38
Example.txt	1 KB	Text Document	20.09.2005 6:36

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\User>cd ..

C:\Documents and Settings>cd ..

C:\>echo This is an example of alternate data stream.>Example.txt:The_Stream

C:\>more<Example.txt:The_stream
This is an example of alternate data stream.

C:\>
```

Step№2. Linux is installed on Ext3 volume on the same PC



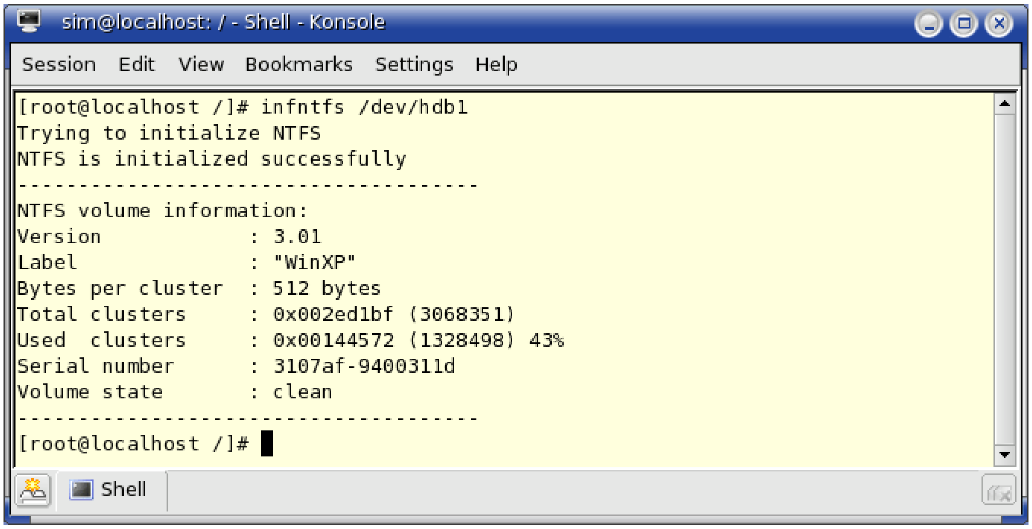
```
                                 cfdisk 2.12a

                              Disk Drive: /dev/hda
                        Size: 6442450944 bytes, 6442 MB
              Heads: 255   Sectors per Track: 63   Cylinders: 783

    Name         Flags        Part Type  FS Type          [Label]        Size (MB)
 ------------------------------------------------------------------------------
    hda1         Boot          Primary   Linux ext3                        3503.97
    hda5                       Logical   Linux swap                         699.15
    hda6                       Logical   Linux ext3                        2237.28

     [Bootable]  [ Delete ]  [  Help  ]  [Maximize]  [ Print  ]
     [  Quit  ]  [  Type  ]  [ Units  ]  [ Write  ]

                 Toggle bootable flag of the current partition
```



```
[root@localhost etc]# fdisk -l

Disk /dev/hda: 6442 MB, 6442450944 bytes
255 heads, 63 sectors/track, 783 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1   *           1         426     3421813+  83  Linux
/dev/hda2             427         783     2867602+   5  Extended
/dev/hda5             427         511      682731   82  Linux swap
/dev/hda6             512         783     2184808+  83  Linux

Disk /dev/hdb: 42.9 GB, 42949672960 bytes
255 heads, 63 sectors/track, 5221 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hdb1   *           1         191     1534176    7  HPFS/NTFS
[root@localhost etc]#
```

Windows XP partition

Step №3. Properties of the “Windows XP” partition.



sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# infntfs /dev/hdb1
Trying to initialize NTFS
NTFS is initialized successfully
---------------------------------------
NTFS volume information:
Version             : 3.01
Label               : "WinXP"
Bytes per cluster   : 512 bytes
Total clusters      : 0x002ed1bf (3068351)
Used  clusters      : 0x00144572 (1328498) 43%
Serial number       : 3107af-9400311d
Volume state        : clean
---------------------------------------
[root@localhost /]#
```

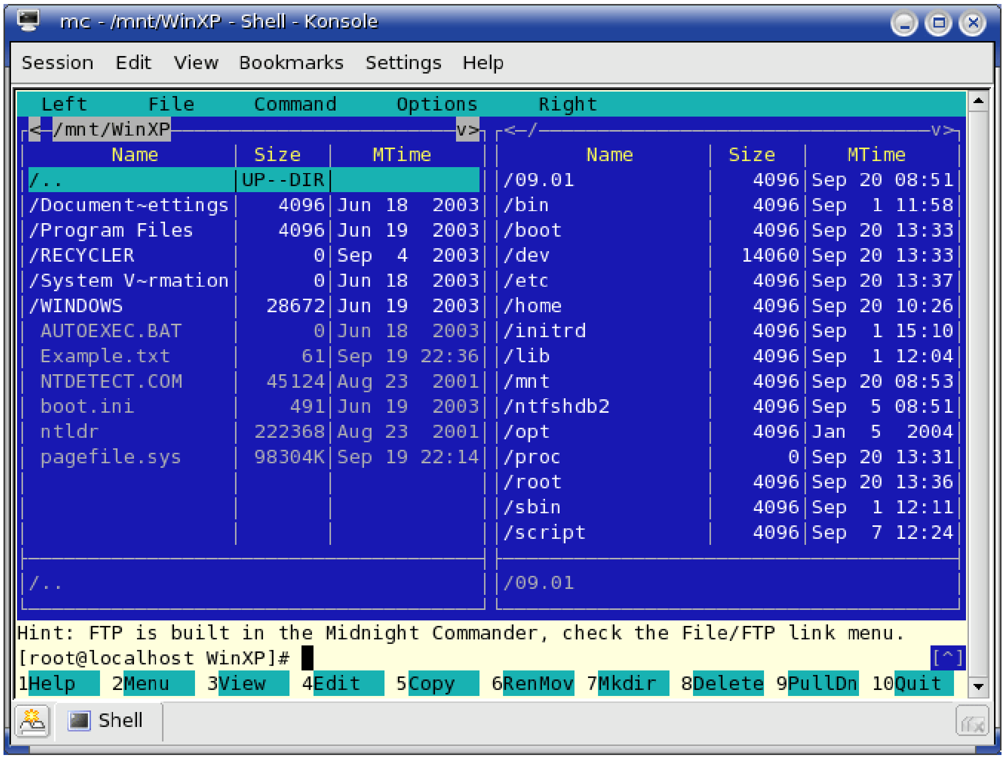
We will use the **INFNTFS** utility to see properties of the “Windows XP” partition before copying.

Note: Bytes per clusters - 512 bytes.

Step №4. Mounting “Windows XP” partition, just to show its contents

To mount a NTFS partition with installed Windows XP we may use the generic Linux mount command:

```
[root@localhost /]# mkdir /mnt/WinXP
[root@localhost /]# mount -t ntfs /dev/hdb1 /mnt/WinXP
```


