NAME

ntfsclone − Efficiently clone, image, restore or rescue an NTFS

SYNOPSIS

ntfsclone [*OPTIONS*] *SOURCE*
ntfsclone −−save−image [*OPTIONS*] *SOURCE*
ntfsclone −−restore−image [*OPTIONS*] *SOURCE*
ntfsclone −−metadata [*OPTIONS*] *SOURCE*

DESCRIPTION

ntfsclone will efficiently clone (copy, save, backup, restore) or rescue an NTFS filesystem to a sparse file, image, device (partition) or standard output. It works at disk sector level and copies only the used data. Unused disk space becomes zero (cloning to sparse file), encoded with control codes (saving in special image format), left unchanged (cloning to a disk/partition) or filled with zeros (cloning to standard output).

ntfsclone can be useful to make backups, an exact snapshot of an NTFS filesystem and restore it later on, or for developers to test NTFS read/write functionality, troubleshoot/investigate users' issues using the clone without the risk of destroying the original filesystem.

The clone, if not using the special image format, is an exact copy of the original NTFS filesystem from sector to sector thus it can be also mounted just like the original NTFS filesystem. For example if you clone to a file and the kernel has loopback device and NTFS support then the file can be mounted as

mount −t ntfs −o loop ntfsclone.img /mnt/ntfsclone

Windows Cloning

If you want to copy, move or restore a system or boot partition to another computer, or to a different disk or partition (e.g. hda1−>hda2, hda1−>hdb1 or to a different disk sector offset) then you will need to take extra care.

Usually, Windows will not be able to boot, unless you copy, move or restore NTFS to the same partition which starts at the same sector on the same type of disk having the same BIOS legacy cylinder setting as the original partition and disk had.

The ntfsclone utility guarantees to make an exact copy of NTFS but it won't deal with booting issues. This is by design: ntfsclone is a filesystem, not system utility. Its aim is only NTFS cloning, not Windows cloning. Hereby ntfsclone can be used as a very fast and reliable build block for Windows cloning but itself it's not enough.

Sparse Files

A file is sparse if it has unallocated blocks (holes). The reported size of such files are always higher than the disk space consumed by them. The **du** command can tell the real disk space used by a sparse file. The holes are always read as zeros. All major Linux filesystem like, ext2, ext3, reiserfs, Reiser4, JFS and XFS, supports sparse files but for example the ISO 9600 CD−ROM filesystem doesn't.

Handling Large Sparse Files

As of today Linux provides inadequate support for managing (tar, cp, gzip, gunzip, bzip2, bunzip2, cat, etc) large sparse files. The only main Linux filesystem having support for efficient sparse file handling is XFS by the XFS_IOC_GETBMAPX **ioctl**(2)**.** However none of the common utilities supports it. This means when you tar, cp, gzip, bzip2, etc a large sparse file they will always read the entire file, even if you use the "sparse support" options.

bzip2(1) compresses large sparse files much better than **gzip**(1) but it does so also much slower. Moreover neither of them handles large sparse files efficiently during uncompression from disk space usage point of view.

At present the most efficient way, both speed and space−wise, to compress and uncompress large sparse files by common tools would be using **tar**(1) with the options **−S** (handle sparse files "efficiently") and **−j** (filter the archive through bzip2). Although **tar** still reads and analyses the entire file, it doesn't pass on the large data blocks having only zeros to filters and it also avoids writing large amount of zeros to the disk needlessly. But since **tar** can't create an archive from the standard input, you can't do this in−place by just reading **ntfsclone** standard output. Even more sadly, using the −S option results serious data loss since the end of 2004 and the GNU **tar** maintainers didn't release fixed versions until the present day.

The Special Image Format

It's also possible, actually it's recommended, to save an NTFS filesystem to a special image format. Instead of representing unallocated blocks as holes, they are encoded using control codes. Thus, the image saves space without requiring sparse file support. The image format is ideal for streaming filesystem images over the network and similar, and can be used as a replacement for Ghost or Partition Image if it is combined with other tools. The downside is that you can't mount the image directly, you need to restore it first.

To save an image using the special image format, use the **−s** or the **−−save−image** option. To restore an image, use the **−r** or the **−−restore−image** option. Note that you can restore images from standard input by using '−' as the *SOURCE* file.

Metadata−only Cloning

One of the features of **ntfsclone** is that, it can also save only the NTFS metadata using the option **−m** or **−−metadata** and the clone still will be mountable. In this case all non−metadata file content will be lost and reading them back will result always zeros.

The metadata−only image can be compressed very well, usually to not more than 1−8 MB thus it's easy to transfer for investigation, troubleshooting.

In this mode of ntfsclone, **NONE** of the user's data is saved, including the resident user's data embedded into metadata. All is filled with zeros. Moreover all the file timestamps, deleted and unused spaces inside the metadata are filled with zeros. Thus this mode is inappropriate for example for forensic analyses. This mode may be combined with **−−save−image** to create a special image format file instead of a sparse file.

Please note, filenames are not wiped out. They might contain sensitive information, so think twice before sending such an image to anybody.

OPTIONS

Below is a summary of all the options that **ntfsclone** accepts. Nearly all options have two equivalent names. The short name is preceded by **−** and the long name is preceded by **−− .** Any single letter options, that don't take an argument, can be combined into a single command, e.g. **−fv** is equivalent to **−f −v .** Long named options can be abbreviated to any unique prefix of their name.

−o, **−−output** FILE

Clone NTFS to the non−existent *FILE*. If *FILE* is '−' then clone to the standard output. This option cannot be used for creating a partition, use **−−overwrite** for an existing partition.

