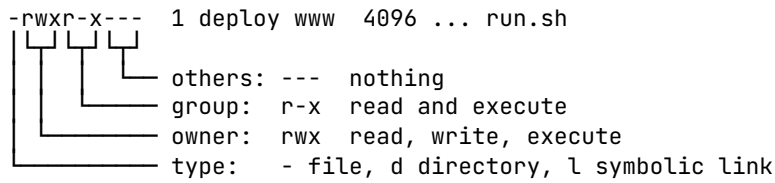
$ cd reference / File permissions (chmod)

File permissions with chmod: octal, symbolic, special bits

Every file has three sets of three bits: for the owner, for the group and for everyone else. chmod sets them either as an octal number or with symbolic changes.

[01] Reading ls -l



DIGIT	BITS	SYMBOLIC	MEANING
0	000	---	no access
1	001	--x	execute only
2	010	-w-	write only
3	011	-wx	write and execute
4	100	r--	read only
5	101	r-x	read and execute
6	110	rw-	read and write
7	111	rwx	everything

Each octal digit is the sum of read 4, write 2 and execute 1. The three digits are owner, group and others, in that order.

[02] What the bits mean on a directory

BIT	ON A FILE	ON A DIRECTORY
r	read the contents	list the names in it
w	change the contents	together with x: create, delete and rename entries in it, whatever the files' own modes
x	run it as a program	enter it and reach anything inside by name

Deleting a file needs write permission on the directory, not on the file. A directory with `r` but no `x` lets you list names but not open anything in it. With `x` but no `r` you can open files whose names you already know.

[03] Common modes

MODE	SYMBOLIC	TYPICAL USE
644	rw-r--r--	ordinary files, web content
600	rw-------	private keys, files with passwords; ssh refuses keys that are more open
640	rw-r-----	config readable by a service group
755	rwxr-xr-x	directories, scripts and binaries everyone may run
750	rwxr-x---	directories for one group
700	rwx------	~/.ssh and other private directories
775	rwxrwxr-x	shared project directories, usually with setgid
400	r--------	read-only secrets
777	rwxrwxrwx	almost never right; use group ownership instead

[04] Symbolic form

TASK	COMMAND
Make a script executable for its owner	`chmod u+x deploy.sh`
Remove write for group and others	`chmod go-w file`
Set others to nothing, exactly	`chmod o= file`
Read for everyone	`chmod a+r file`
Several changes at once	`chmod u=rwx,g=rx,o= dir`
Directories 755, files 644 (755 if already executable), in one tree	`chmod -R u=rwX,go=rX dir`
Copy the owner's bits to the group	`chmod g=u file`
Show modes in octal	`stat -c "%a %n" *`

Capital `X` adds execute only to directories and to files that already have execute for someone. It is the safe way to fix a whole tree, because `chmod -R 755` would make every file executable.

[05] Special bits

BIT	OCTAL	ON A FILE	ON A DIRECTORY	IN LS -L
setuid	`4000`	runs with the owner's privileges, as `passwd` does	ignored on Linux	`rws` in the owner triple
setgid	`2000`	runs with the group's privileges	new files inherit the directory's group	`rws` in the group triple
sticky	`1000`	ignored on Linux	only the owner of the file, the owner of the directory or root may delete or rename it, as in `/tmp`	`rwt` in the others triple

The special bits form a fourth leading digit: `chmod 2775 shared/`, `chmod 1777 /tmp`. A capital `S` or `T` in `ls -l` means the special bit is set but the execute bit under it is not, which is usually a mistake. The kernel ignores setuid on scripts.

[06] umask: the default for new files

New files start from 666 and new directories from 777, and every bit set in the umask is cleared. The common umask `022` gives files 644 and directories 755. A umask of `027` gives 640 and 750, and `077` gives 600 and 700.

TASK	COMMAND
Show the current umask	`umask`
Show it symbolically	`umask -S`
Set it for this shell	`umask 027`

A service's umask comes from its unit or init script, not from your shell. In systemd, set `UMask=0027` in the unit.

[07] Ownership and ACLs

TASK	COMMAND
Change owner and group	`sudo chown deploy:www file`
Change recursively	`sudo chown -R deploy:www /srv/app`
Change only the group	`chgrp www file`
Show ACL entries	`getfacl file`

TASK	COMMAND
Give one more user read access	`setfacl -m u:backup:r file`

A `+` after the mode in `ls -l` (`-rw-r-----+`) means an ACL adds rules that the nine bits do not show.

[08] Sources

[1] GNU coreutils manual: file permissions and modes

[2] chmod(1)

[3] umask(2)

[4] inode(7): set-user-ID, set-group-ID and sticky bits

[5] systemd.exec(5): UMask=