mc - /mnt/WinXP - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
  Left     File     Command     Options     Right
┌<─/mnt/WinXP──────────────────────v>┐┌<─/───────────────────────────────v>┐
│        Name      │ Size  │  MTime  ││       Name       │ Size│  MTime    │
│/..               │UP--DIR│         ││/09.01            │ 4096│Sep 20 08:51│
│/Document~ettings │   4096│Jun 18  2003││/bin              │ 4096│Sep  1 11:58│
│/Program Files    │   4096│Jun 19  2003││/boot             │ 4096│Sep 20 13:33│
│/RECYCLER         │      0│Sep  4  2003││/dev              │14060│Sep 20 13:33│
│/System V~rmation │      0│Jun 18  2003││/etc              │ 4096│Sep 20 13:37│
│/WINDOWS          │  28672│Jun 19  2003││/home             │ 4096│Sep 20 10:26│
│ AUTOEXEC.BAT     │      0│Jun 18  2003││/initrd           │ 4096│Sep  1 15:10│
│ Example.txt      │     61│Sep 19 22:36││/lib              │ 4096│Sep  1 12:04│
│ NTDETECT.COM     │  45124│Aug 23  2001││/mnt              │ 4096│Sep 20 08:53│
│ boot.ini         │    491│Jun 19  2003││/ntfshdb2         │ 4096│Sep  5 08:51│
│ ntldr            │ 222368│Aug 23  2001││/opt              │ 4096│Jan  5  2004│
│ pagefile.sys     │ 98304K│Sep 19 22:14││/proc             │    0│Sep 20 13:31│
│                  │       │            ││/root             │ 4096│Sep 20 13:36│
│                  │       │            ││/sbin             │ 4096│Sep  1 12:11│
│                  │       │            ││/script           │ 4096│Sep  7 12:24│
├────────────────────────────────────┤├────────────────────────────────────┤
│/..                                 ││/09.01                              │
└────────────────────────────────────┘└────────────────────────────────────┘
Hint: FTP is built in the Midnight Commander, check the File/FTP link menu.
[root@localhost WinXP]#                                                    [^]
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir 8Delete 9PullDn 10Quit
```

Step №5. Attributes and streams of a file

For instance, we can use the **FSUTIL** utility to show all attributes and streams of a file to make sure that all attributes and streams will be saved after copying back.

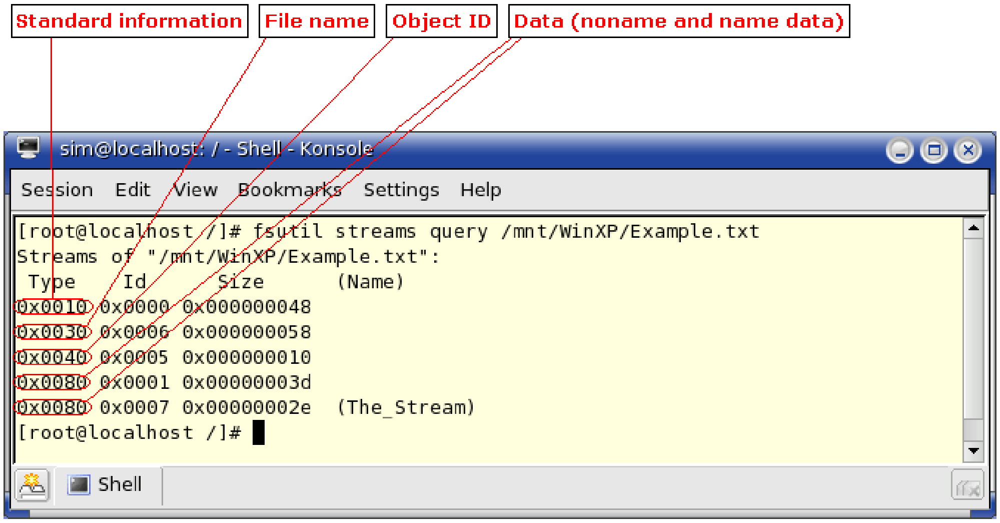
1. Mount the “Windows XP” partition via the NTFS for Linux driver:

   umount /dev/hdb1

   mount –t ufsd /dev/hdb1 /mnt/WinXP

2. Use the **fsutil streams query** command:

   fsutil streams query /mnt/WinXP/Example.txt



```
[root@localhost /]# fsutil streams query /mnt/WinXP/Example.txt
Streams of "/mnt/WinXP/Example.txt":
 Type    Id       Size      (Name)
0x0010 0x0000 0x000000048
0x0030 0x0006 0x000000058
0x0040 0x0005 0x000000010
0x0080 0x0001 0x00000003d
0x0080 0x0007 0x00000002e  (The_Stream)
[root@localhost /]#
```

As shown in the screenshot, the **Example.txt** file has the following attributes:

- Standard information;
- File name;
- Object ID;
- Data (no-name and name data). The name data is the alternate data stream that we have created under Windows XP. The no-name data is the default data.

Step №6. Copying “Windows XP” partition to Ext3FS by using the CPNTFS utility

Create a directory where you will copy the “Windows XP” partition:

mkdir /home/WinXPLinux

There are two ways to work with the CPNTFS utility:

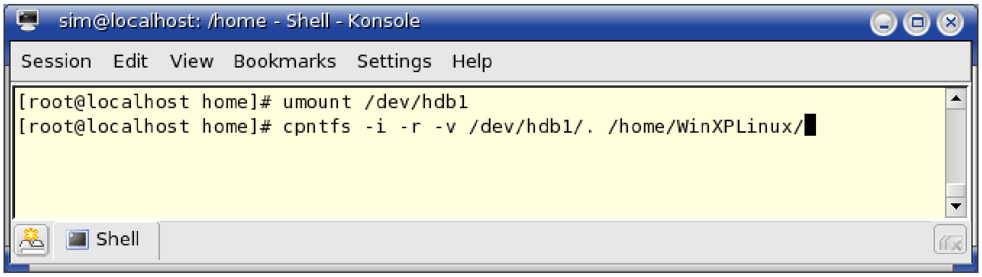
1) NTFS volume is accessed via the NTFS for Linux driver. It follows that before you can begin using the CPNTFS utility you should mount a NTFS partition via the NTFS for Linux driver:

   mount –t ufsd /dev/hdb1 /mnt/WinXP;

   cpntfs -i –r –v /dev/hdb1/. /home/WinXPLinux;

2) NTFS volume is accessed via UFSD. It follows that you access to a NTFS volume via the UFSD library, namely via CPNTFS utility only. In this case you must un-mount the NTFS

partition, if the partition was mounted by means of Linux, before using the CPNTFS utility.



