

Linux Administration

File permissions

Xavier Belanger

**This work is licensed under
a Creative Commons Attribution-ShareAlike 4.0 International License.**

<http://creativecommons.org/licenses/by-sa/4.0/>

You are free to:

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

- **Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
- **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

File permissions

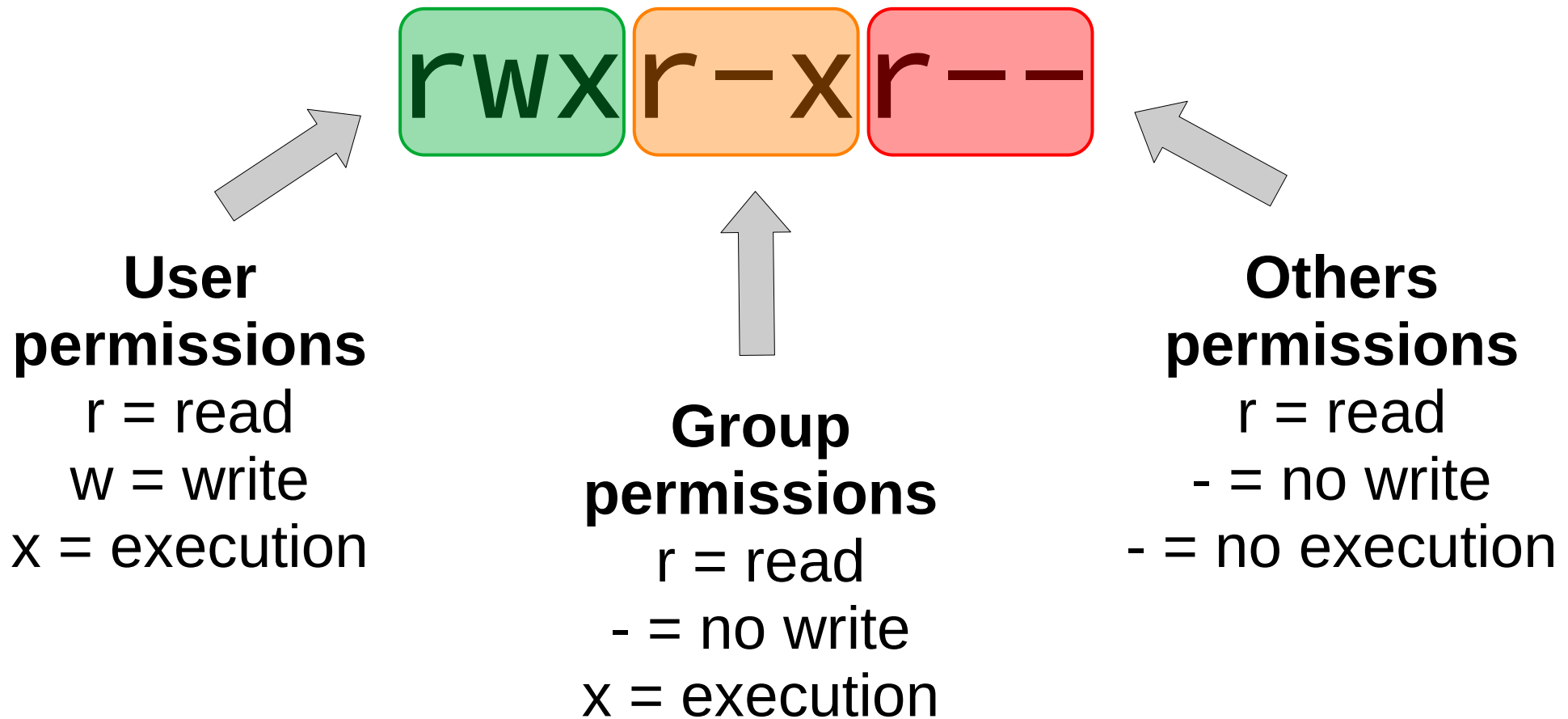
- There is three main permissions that could be set on a file or a directory:
 - read
 - write
 - execute ¹
- For each file or directory, permissions are defined for the user, the group and others (those who are not part the two first entities).

1: this permission is also used to grant or limit access into a directory

Checking permissions

- Permissions on files and directories are displayed with the `ls -l` command.
- You can also use the `stat` command against a file, and read the “Access” line.

File permissions details



File permissions: symbolic mode

- The `chmod` command can be used to define permissions for each entity:
 - `chmod u+rw <file>` grant read and write permissions to the user
 - `chmod g-w <file>` revoke the write permission for the group
 - `chmod o-r <file>` revoke the read permission for other users (not the user, not the group)

File permissions: octal mode

- In octal mode, each permission is given a value
 - read = 4
 - write = 2
 - execute = 1
 - no permission = 0
- Then you need to set the values for all three entities:
 - *chmod 644 <file>* = user (rw-) group (r--) others (r--)
 - *chmod 700 <file>* = user (rwx) group (---) others (---)

File ownership

- In order to change the user ownership and group ownership of a file, you can use the `chown` and `chgrp` commands:
 - `chown <user> <file>`
 - `chgrp <group> <file>`
- You can change the owner and group in one command: `chown <user>:<group> <file>`

Special permissions: setuid and setgid

- The setuid permission will allow a command or a script to run under the identity of the user.
- Use the *chmod u+s <file>* command to define this permission.
- The setgid permission work in a similar way based on the group.
- This can be set with the *chmod g+s <file>* command.

setgid on directories

- When used on a directory, the setgid permission will allow ownership inheritance for any new content in that directory.

Special permission: sticky bit

- The sticky bit is a protection limiting a file to be deleted only by the file owner, the directory owner or root.
- Use *chmod +t <directory>* to set that permission.