

Ubuntu Chmod and Chown Commands Cheat Sheet

Get more Linux command cheat sheets at [UbuntuFree.com](https://ubuntufree.com)

Full guide with explanations: www.ubuntufree.com/chmod-chown-commands-in-ubuntu-with-examples/

Updated for Ubuntu 26.04 LTS (2026)

chmod changes what users can do with a file, and chown changes who owns it. Both work the same on Ubuntu 26.04 LTS and Ubuntu 26.10. This guide explains how to read permissions, use numeric and symbolic modes, change owners and groups, and handle special bits, with copy-ready examples and a free cheat sheet.

Read permissions first

Every file has an owner, a group and three sets of permissions: for the owner (u), the group (g) and everyone else (o).

1. List permissions

The first column shows the type and permissions, then the owner and group.

```
ls -l
ls -ld ~/projects
```

2. Show numeric permissions

stat prints the mode as letters and as an octal number.

```
stat -c '%A %a %U:%G %n' notes.txt
```

3. Check your user and groups

Group membership decides which group permissions apply to you.

```
id
groups
```

Chmod examples

chmod accepts symbolic modes such as u+x and numeric modes such as 755. Symbolic modes change only what you name. Numeric modes set every bit at once.

4. Make a script executable

Adds execute permission. u+x limits it to the owner.

```
chmod +x deploy.sh
chmod u+x deploy.sh
```

5. Add and remove with symbolic modes

Give the group write access and remove read access for others in one command.

```
chmod g+w,o-r notes.txt
```

6. Set a file to 644

Owner can read and write. Everyone else can only read. The usual mode for documents and web files.

```
chmod 644 notes.txt
```

7. Set a folder or script to 755

Owner has full access. Others can read and enter or run it.

```
chmod 755 deploy.sh
chmod 755 ~/public
```

8. Protect private keys

SSH refuses keys that other users can read.

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh/id_ed25519
```

10. Folders 755, files 644

find applies a different mode to each type, which is the clean way to fix a messy tree.

```
find project -type d -exec chmod 755 {} +
find project -type f -exec chmod 644 {} +
```

12. Show what changed

-c reports only files whose mode changed. -v reports every file.

```
chmod -c 644 *.txt
```

Chown and chgrp examples

Only root can give a file to another user, so chown almost always needs sudo.

13. Change the owner

Gives the file to the user alice.

```
sudo chown alice notes.txt
```

15. Change only the group

chgrp, or chown with a leading colon, changes just the group.

```
sudo chgrp developers notes.txt
sudo chown :developers notes.txt
```

17. Find root-owned files in your home

Lists files that will give you Permission denied errors.

```
find ~ -user root
```

19. Give a web server access

Ubuntu web servers run as www-data. Group ownership plus g+rwX lets you and the server both write, without making every file executable.

```
sudo chown -R "$USER":www-data /var/www/html
sudo chmod -R g+rwX /var/www/html
```

9. Change a tree recursively

Capital X adds execute only to folders and to files that are already executable.

```
chmod -R u+rwX,go+rX,go-w project/
```

11. Copy permissions from another file

Matches the mode of a reference file.

```
chmod --reference=good.txt bad.txt
```

14. Change owner and group

Sets both at once with user:group.

```
sudo chown alice:developers notes.txt
```

16. Take ownership of a folder tree

Fixes files created with sudo inside your home folder.

```
sudo chown -R "$USER":"$USER" ~/projects
```

18. Change only matching owners

--from changes ownership only if the current owner matches.

```
sudo chown --from=root:root "$USER":"$USER"
report.pdf
```

20. Add a user to a group

Group changes apply after the user logs out and back in.

```
sudo usermod -aG developers alice
```

Special bits, umask and ACLs

Special bits add a fourth digit in front of the normal mode, for example 2775.

21. Setgid on a shared folder

New files inherit the folder group, which keeps team folders consistent.

```
sudo chmod 2775 /srv/shared
sudo chmod g+s /srv/shared
```

22. Sticky bit

Users can only delete their own files in the folder, as in /tmp.

```
sudo chmod +t /srv/shared
ls -ld /tmp
```

23. Find setuid programs

Setuid programs run with the owner rights. Review this list when auditing a system.

```
find /usr/bin -perm -4000 -type f
```

24. Check and set the umask

umask removes bits from new files. The Ubuntu default 002 gives 664 files and 775 folders.

```
umask
umask 027
```

25. Grant one extra user with an ACL

ACLs add per-user rules on top of normal permissions. Install acl if setfacl is missing.

```
setfacl -m u:alice:rw report.txt
getfacl report.txt
```

Permission numbers quick reference

Mode	Letters	Typical use
777	rw-rw-rwx	Avoid. Anyone can change or delete
775	rw-rw-r-x	Shared team folders
755	rw-r-x-r-x	Folders, scripts and programs
700	rw-x-----	Private folders such as ~/.ssh
664	rw-rw-r--	Shared team files
644	rw-r--r--	Documents and web files
600	rw-----	Private keys and secrets
400	r-----	Read-only secrets

Tips and best practices

If something says Permission denied, check three things in order: who you are (id), who owns the file (ls -l), and whether every parent folder lets you in (ls -ld on each folder).

Avoid chmod 777 as a quick fix. It lets every user and every service change the file. Fix the owner or group instead.