ask Ubuntu

How do I use 'chmod' on an NTFS (or FAT32) partition?

Asked 13 years, 5 months ago Modified 1 year, 2 months ago Viewed 313k times

165

I have a script that I need to execute on an NTFS partition. The script's permission is set to 600.

I attempted to modify the permissions by running `chmod 755 script.sh`, which doesn't report a failure or anything - but it also doesn't change the permissions on the file:

```
$ stat script.sh

  File: `script.sh'
  Size: 297070      Blocks: 584        IO Block: 4096   regular file
Device: 811h/2065d  Inode: 35515       Links: 1
Access: (0600/-rw-------)  Uid: ( 1000/  xxxxxx)   Gid: ( 1000/  xxxxxx)
Access: 2010-09-30 14:05:16.041621000 -0700
Modify: 2010-09-30 14:05:05.070157000 -0700
Change: 2010-09-30 14:05:05.070475000 -0700

$ chmod 755 script.sh
$ stat script.sh

  File: `script.sh'
  Size: 297070      Blocks: 584        IO Block: 4096   regular file
Device: 811h/2065d  Inode: 35515       Links: 1
Access: (0600/-rw-------)  Uid: ( 1000/  xxxxxx)   Gid: ( 1000/  xxxxxx)
Access: 2010-09-30 14:05:16.041621000 -0700
Modify: 2010-09-30 14:05:05.070157000 -0700
Change: 2010-09-30 14:05:05.070475000 -0700
```

As you can see, it remains unchanged.

permissions ntfs chmod

Share Improve this question Follow

edited Feb 11, 2014 at 11:37
kiri
28.2k 16 81 118

asked Nov 6, 2010 at 23:12
Nathan Osman
32.2k 40 179 259

A better solution can be found here This two questions should be linked! – user321419 Aug 28, 2014 at 20:39

10 Answers

Sorted by: Highest score (default)

113

Contrary to what most people believe, NTFS is a POSIX-compatible[1] filesystem, and **it is possible to use permissions on NTFS**.

To enable this, you need a "User Mapping File" or just give the `permissions` option when mounting (when no compatibility with Windows is needed). This maps linux users on your system with the user IDs like NTFS/Windows use them internally.

See the ntfs-3g manpage for some info and some examples. If you need more information, see the ntfs-3g advanced documentation about ownership and permissions.

(Note that this does not work on FAT filesystems.)

[1] Yes, it can also store filenames that are valid in linux/unix but not under Windows, supports symlinks & hardlinks, etc.

Share Improve this answer Follow

edited Aug 29, 2020 at 16:38
geekley
549 5 21

answered Nov 2, 2011 at 16:57
JanC
19.4k 4 44 50

6 here is good documentation. in short: `sudo ntfs-3g.usermap /dev/disk/by-label/MY-NTFS` and then `sudo mv UserMapping /media/MY-NTFS/.NTFS-3G/` – flying sheep Jan 6, 2013 at 23:40

1 So this will allow you to arbitrarily set permissions like `chmod 655 /some/file` on the NTFS partition mounted in Linux? I'm trying to figure out how to merge my home partition from linux into c:\Users. Will using usermap allow me to retain all the permissions? I was planning on mounting the c:\Users directory to

/home in linux. – trusktr Feb 13, 2014 at 7:22

11 Let me re-emphasize your remark: "when no compatibility with Windows is needed". ref: askubuntu.com/questions/92863/… – Elliptical view Sep 13, 2016 at 1:15

1 `ntfs-3g manpage` link broken – ctrl-alt-delor Sep 29, 2016 at 20:17

2 Eduardo Cuomo wrote a pretty good example of all steps necessary to make a "User Mapping File" here: askubuntu.com/a/887502/327339 – Gabriel Staples Apr 5, 2017 at 2:54

93

The mode is determined by the partition's mount options (you cannot change it via chmod).

For '755' on files and '777' on directories you would use something like

```
sudo mount -t ntfs -o rw,auto,user,fmask=0022,dmask=0000 /dev/whatever /mnt/whatever
```

Share Improve this answer Follow

edited Nov 27, 2013 at 3:49

Seth

58.1k ● 44 ● 147 ● 201

answered Nov 6, 2010 at 23:35

htorque

64.8k ● 41 ● 195 ● 219

55

For NTFS partitions, use the `permissions` option in fstab.

First unmount the ntfs partition.

Identify your partition UUID with `blkid`

```
sudo blkid
```

Then edit `/etc/fstab`

```
# Graphical
gksu gedit /etc/fstab

# Command line
sudo -e /etc/fstab
```

And add or edit a line for the ntfs partition

```
# change the "UUID" to your partition UUID
UUID=12102C02102CEB83 /media/windows ntfs-3g auto,users,permissions 0 0
```

Make a mount point (if needed)

```
sudo mkdir /media/windows
```

Now mount the partition

```
mount /media/windows
```

The options I gave you, `auto`, will automatically mount the partition when you boot and `users` allows users to mount and umount .