```
[root@localhost home]# umount /dev/hdb1
[root@localhost home]# cpntfs -i -r -v /dev/hdb1/. /home/WinXPLinux/
```

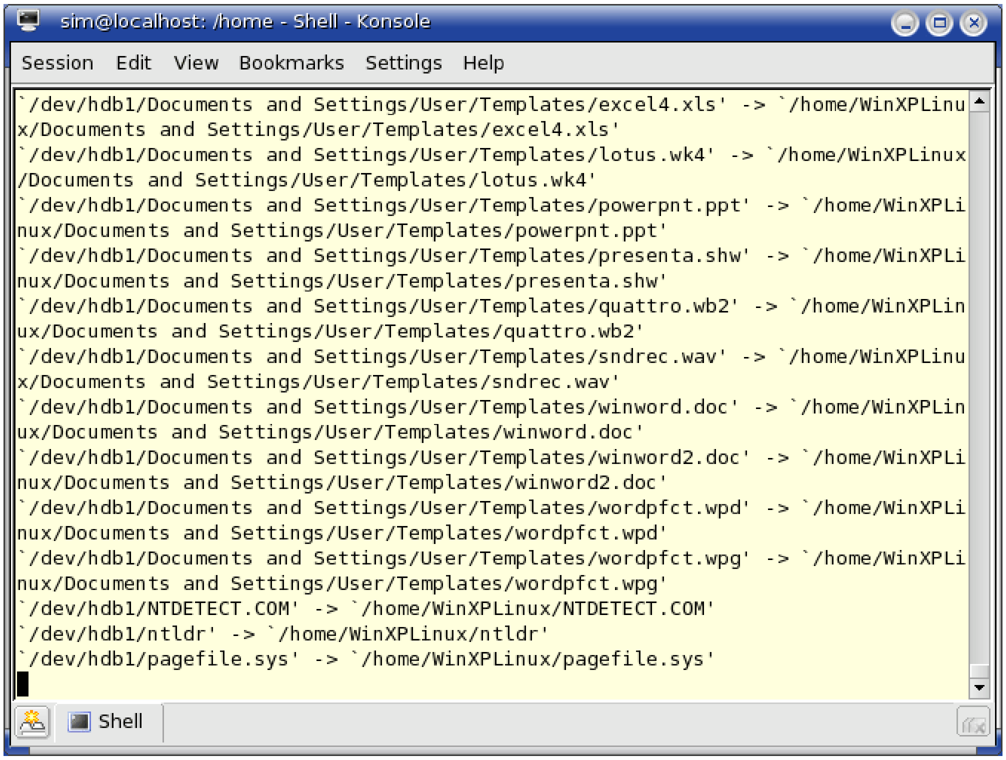
Note: To copy the whole volume (all files) you can specify either **“/.”** or **“/”** after a short name of the device, for example:

cpntfs –i –r –v /dev/hdb1/. /home/WinXPLinux

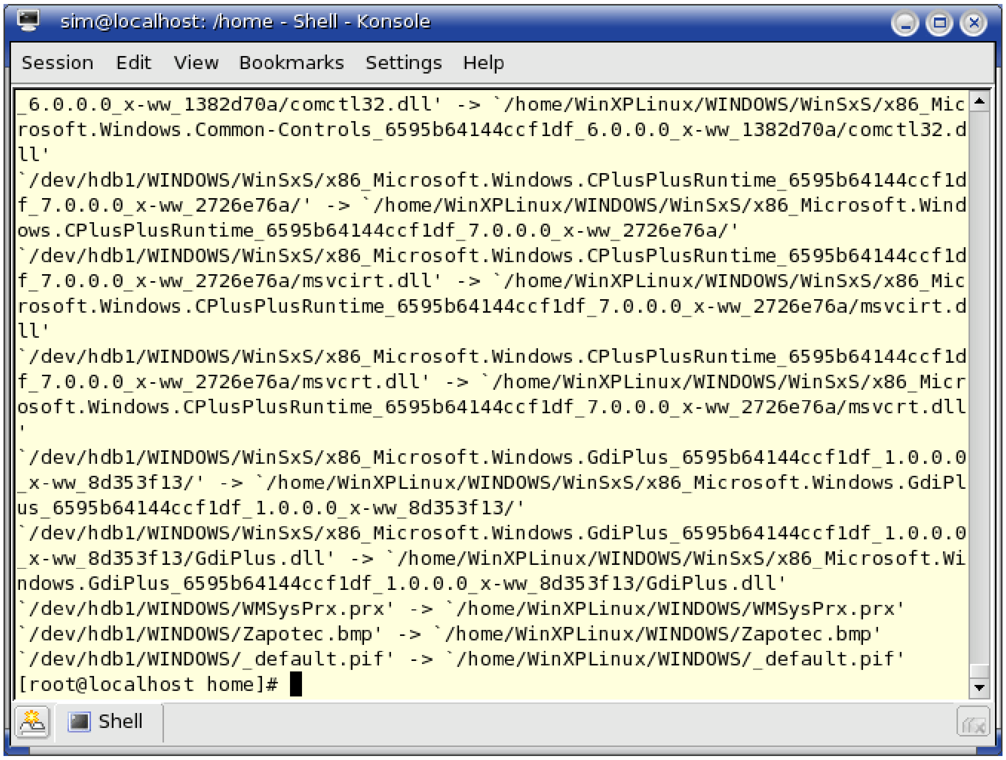
or

cpntfs –i –r –v /dev/hdb1/ /home/WinXPLinux

Copying…

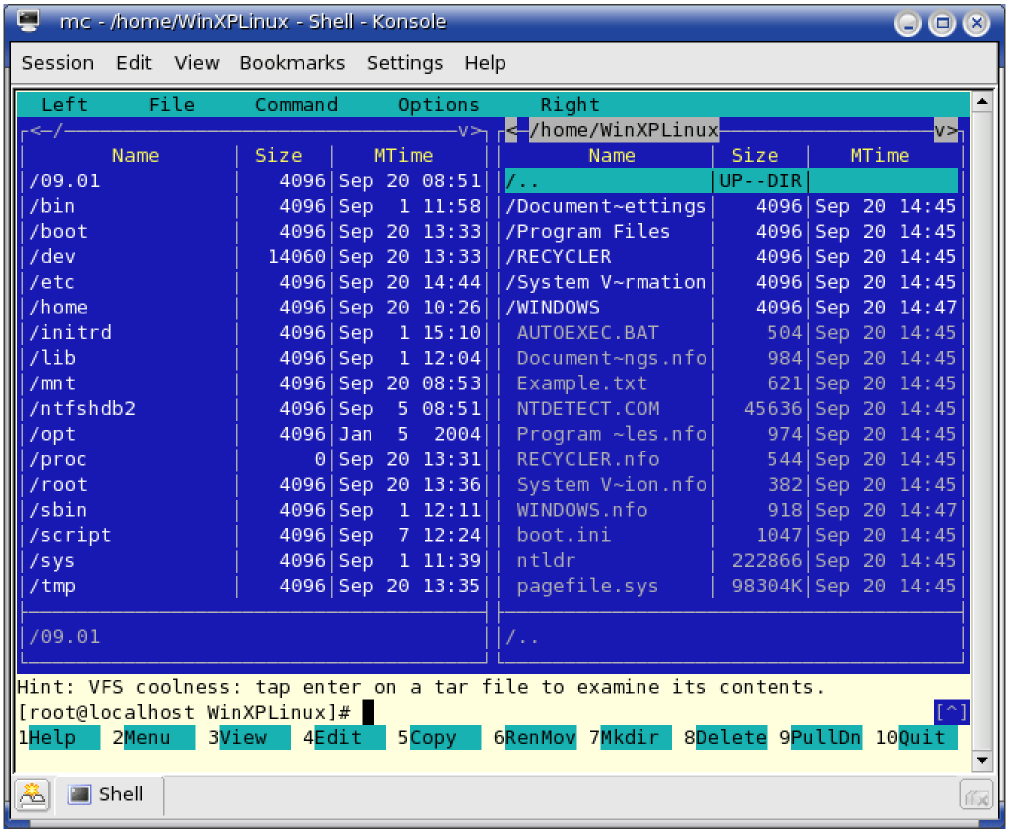