−O, **−−overwrite** FILE

Clone NTFS to *FILE*, which can be an existing partition or a regular file which will be overwritten if it exists.

−s, **−−save−image**

Save to the special image format. This is the most efficient way space and speed−wise if imaging is done to the standard output, e.g. for image compression, encryption or streaming through a network.

−r, **−−restore−image**

Restore from the special image format specified by *SOURCE* argument. If the *SOURCE* is '−' then the image is read from the standard input.

−n, **−−no−action**
Test the consistency of a saved image by simulating its restoring without writing anything. The NTFS data contained in the image is not tested. The option **−−restore−image** must also be present, and the options **−−output** and **−−overwrite** must be omitted.

−−rescue
Ignore disk read errors so disks having bad sectors, e.g. dying disks, can be rescued the most efficiently way, with minimal stress on them. Ntfsclone works at the lowest, sector level in this mode too thus more data can be rescued. The contents of the unreadable sectors are filled by character '?' and the beginning of such sectors are marked by "BadSectoR\0".

−m, **−−metadata**
Clone **ONLY METADATA** (for NTFS experts). Only cloning to a (sparse) file is allowed, unless used the option **−−save−image** is also used. You can't metadata−only clone to a device.

−−ignore−fs−check
Ignore the result of the filesystem consistency check. This option is allowed to be used only with the **−−metadata** option, for the safety of user's data. The clusters which cause the inconsistency are saved too.

−t, **−−preserve−timestamps**
Do not wipe the timestamps, to be used only with the **−−metadata** option.

−−full−logfile
Include the Windows log file in the copy. This is only useful for extracting metadata, saving or cloning a file system which was not properly unmounted from Windows.

−−new−serial, or

−−new−half−serial
Set a new random serial number to the clone. The serial number is a 64 bit number used to identify the device during the mounting process, so it has to be changed to enable the original file system and the clone to be mounted at the same time on the same computer.

The option **−−new−half−serial** only changes the upper part of the serial number, keeping the lower part which is used by Windows unchanged.

The options **−−new−serial** and **−−new−half−serial** can only be used when cloning a file system of restoring from an image.

The serial number is not the volume UUID used by Windows to locate files which have been moved to another volume.

−f, **−−force**
Forces ntfsclone to proceed if the filesystem is marked "dirty" for consistency check.

−q, **−−quiet**
Do not display any progress-bars during operation.

−h, **−−help**
Show a list of options with a brief description of each one.

EXIT CODES

The exit code is 0 on success, non−zero otherwise.

EXAMPLES

Clone NTFS on /dev/hda1 to /dev/hdc1:

ntfsclone −−overwrite /dev/hdc1 /dev/hda1

Save an NTFS to a file in the special image format:

ntfsclone −−save−image −−output backup.img /dev/hda1

Restore an NTFS from a special image file to its original partition:

ntfsclone −−restore−image −−overwrite /dev/hda1 backup.img

Save an NTFS into a compressed image file:

ntfsclone −−save−image −o − /dev/hda1 | gzip −c > backup.img.gz

Restore an NTFS volume from a compressed image file:

**gunzip −c backup.img.gz | **
ntfsclone −−restore−image −−overwrite /dev/hda1 −

Backup an NTFS volume to a remote host, using ssh. Please note, that ssh may ask for a password!

**ntfsclone −−save−image −−output − /dev/hda1 | **
gzip −c | ssh host 'cat > backup.img.gz'

Restore an NTFS volume from a remote host via ssh. Please note, that ssh may ask for a password!

**ssh host 'cat backup.img.gz' | gunzip −c | **
ntfsclone −−restore−image −−overwrite /dev/hda1 −

Stream an image file from a web server and restore it to a partition:

**wget −qO − http://server/backup.img | **
ntfsclone −−restore−image −−overwrite /dev/hda1 −

Clone an NTFS volume to a non−existent file:

ntfsclone −−output ntfs−clone.img /dev/hda1

Pack NTFS metadata for NTFS experts. Please note that bzip2 runs very long but results usually at least 10 times smaller archives than gzip on a sparse file.

ntfsclone −−metadata −−output ntfsmeta.img /dev/hda1
bzip2 ntfsmeta.img

Or, outputting to a compressed image :
ntfsclone −mst −−output - /dev/hda1 | bzip2 > ntfsmeta.bz2

Unpacking NTFS metadata into a sparse file:

**bunzip2 −c ntfsmeta.img.bz2 | **
cp −−sparse=always /proc/self/fd/0 ntfsmeta.img

KNOWN ISSUES

There are no known problems with **ntfsclone**. If you think you have found a problem then please send an email describing it to the development team: ntfs−3g−devel@lists.sf.net

Sometimes it might appear ntfsclone froze if the clone is on ReiserFS and even CTRL−C won't stop it. This is not a bug in ntfsclone, however it's due to ReiserFS being extremely inefficient creating large sparse files

and not handling signals during this operation. This ReiserFS problem was improved in kernel 2.4.22. XFS, JFS and ext3 don't have this problem.

AUTHORS

ntfsclone was written by Szabolcs Szakacsits with contributions from Per Olofsson (special image format support) and Anton Altaparmakov. It was ported to ntfs-3g by Erik Larsson and Jean-Pierre Andre.

AVAILABILITY

ntfsclone is part of the **ntfs-3g** package and is available at:
http://www.tuxera.com/community/

SEE ALSO

ntfsresize(8) **ntfsprogs**(8) **xfs_copy**(8) **debugreiserfs**(8) **e2image**(8)