You can then use chown and chmod on the ntfs partition.

Share Improve this answer Follow

edited Aug 28, 2017 at 1:47

muru

198k ● 55 ● 485 ● 741

answered Dec 28, 2011 at 16:31

Panther

102k ● 19 ● 197 ● 286

20

In addition to setting the fmask and/or dmask in htorque's answer above, if you want to execute scripts on the drive, I had to also set the "exec" mount option.

So the example would be:

```
sudo mount -t ntfs -o rw,auto,user,fmask=0022,dmask=0000,exec /dev/whatever /mnt/whatever
```

Share Improve this answer Follow

answered Dec 14, 2010 at 8:26

dbrews

301 ● 1 ● 3

`exec` option controls whether partition can execute binaries, so it works for me – lucidyan Apr 5, 2023 at 17:12

16

You can always explicitly invoke the script interpreter, in which case execution permissions are not required. If the script uses *bash*, as can be verified by looking at the first line of the script, just run

```
bash script.sh
```

Note that the script calls other scripts or binaries on the same partition, this won't work. Note also that the strategy doesn't work with binaries as opposed to textual script files written in Bash Script, Perl, Python or the like.

Share Improve this answer Follow

answered Dec 14, 2010 at 10:00

loevborg

7,282 ● 1 ● 27 ● 24

10 To execute binaries, use `/lib64/ld-linux-x86-64.so.2 ./program.bin` for 64-bit programs and `/lib/ld-linux.so.2 ./program.bin` for 32-bit ones. – Lekensteyn Apr 13, 2012 at 19:52

16

Check that Fast Startup is turned off

If Windows uses Fast Startup, it is semi-hibernated, and its file system is 'dirty', and Linux mounts it read-only to avoid causing damage. Either **reboot Windows** (instead of shutdown) or **turn off Fast Startup** (a setting in Windows), and Linux is willing to mount the Windows file system with write access.

Mount NTFS partition in a USB drive with custom permissions and owner

In Linux the mode of NTFS (and FAT32 and exFAT) is determined by the partition's *mount options*. You cannot change it via chmod.

Assumption: the USB drive is seen as `sdb1`, *modify to match the drive letter and partition number in your case*. The general syntax is `sdxn`, where `x` is the drive letter and `n` is the partition number as seen by for example `sudo lsblk -f`

Preparing

- Unmount the NTFS partition.

```
sudo umount /dev/sdxn   # general syntax
sudo umount /dev/sdb1   # modify to match your case
```

- Create a custom mountpoint (only if you want a new mountpoint), for example with

```
sudo mkdir -p /mnt/sd1
```

- Check your userID's `uid` number (it is usually 1000, sometimes 1001 or 1002 ...)

```
grep ^"$USER" /etc/group
```

and use that number if you want to grab ownership (default is `root`).

Mount the NTFS partition

Example 1 (without execute permissions for files, no access for 'others'),

```
sudo mount -o rw,user,uid=1000,dmask=007,fmask=117 /dev/sdxn /mnt/sd1  # general syntax
sudo mount -o rw,user,uid=1000,dmask=007,fmask=117 /dev/sdb1 /mnt/sd1  # modify to match your case
```

- in this case you can run the script `this-script` with

```
bash /mnt/sd1/this-script
```

Example 2 (with execute permissions for files, no access for 'others'),

```
sudo mount -o rw,user,uid=1000,umask=007,exec /dev/sdxn /mnt/sd1  # general syntax
sudo mount -o rw,user,uid=1000,umask=007,exec /dev/sdb1 /mnt/sd1  # modify to match your case
```

- In this case you can run the script `this-script` with

  ```
  /mnt/sd1/this-script
  ```

and you can run executable programs too from this location (not that it is recommended).

Example 3 (full permissions for everybody, which is convenient but not safe, when there are several users),

```
sudo mount -o rw,users,umask=000,exec /dev/sdxn /mnt/sd1  # general
sudo mount -o rw,users,umask=000,exec /dev/sdb1 /mnt/sd1  # modify to match your case
```

Share Improve this answer Follow

edited Nov 24, 2021 at 8:37

answered Sep 15, 2017 at 11:02

sudodus

46.3k ● 5 ● 88 ● 152

`/media$ sudo mkdir -p sdb1` `/media$ sudo mount -o rw,users,umask=000,exec /dev/sdb1 ./sdb1/ mount: block device /dev/sdb1 is write-protected, mounting read-only` – alhelal Nov 29, 2017 at 16:02