```
`/dev/hdb1/Documents and Settings/User/Templates/excel4.xls' -> `/home/WinXPLinu
x/Documents and Settings/User/Templates/excel4.xls'
`/dev/hdb1/Documents and Settings/User/Templates/lotus.wk4' -> `/home/WinXPLinux
/Documents and Settings/User/Templates/lotus.wk4'
`/dev/hdb1/Documents and Settings/User/Templates/powerpnt.ppt' -> `/home/WinXPLi
nux/Documents and Settings/User/Templates/powerpnt.ppt'
`/dev/hdb1/Documents and Settings/User/Templates/presenta.shw' -> `/home/WinXPLi
nux/Documents and Settings/User/Templates/presenta.shw'
`/dev/hdb1/Documents and Settings/User/Templates/quattro.wb2' -> `/home/WinXPLin
ux/Documents and Settings/User/Templates/quattro.wb2'
`/dev/hdb1/Documents and Settings/User/Templates/sndrec.wav' -> `/home/WinXPLinu
x/Documents and Settings/User/Templates/sndrec.wav'
`/dev/hdb1/Documents and Settings/User/Templates/winword.doc' -> `/home/WinXPLin
ux/Documents and Settings/User/Templates/winword.doc'
`/dev/hdb1/Documents and Settings/User/Templates/winword2.doc' -> `/home/WinXPLi
nux/Documents and Settings/User/Templates/winword2.doc'
`/dev/hdb1/Documents and Settings/User/Templates/wordpfct.wpd' -> `/home/WinXPLi
nux/Documents and Settings/User/Templates/wordpfct.wpd'
`/dev/hdb1/Documents and Settings/User/Templates/wordpfct.wpg' -> `/home/WinXPLi
nux/Documents and Settings/User/Templates/wordpfct.wpg'
`/dev/hdb1/NTDETECT.COM' -> `/home/WinXPLinux/NTDETECT.COM'
`/dev/hdb1/ntldr' -> `/home/WinXPLinux/ntldr'
`/dev/hdb1/pagefile.sys' -> `/home/WinXPLinux/pagefile.sys'
```


sim@localhost: /home - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

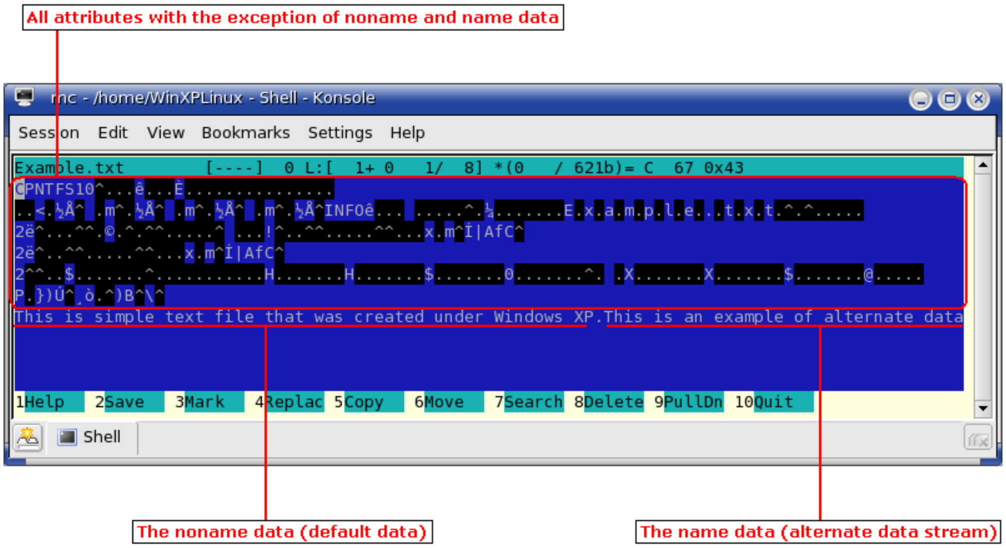
```
_6.0.0.0_x-ww_1382d70a/comctl32.dll' -> `/home/WinXPLinux/WINDOWS/WinSxS/x86_Mic
rosoft.Windows.Common-Controls_6595b64144ccf1df_6.0.0.0_x-ww_1382d70a/comctl32.d
ll'
`/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Windows.CPlusPlusRuntime_6595b64144ccf1d
f_7.0.0.0_x-ww_2726e76a/' -> `/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Wind
ows.CPlusPlusRuntime_6595b64144ccf1df_7.0.0.0_x-ww_2726e76a/'
`/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Windows.CPlusPlusRuntime_6595b64144ccf1d
f_7.0.0.0_x-ww_2726e76a/msvcirt.dll' -> `/home/WinXPLinux/WINDOWS/WinSxS/x86_Mic
rosoft.Windows.CPlusPlusRuntime_6595b64144ccf1df_7.0.0.0_x-ww_2726e76a/msvcirt.d
ll'
`/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Windows.CPlusPlusRuntime_6595b64144ccf1d
f_7.0.0.0_x-ww_2726e76a/msvcrt.dll' -> `/home/WinXPLinux/WINDOWS/WinSxS/x86_Micr
osoft.Windows.CPlusPlusRuntime_6595b64144ccf1df_7.0.0.0_x-ww_2726e76a/msvcrt.dll
'
`/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Windows.GdiPlus_6595b64144ccf1df_1.0.0.0
_x-ww_8d353f13/' -> `/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Windows.GdiPl
us_6595b64144ccf1df_1.0.0.0_x-ww_8d353f13/'
`/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Windows.GdiPlus_6595b64144ccf1df_1.0.0.0
_x-ww_8d353f13/GdiPlus.dll' -> `/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Wi
ndows.GdiPlus_6595b64144ccf1df_1.0.0.0_x-ww_8d353f13/GdiPlus.dll'
`/dev/hdb1/WINDOWS/WMSysPrx.prx' -> `/home/WinXPLinux/WINDOWS/WMSysPrx.prx'
`/dev/hdb1/WINDOWS/Zapotec.bmp' -> `/home/WinXPLinux/WINDOWS/Zapotec.bmp'
`/dev/hdb1/WINDOWS/_default.pif' -> `/home/WinXPLinux/WINDOWS/_default.pif'
[root@localhost home]#
```

Step №7. The Windows XP partition copying is complete


mc - /home/WinXPLinux - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
  Left     File     Command     Options     Right
┌<─/───────────────────────────────────v>┐┌<─/home/WinXPLinux───────────────────v>┐
│      Name        │ Size │   MTime      ││      Name        │ Size  │   MTime     │
│/09.01            │  4096│Sep 20 08:51  ││/..               │UP--DIR│             │
│/bin              │  4096│Sep  1 11:58  ││/Document~ettings │   4096│Sep 20 14:45 │
│/boot             │  4096│Sep 20 13:33  ││/Program Files    │   4096│Sep 20 14:45 │
│/dev              │ 14060│Sep 20 13:33  ││/RECYCLER         │   4096│Sep 20 14:45 │
│/etc              │  4096│Sep 20 14:44  ││/System V~rmation │   4096│Sep 20 14:45 │
│/home             │  4096│Sep 20 10:26  ││/WINDOWS          │   4096│Sep 20 14:47 │
│/initrd           │  4096│Sep  1 15:10  ││ AUTOEXEC.BAT     │    504│Sep 20 14:45 │
│/lib              │  4096│Sep  1 12:04  ││ Document~ngs.nfo │    984│Sep 20 14:45 │
│/mnt              │  4096│Sep 20 08:53  ││ Example.txt      │    621│Sep 20 14:45 │
│/ntfshdb2         │  4096│Sep  5 08:51  ││ NTDETECT.COM     │  45636│Sep 20 14:45 │
│/opt              │  4096│Jan  5  2004  ││ Program ~les.nfo │    974│Sep 20 14:45 │
│/proc             │     0│Sep 20 13:31  ││ RECYCLER.nfo     │    544│Sep 20 14:45 │
│/root             │  4096│Sep 20 13:36  ││ System V~ion.nfo │    382│Sep 20 14:45 │
│/sbin             │  4096│Sep  1 12:11  ││ WINDOWS.nfo      │    918│Sep 20 14:47 │
│/script           │  4096│Sep  7 12:24  ││ boot.ini         │   1047│Sep 20 14:45 │
│/sys              │  4096│Sep  1 11:39  ││ ntldr            │ 222866│Sep 20 14:45 │
│/tmp              │  4096│Sep 20 13:35  ││ pagefile.sys     │ 98304K│Sep 20 14:45 │
├──────────────────────────────────────────┤├──────────────────────────────────────────┤
│/09.01                                    ││/..                                       │
└──────────────────────────────────────────┘└──────────────────────────────────────────┘
Hint: VFS coolness: tap enter on a tar file to examine its contents.
[root@localhost WinXPLinux]#                                                     [^]
1Help   2Menu   3View   4Edit   5Copy   6RenMov 7Mkdir  8Delete 9PullDn 10Quit
```

The Example.txt file:



All attributes with the exception of noname and name data

```
Example.txt          [----]  0 L:[  1+ 0   1/  8] *(0   / 621b)= C  67 0x43
CPNTFS10^...ê...È...............
..<.½Å^ .m^.½Å^ .m^.½Å^ .m^.½Å^INFOê... .....^.¼.......E.x.a.m.p.l.e...t.x.t.^.^.....
2ë^...^^.©.^.^^.....^ ...!^..^^.....^^...x.m^Í|AfC^
2ë^..^^.....^^...x.m^Í|AfC^
2^^..$.......^...........H.......H.......$.......0.......^. .X.......X.......$.......@.....
P.})Ú^¸ò.^)B^\^
This is simple text file that was created under Windows XP.This is an example of alternate data
```

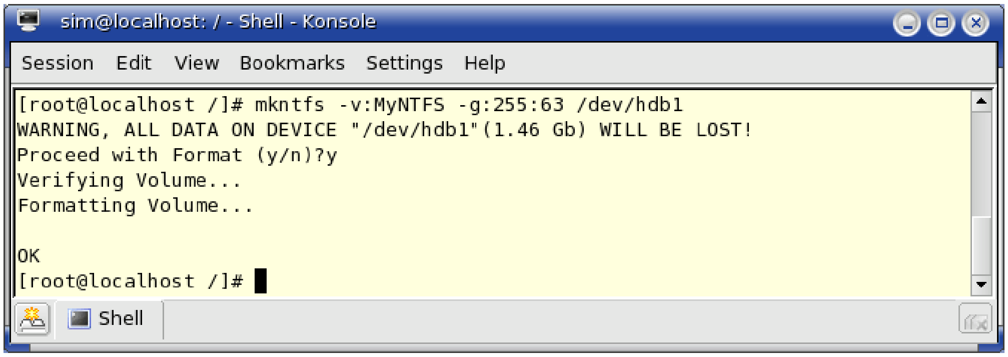
The noname data (default data)

The name data (alternate data stream)

mc - /home/WinXPLinux - Shell - Konsole
Session Edit View Bookmarks Settings Help
1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
Shell

Step №8. Formatting the “Windows XP” partition to simulate destruction of the partition

We will use the **MKNTFS** utility to format the partition. After formatting the partition all meta-files will be created.