@alhelal, I am afraid that the hardware of your USB drive has become read-only or 'grid-locked'. But there might also be some problem with the file system, and if the file system is corrupted, you might be able to fix it by repairing it in Windows, either with the GUI method or with the command line `chkdsk /f X:` according to this link ubuntuforums.org/… -- If still no luck, backup the data and try according to askubuntu.com/questions/144852/… – sudodus Nov 29, 2017 at 17:21

`You cannot change it via chmod.` This is blatantly wrong. You CAN chmod/chown on an NTFS partition, if you use a usermap file. Just `man ntfsusermap` and you will see how. I've been chmodding NTFS partitions since ages. You could also read the NTFS-3G FAQs on how to do this: github.com/tuxera/ntfs-3g/wiki/… – shivams Jul 1, 2022 at 0:20

Thanks for that link, @shivams. – sudodus Jul 1, 2022 at 6:55

1 Today I realized that I have subscribed AU (4 years ago!) so I can finally give a follow-up to this comment! – gboffi Mar 6, 2023 at 10:59

12

All steps:

1. Install `ntfs-3g`:

```
sudo apt-get install -y ntfs-3g
```

2. Unmount *NTFS* partition:

```
sudo umount /mnt/windows
```

3. Use `ntfsusermap` (ex `ntfs-3g.usermap`) to generate your `UserMapping` file:

```
#sudo ntfs-3g.usermap /dev/disk/by-label/MY-NTFS
sudo ntfsusermap /dev/disk/by-label/MY-NTFS

# or
```

```
#sudo ntfs-3g.usermap /dev/sdb1
sudo ntfsusermap /dev/sdb1
```

4. Remount *NTFS* partition to add `UserMapping` file:

```
mount -a
sudo mkdir /mnt/windows/.NTFS-3G
sudo mv UserMapping /mnt/windows/.NTFS-3G/
```

5. Update your `fstab` file:

```
sudo vim /etc/fstab
```

Update *mount* line:

1. Backup your current mount line! Duplicate the line and comment it by adding a `#` at the beginning.
2. Change next: `UUID=34A0456DA04536A0 /mnt/windows ntfs defaults,uid=1000,gid=1000 0 0`
3. By next: `UUID=34A0456DA04536A0 /mnt/windows ntfs-3g defaults 0 0` (Use `ntfs-3g` and only `default` option)

It should look something like this:

```
#UUID=34A0456DA04536A0 /mnt/windows ntfs defaults,uid=1000,gid=1000 0 0
UUID=34A0456DA04536A0 /mnt/windows ntfs-3g defaults 0 0
```

6. Finally, remount using your `fstab`:

```
sudo umount /mnt/windows
sudo mount -a
```

Do this once for every *NTFS* partition you have!

WARNING WITH WINDOWS OS!

I check it with **Windows 7+** and the permissions affect Windows OS! I change the permissions of my Home Directory on Windows partition, and when I used Windows again I could see that the user was broken!

Share Improve this answer Follow

edited Feb 6, 2023 at 13:42

answered Feb 26, 2017 at 14:49

Eduardo Cuomo

332 ● 3 ● 10

8 `ntfs-3g.usermap` has been renamed to `ntfsusermap` on recent distributions -- see: askubuntu.com/a/996255/605262 – Eric Reed Jun 6, 2020 at 21:24

11

According to the Ownership and Permissions section of the NTFS-3G documentation, we can use mount options to control file *access* and *creation*. The combinations are very complicated (see the two tables there). Also I do not read and get all of them. For example, I do not know whether POSIX ACLs is selected at compile-time or not of the NTFS-3G binary package. But the best I have come out is using a User Mapping file combined with some mount options to approximate a plausible mapping of file ownership and permissions between Windows and Linux.

Warning: This is only what works best for my sharing a NTFS *data partition* (drive `D:` on Windows) between dual-booted Windows 8 and Kubuntu 14.04. The instructions are recorded in careful retrospection but not thoroughly tested. It is too tiring and tedious to repeat the whole procedure again. So follow it at your own risk. But if you do, share back your experience. If you decide to follow the instructions, please read it fully to have a whole picture before actually acting. Good luck!

Alright, here you go! The detailed instructions consist of three parts. Part 1 should be carried out on Windows while Part 2 on Linux. Part 3 is for test.

Part 1

The User Mapping section of the NTFS-3G documentation specifies two versions to set up user mapping between Windows and Linux, one Windows version and one Linux version. My experience was that the Linux version ended up with a *miss*. The Linux account was *not* mapped to my Windows account but some *unknown* account appeared under an SID. The result was a mess since this unknown account takes

ownership of all files of my Windows account. In that situation, unless you have an administrative privilege to take your ownership back, files under your Windows account become inaccessible. But even if you mange, it is *still* a wrong mapping. That means, later whatever files you create on Linux get assigned to that unknown account on Windows and those on Windows get assigned to root on Linux (if I remember correctly). So on Windows you need to take ownership back again and on Linux change ownership. That is not what we expect it to be. After several hopeless attempts to fix the issue, I gave up and turned to the Windows version. That one worked. Detailed instructions extracted from the relevant section of the NTFS-3G documentation follow:

1. Download the usermap tool, extract it somewhere (in my case, drive `C:`), better outside the NTFS partition (in my case drive `D:`) to be shared.
2. Open the Windows command line. Change to the extracted directory `tools` (by default) of the `usermap` tool. Then run the following command:

   ```
   C:\tools> mapuser > UserMapping
   ```

   This generates a template and redirects it to a file named `UserMapping`. Open the file with a text editor, say Notepad, you should see the following lines:

   ```
   # Generated by usermap for Windows, v 1.1.5
   # For Windows account "Account" in domain "Domain"
   # Replace "user" and "group" hereafter by matching Linux login
   user::SID
   :group:SID
   ```

   Presumably, the first `SID` should be your user SID while the second your group SID. You can check them respectively by commands `whoami /user` and `whoami /groups`.
3. After you make sure the SIDs are correct, following the instructions in the comment, that is, change `user` in the `user::SID` line to your user name and `group` in the `:group:SID` line to your primary group name on Linux. On Ubuntu, they are the same. Moreover, add your Linux group name also after the first colon of the `user::SID` line. So the line should look something like `user:group:SID`. It seems that if not doing so files created on Windows will be assigned to `user:root` on Linux.
4. Save the file. Move it to a directory named `.NTFS-3G` (create it if not existent yet) on the NTFS partition to be shared (in my case drive `D:`).
5. This step is for test in Part 3. On the shared NTFS partition, create a new directory and a new file.

Part 2

Now boot into Linux. `sudo` edit the file `/etc/fstab`. Add or modify the line for the shared NTFS partition to something like the following:

```
UUID=...    /data    ntfs    defaults,umask=077,utf8    0    0
```

The essential is to set the `umask` (`dmask` and `fmask` may also work but not tested). Pick a value for `umask` you like, although I picked `077`. It seems without this setting, full permissions will be given to `o`thers for newly-created files.

Save the file. Now `sudo mount` or remount (`sudo umount` and then `sudo mount`) the shared NTFS partition (in my case `/data`):

```
$ sudo mount /data
```

Part 3

Now (still on Linux) `cd` to the mount point (in my case, `/data`), `ls -l` the files there. Check whether their ownership and permissions match respectively that you specified in the `UserMapping` file and the `umask` you set in `/etc/fstab` (the match between permissions and `umask` requires some complement calculation, see man (1) umask for more information). If they do, congratulations, half goal is achieved. Otherwise, poor you. Ask Ubuntu or Windows.

Then create a new directory and a new file. `ls -l` to check their ownership and permissions. The ownership should be your user name and primary group as usual. The permissions should match the `umask`. Now restart your computer and boot into Windows. Locate on the shared NTFS partition the directory and file you just created on Linux. Check their properties to see if they are assigned to your Windows account. If they are, congratulations, you are all done. Otherwise, bad luck. Ask Windows or Ubuntu.

EOF

Share Improve this answer Follow

edited Sep 14, 2017 at 18:50

Fabian N.
95 ● 11

answered Aug 22, 2014 at 22:35

reflectionalist
111 ● 1 ● 4

8

+50

Old thread, I know, but still relevant and missing a particular use case tip, composed from different suggestions on various other forums/threads and tested on Ubuntu GNOME 13.04 where I wanted an external drive to hold a Steam library...

When the NTFS partition is on an external usb drive, for example -- which means the partition is mounted on the fly upon connection -- then you can use the following method to make udev mount ntfs partitions with execution rights.

Open a terminal window and do:

```
$ sudo nano /etc/udev/rules.d/90-usb-disks.rules
```

Then paste this line in what should be a blank/new file (if not, then exit nano and reissue the command but starting the file name with a higher number like 91-...):

```
ENV{ID_FS_TYPE}=="ntfs", ENV{ID_FS_TYPE}="ntfs-3g"
```

Then save and close. Unplug the drive and then do in terminal:

```
$ sudo service udev restart
```

Next, plug the drive back in and enjoy :)

Share Improve this answer Follow

answered Sep 14, 2013 at 23:51

user17254

613 ● 5 ● 10

1

There is n related question for USB devices. This answer provides an ugly hack if you want to mount every USB device automatically with execute permissions.

Share Improve this answer Follow

edited Apr 13, 2017 at 12:25

answered Dec 28, 2011 at 15:06

Community Bot
1

lumbric
3,994 ● 3 ● 34 ● 48

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.