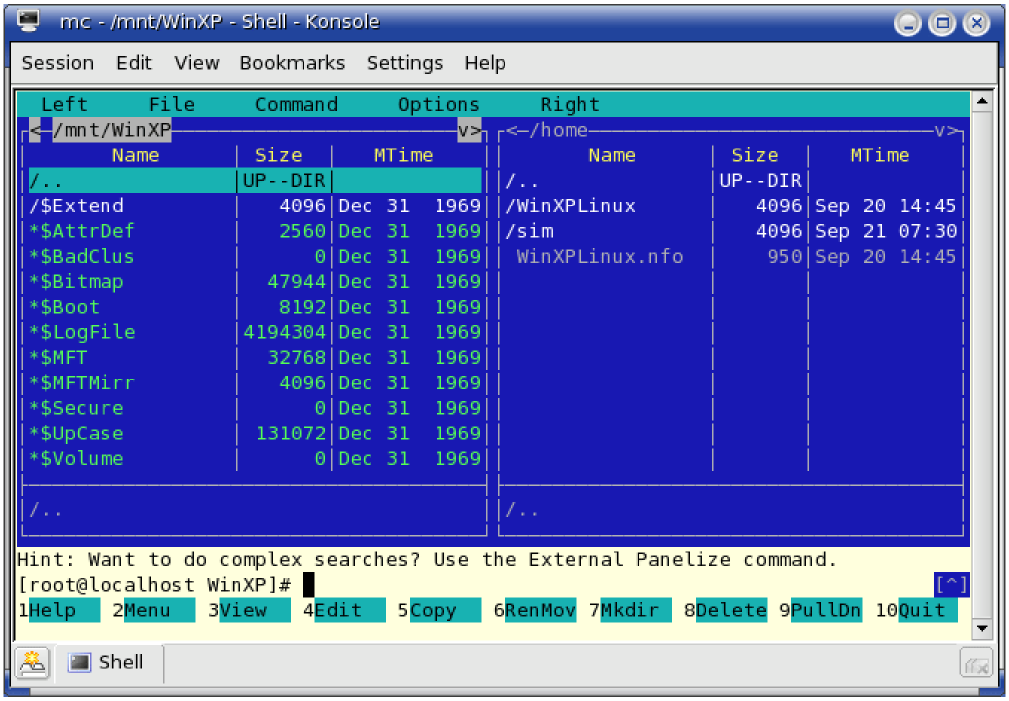
```
[root@localhost /]# mkntfs -v:MyNTFS -g:255:63 /dev/hdb1
WARNING, ALL DATA ON DEVICE "/dev/hdb1"(1.46 Gb) WILL BE LOST!
Proceed with Format (y/n)?y
Verifying Volume...
Formatting Volume...

OK
[root@localhost /]#
```

sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

Meta-files were created:

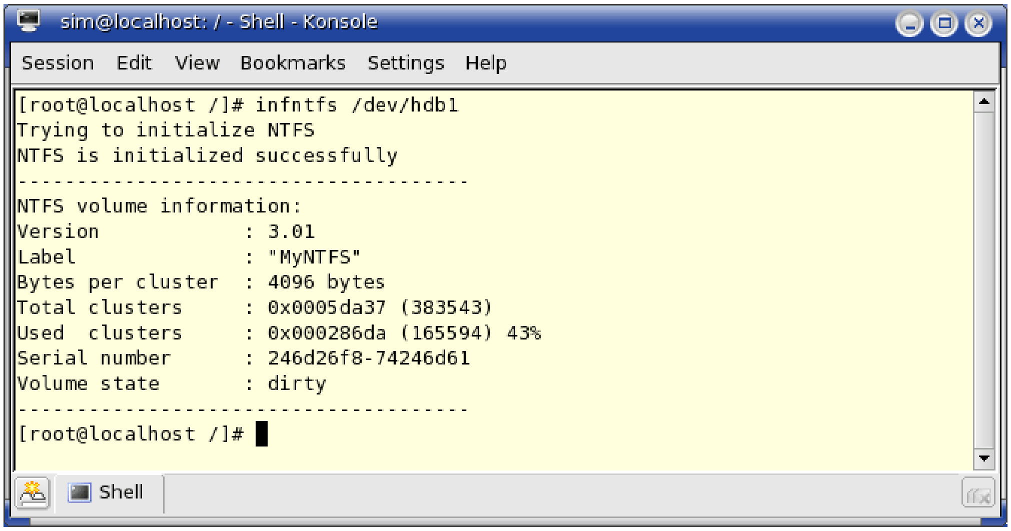
To show the meta-files we should mount the partition via the NTFS for Linux driver:

mount –t ufsd /dev/hdb1 /mnt/WinXP


mc - /mnt/WinXP - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
  Left     File      Command     Options     Right
┌<─/mnt/WinXP──────────────────────────v>┐┌<─/home────────────────────────────v>┐
│      Name        │  Size │   MTime     ││      Name        │  Size  │   MTime   │
│/..               │UP--DIR│             ││/..               │UP--DIR │           │
│/$Extend          │   4096│Dec 31  1969 ││/WinXPLinux       │    4096│Sep 20 14:45│
│*$AttrDef         │   2560│Dec 31  1969 ││/sim              │    4096│Sep 21 07:30│
│*$BadClus         │      0│Dec 31  1969 ││ WinXPLinux.nfo   │     950│Sep 20 14:45│
│*$Bitmap          │  47944│Dec 31  1969 ││                  │        │           │
│*$Boot            │   8192│Dec 31  1969 ││                  │        │           │
│*$LogFile         │4194304│Dec 31  1969 ││                  │        │           │
│*$MFT             │  32768│Dec 31  1969 ││                  │        │           │
│*$MFTMirr         │   4096│Dec 31  1969 ││                  │        │           │
│*$Secure          │      0│Dec 31  1969 ││                  │        │           │
│*$UpCase          │ 131072│Dec 31  1969 ││                  │        │           │
│*$Volume          │      0│Dec 31  1969 ││                  │        │           │
├──────────────────────────────────────────┤├─────────────────────────────────────┤
│/..                                       ││/..                                  │
└──────────────────────────────────────────┘└─────────────────────────────────────┘
Hint: Want to do complex searches? Use the External Panelize command.
[root@localhost WinXP]#                                                       [^]
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir  8Delete 9PullDn 10Quit
```

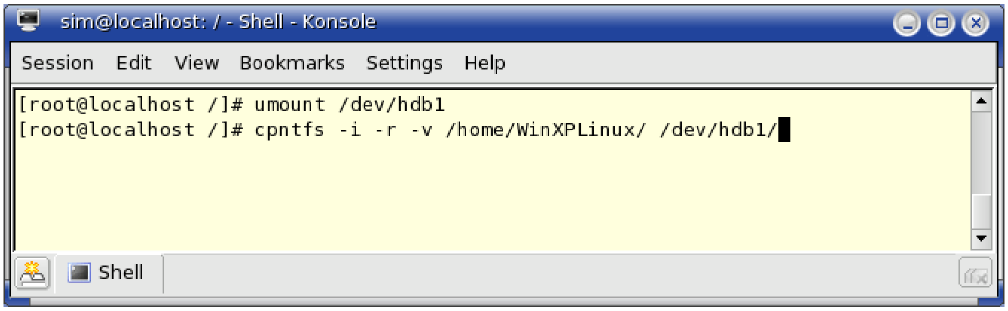
The **INFNTFS** utility will show us the new properties of the NTFS volume:


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

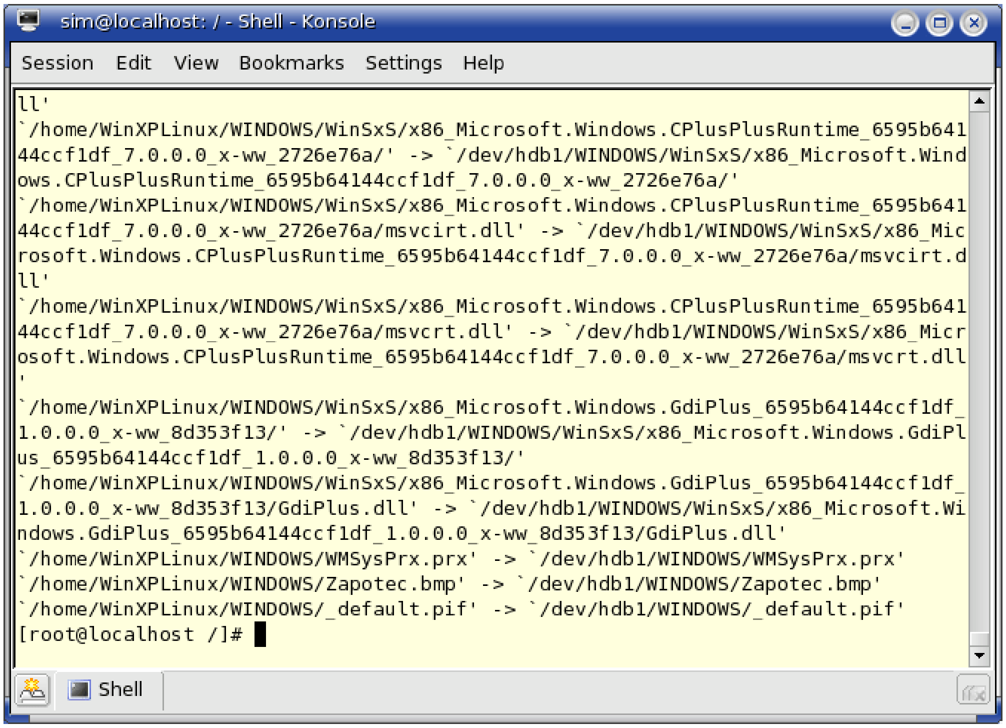
```
[root@localhost /]# infntfs /dev/hdb1
Trying to initialize NTFS
NTFS is initialized successfully
--------------------------------------
NTFS volume information:
Version            : 3.01
Label              : "MyNTFS"
Bytes per cluster  : 4096 bytes
Total clusters     : 0x0005da37 (383543)
Used  clusters     : 0x000286da (165594) 43%
Serial number      : 246d26f8-74246d61
Volume state       : dirty
--------------------------------------
[root@localhost /]#
```

Note: Bytes per clusters - 4096 bytes. The dirty flag is set, it follows that Windows suggests you to check the partition at start up. You can clear this flag using the **INFNTFS** and **FSUTIL** utilities.

Step №9. Copying all Files and Folders from the WinXPLinux Directory to the Formatted NTFS Partition

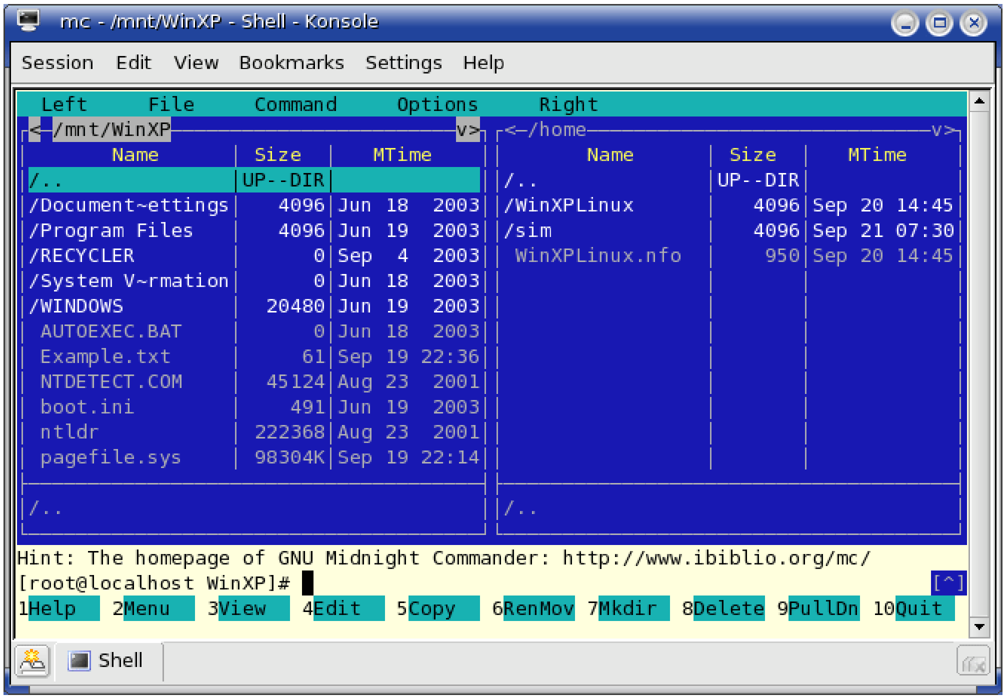

sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# umount /dev/hdb1
[root@localhost /]# cpntfs -i -r -v /home/WinXPLinux/ /dev/hdb1/
```


sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

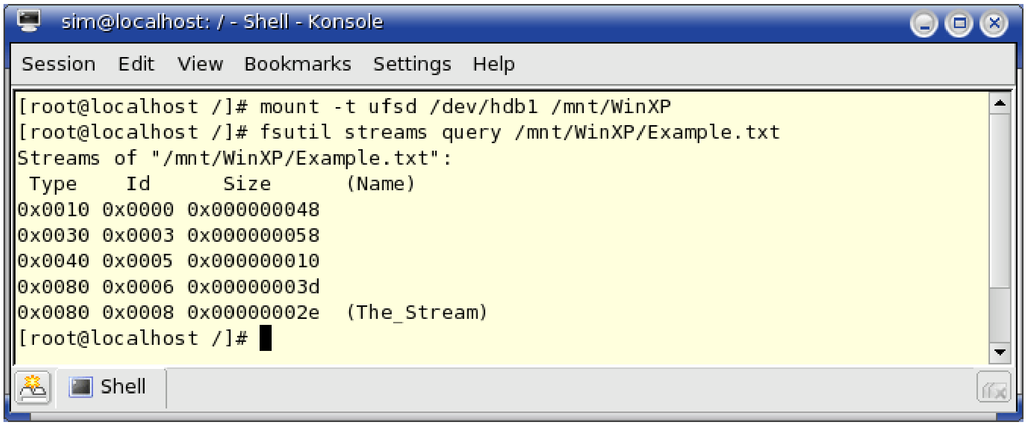
```
ll'
`/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Windows.CPlusPlusRuntime_6595b641
44ccf1df_7.0.0.0_x-ww_2726e76a/' -> `/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Wind
ows.CPlusPlusRuntime_6595b64144ccf1df_7.0.0.0_x-ww_2726e76a/'
`/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Windows.CPlusPlusRuntime_6595b641
44ccf1df_7.0.0.0_x-ww_2726e76a/msvcirt.dll' -> `/dev/hdb1/WINDOWS/WinSxS/x86_Mic
rosoft.Windows.CPlusPlusRuntime_6595b64144ccf1df_7.0.0.0_x-ww_2726e76a/msvcirt.d
ll'
`/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Windows.CPlusPlusRuntime_6595b641
44ccf1df_7.0.0.0_x-ww_2726e76a/msvcrt.dll' -> `/dev/hdb1/WINDOWS/WinSxS/x86_Micr
osoft.Windows.CPlusPlusRuntime_6595b64144ccf1df_7.0.0.0_x-ww_2726e76a/msvcrt.dll
'
`/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Windows.GdiPlus_6595b64144ccf1df_
1.0.0.0_x-ww_8d353f13/' -> `/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Windows.GdiPl
us_6595b64144ccf1df_1.0.0.0_x-ww_8d353f13/'
`/home/WinXPLinux/WINDOWS/WinSxS/x86_Microsoft.Windows.GdiPlus_6595b64144ccf1df_
1.0.0.0_x-ww_8d353f13/GdiPlus.dll' -> `/dev/hdb1/WINDOWS/WinSxS/x86_Microsoft.Wi
ndows.GdiPlus_6595b64144ccf1df_1.0.0.0_x-ww_8d353f13/GdiPlus.dll'
`/home/WinXPLinux/WINDOWS/WMSysPrx.prx' -> `/dev/hdb1/WINDOWS/WMSysPrx.prx'
`/home/WinXPLinux/WINDOWS/Zapotec.bmp' -> `/dev/hdb1/WINDOWS/Zapotec.bmp'
`/home/WinXPLinux/WINDOWS/_default.pif' -> `/dev/hdb1/WINDOWS/_default.pif'
[root@localhost /]#
```

Step №10. The “Windows XP” partition after the copying


mc - /mnt/WinXP - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
  Left      File      Command      Options      Right
┌<─/mnt/WinXP──────────────────────────v>┐┌<─/home──────────────────────────────v>┐
│      Name        │  Size  │   MTime     ││      Name       │  Size  │   MTime     │
│/..               │UP--DIR │             ││/..              │UP--DIR │             │
│/Document~ettings │    4096│Jun 18  2003 ││/WinXPLinux      │    4096│Sep 20 14:45 │
│/Program Files    │    4096│Jun 19  2003 ││/sim             │    4096│Sep 21 07:30 │
│/RECYCLER         │       0│Sep  4  2003 ││ WinXPLinux.nfo  │     950│Sep 20 14:45 │
│/System V~rmation │       0│Jun 18  2003 ││                 │        │             │
│/WINDOWS          │   20480│Jun 19  2003 ││                 │        │             │
│ AUTOEXEC.BAT     │       0│Jun 18  2003 ││                 │        │             │
│ Example.txt      │      61│Sep 19 22:36 ││                 │        │             │
│ NTDETECT.COM     │   45124│Aug 23  2001 ││                 │        │             │
│ boot.ini         │     491│Jun 19  2003 ││                 │        │             │
│ ntldr            │  222368│Aug 23  2001 ││                 │        │             │
│ pagefile.sys     │  98304K│Sep 19 22:14 ││                 │        │             │
├─────────────────────────────────────────┤├───────────────────────────────────────┤
│/..                                      ││/..                                    │
└─────────────────────────────────────────┘└───────────────────────────────────────┘
Hint: The homepage of GNU Midnight Commander: http://www.ibiblio.org/mc/
[root@localhost WinXP]#                                                        [^]
1Help  2Menu  3View  4Edit  5Copy  6RenMov 7Mkdir 8Delete 9PullDn 10Quit
```

Attributes and streams of the Example.txt file:

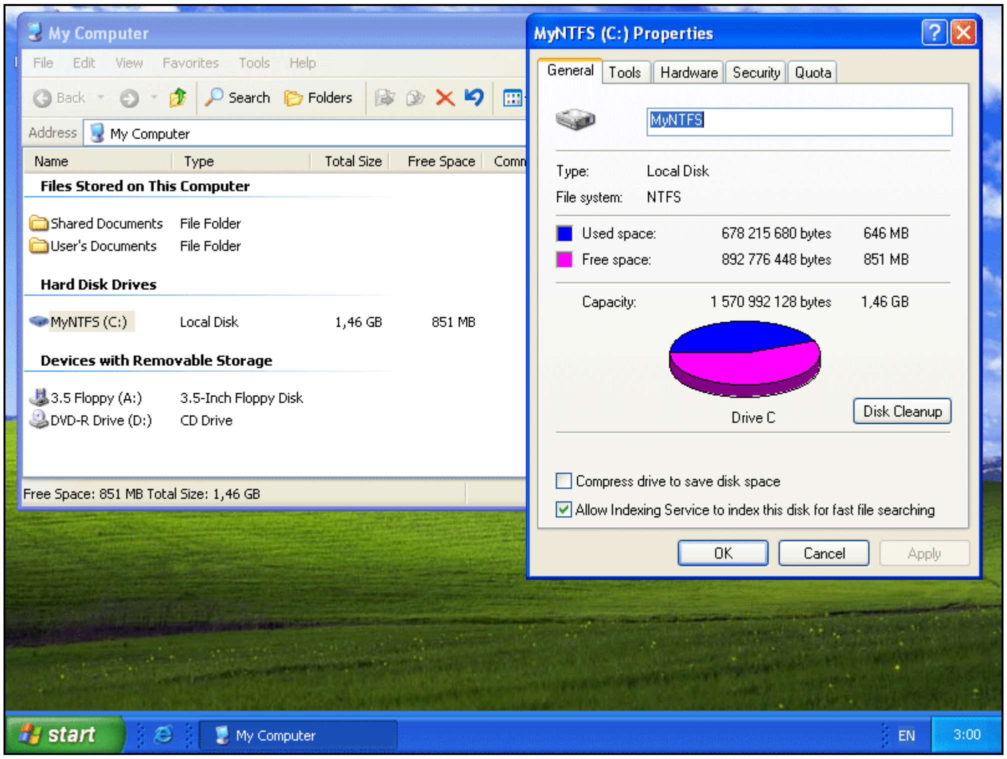

sim@localhost: / - Shell - Konsole
Session Edit View Bookmarks Settings Help
Shell

```
[root@localhost /]# mount -t ufsd /dev/hdb1 /mnt/WinXP
[root@localhost /]# fsutil streams query /mnt/WinXP/Example.txt
Streams of "/mnt/WinXP/Example.txt":
 Type    Id       Size       (Name)
0x0010 0x0000 0x000000048
0x0030 0x0003 0x000000058
0x0040 0x0005 0x000000010
0x0080 0x0006 0x00000003d
0x0080 0x0008 0x00000002e  (The_Stream)
[root@localhost /]#
```

As you can see all attributes and streams were saved.

Step №11. Booting up Windows XP after copying the files the folders back

After all files and folders were copied to the Windows XP partition, we will boot up Windows XP:



Name	Type	Total Size	Free Space	Comm
Files Stored on This Computer				
Shared Documents	File Folder			
User's Documents	File Folder			
Hard Disk Drives				
MyNTFS (C:)	Local Disk	1,46 GB	851 MB	
Devices with Removable Storage				
3.5 Floppy (A:)	3.5-Inch Floppy Disk			
DVD-R Drive (D:)	CD Drive			

Free Space: 851 MB Total Size: 1,46 GB

MyNTFS (C:) Properties

General | Tools | Hardware | Security | Quota

MyNTFS

Type: Local Disk
File system: NTFS

🟦 Used space:	678 215 680 bytes	646 MB
🟪 Free space:	892 776 448 bytes	851 MB
Capacity:	1 570 992 128 bytes	1,46 GB

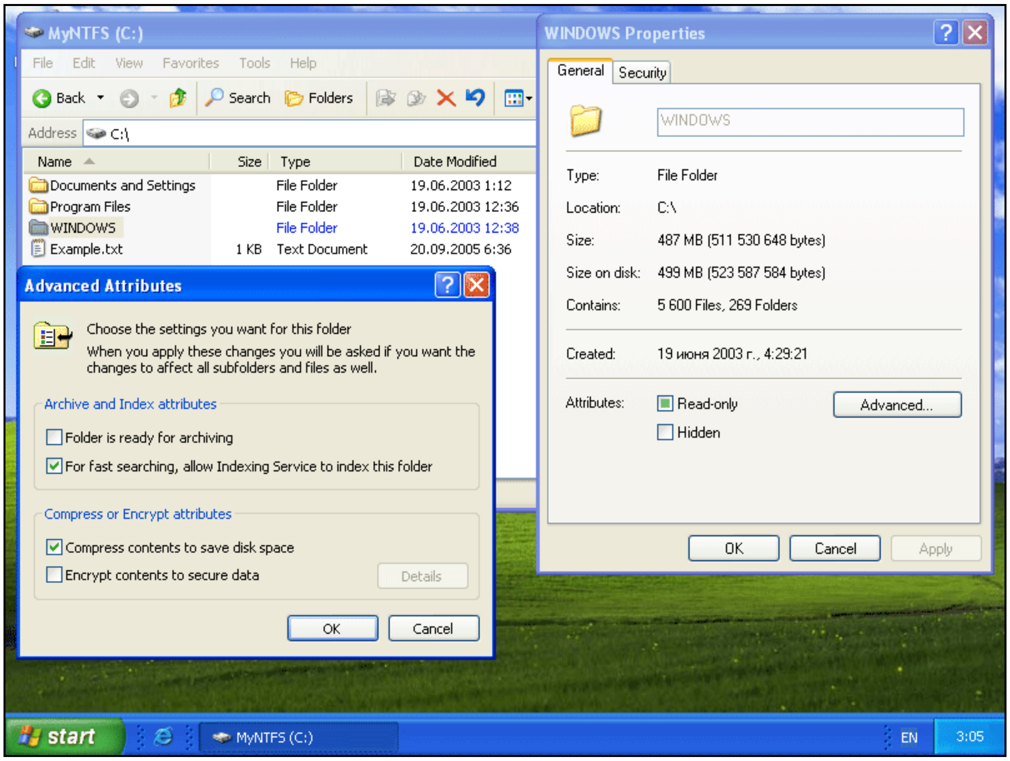
Drive C

Disk Cleanup

☐ Compress drive to save disk space
☑ Allow Indexing Service to index this disk for fast file searching

OK Cancel Apply

Windows folder:



Name	Size	Type	Date Modified
Documents and Settings		File Folder	19.06.2003 1:12
Program Files		File Folder	19.06.2003 12:36
WINDOWS		File Folder	19.06.2003 12:38
Example.txt	1 KB	Text Document	20.09.2005 6:36

Advanced Attributes

Choose the settings you want for this folder

When you apply these changes you will be asked if you want the changes to affect all subfolders and files as well.

Archive and Index attributes

☐ Folder is ready for archiving
☑ For fast searching, allow Indexing Service to index this folder

Compress or Encrypt attributes

☑ Compress contents to save disk space
☐ Encrypt contents to secure data Details

OK Cancel

WINDOWS Properties

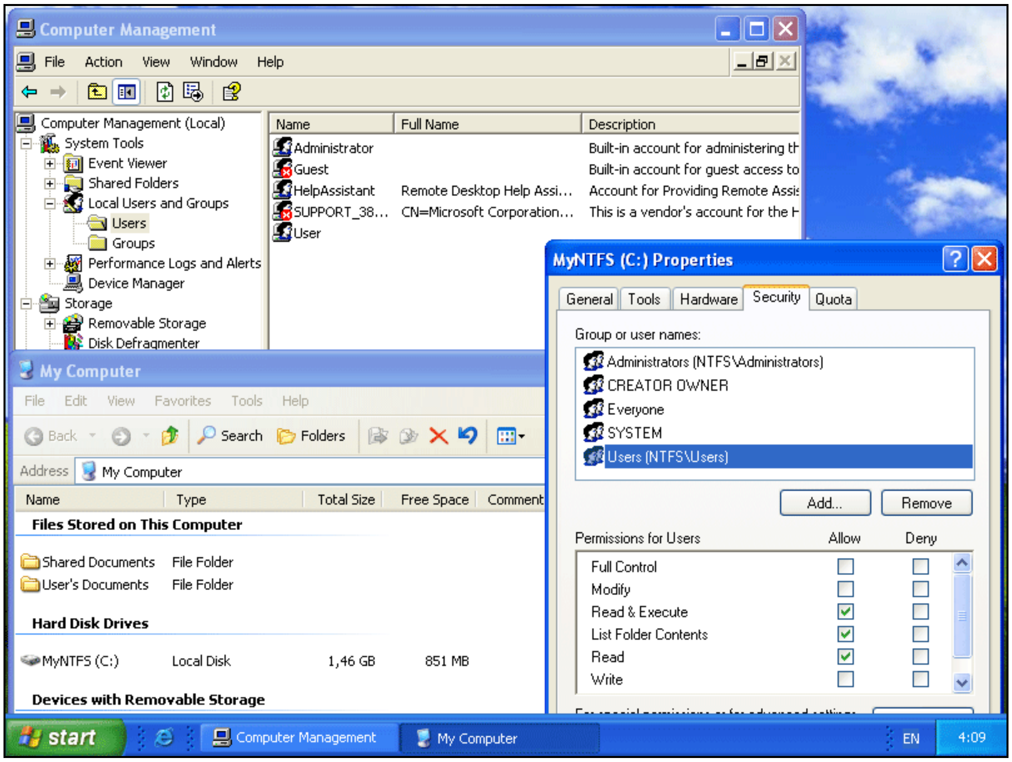
General | Security

WINDOWS

Type: File Folder
Location: C:\
Size: 487 MB (511 530 648 bytes)
Size on disk: 499 MB (523 587 584 bytes)
Contains: 5 600 Files, 269 Folders
Created: 19 июня 2003 г., 4:29:21
Attributes: ☑ Read-only Advanced...
☐ Hidden

OK Cancel Apply

Users and Security permissions after copying:



Name	Full Name	Description
Administrator		Built-in account for administering th
Guest		Built-in account for guest access to
HelpAssistant	Remote Desktop Help Assi...	Account for Providing Remote Assis
SUPPORT_38...	CN=Microsoft Corporation...	This is a vendor's account for the H
User		

My Computer

Address: My Computer

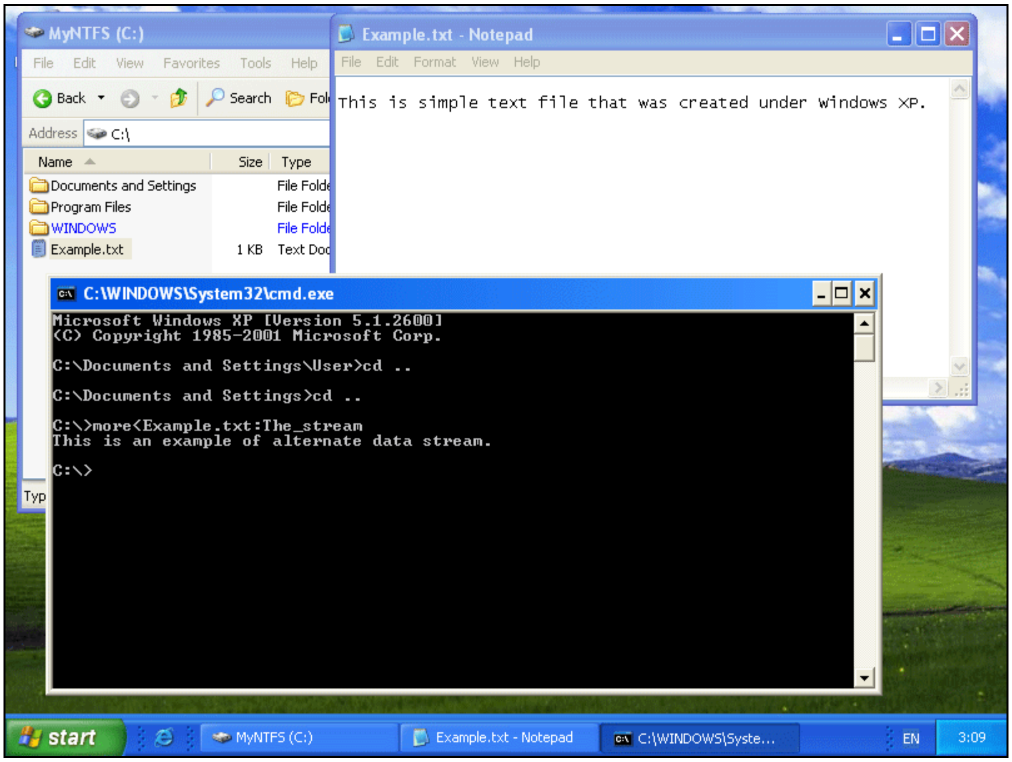
Name	Type	Total Size	Free Space	Comment
Files Stored on This Computer				
Shared Documents	File Folder			
User's Documents	File Folder			
Hard Disk Drives				
MyNTFS (C:)	Local Disk	1,46 GB	851 MB	
Devices with Removable Storage				

Group or user names:

- Administrators (NTFS\Administrators)
- CREATOR OWNER
- Everyone
- SYSTEM
- Users (NTFS\Users)

Permissions for Users	Allow	Deny
Full Control	☐	☐
Modify	☐	☐
Read & Execute	☑	☐
List Folder Contents	☑	☐
Read	☑	☐
Write	☐	☐

The Example.txt file and its alternate data stream:



Address: C:\

Name	Size	Type
Documents and Settings		File Folde
Program Files		File Folde
WINDOWS		File Folde
Example.txt	1 KB	Text Doc

```
This is simple text file that was created under Windows XP.
```

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\User>cd ..

C:\Documents and Settings>cd ..

C:\>more<Example.txt:The_stream
This is an example of alternate data stream.

C:\>
```

Note: These free space redistribution are related to cluster size changes (from 512 to 4096 bytes).

10.5 Results

We have successfully copied all files and folders of the NTFS partition with installed Windows XP to Ext3 file system under Linux. After that we have formatted the NTFS partition and copied all saved files and folders back to the newly formatted NTFS partition. After copying the files and folders back we have successfully booted up Windows XP. All files and folders have the same streams and attributes, including, compression flags, sparse flags and security permissions as before.

Note: If you copy files or folders to a NTFS partition with another Windows OS all security permissions will be set to the default values.

Note: Encrypted files can be read by the same user that has encrypted the file, i.e. has the same account.

Note: The **CPNTFS** utility does not perform low level copying like boot sectors etc. It is designed to copy your data files like documents, images, databases, music etc.

10.6 Conclusion

CPNTFS is a simple one button-click, fast and compact backup/files and folder utility. The **CPNTFS** utility can copy files/folders from selected folders to another location that can be another folder, network drive, zip disk or whatever writeable device you can see under Linux. Even making a backup to a CD-R(W) or DVD-R(W) disk is not a problem. Using this utility you are able to perform full system backup/restore (operating system) on NTFS volume under Linux.