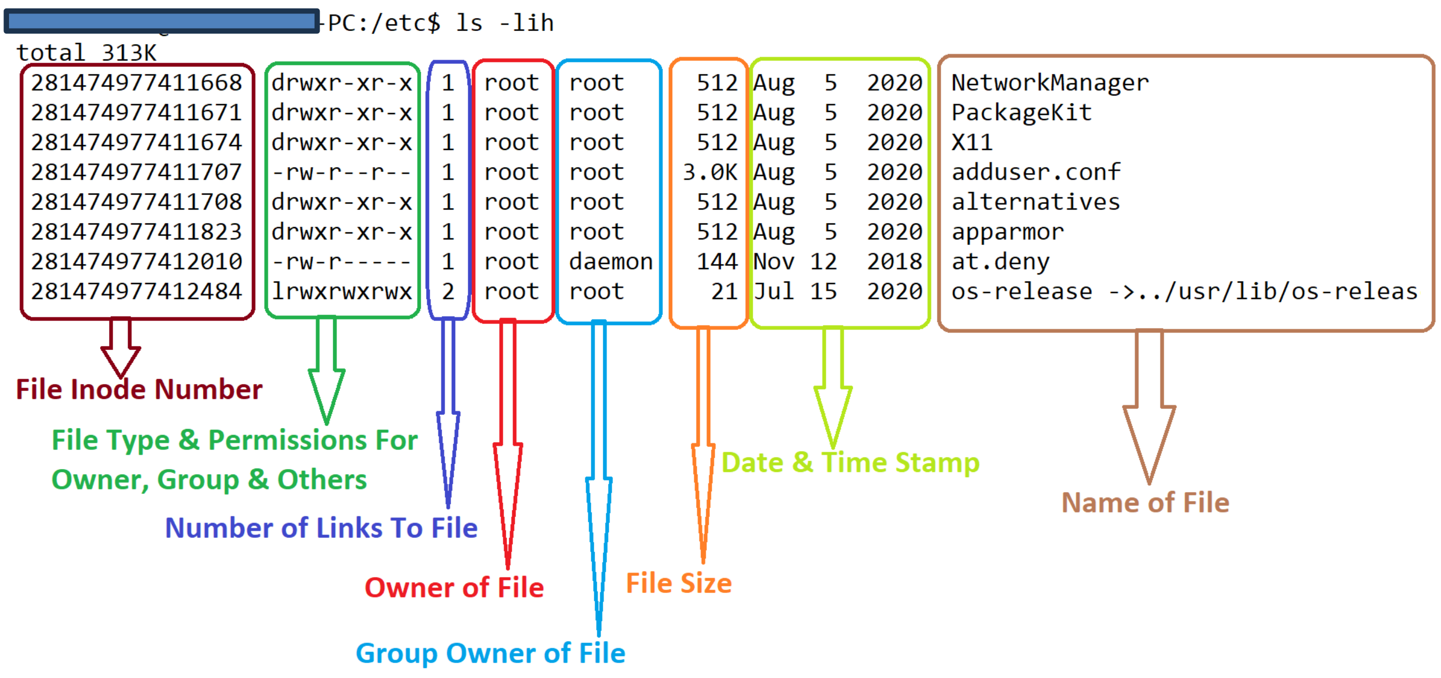
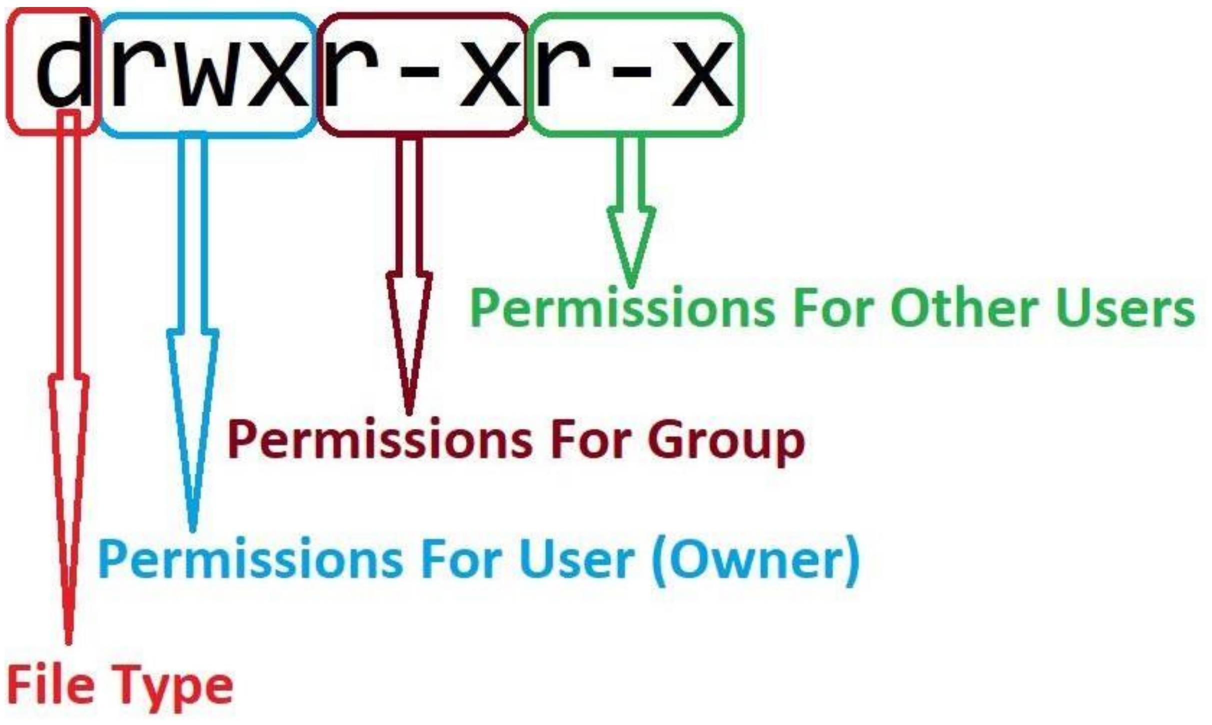
File Security: Standard File Permissions in Linux



```
[redacted]PC:/etc$ ls -lih
total 313K
281474977411668 drwxr-xr-x 1 root root      512 Aug  5  2020 NetworkManager
281474977411671 drwxr-xr-x 1 root root      512 Aug  5  2020 PackageKit
281474977411674 drwxr-xr-x 1 root root      512 Aug  5  2020 X11
281474977411707 -rw-r--r-- 1 root root     3.0K Aug  5  2020 adduser.conf
281474977411708 drwxr-xr-x 1 root root      512 Aug  5  2020 alternatives
281474977411823 drwxr-xr-x 1 root root      512 Aug  5  2020 apparmor
281474977412010 -rw-r----- 1 root daemon    144 Nov 12  2018 at.deny
281474977412484 lrwxrwxrwx 2 root root       21 Jul 15  2020 os-release ->../usr/lib/os-releas
```



first character	file type
-	normal file
d	directory
l	symbolic link
p	named pipe
b	block device
c	character device
s	socket

permission	on a file	on a directory
r (read)	read file contents (cat)	read directory contents (ls)
w (write)	change file contents (vi)	create files in (touch)
x (execute)	execute the file	enter the directory (cd)

<u>Permissions Bit Weightage:</u>

Read Permission → r = 4

Write Permission → w = 2

Execute Permission → x = 1

Maximum Permissions (Full Permission) → rwx = 4+2+1 = 7

Minimum Permissions (No Permission) → --- = 0+0+0 = 0

Permissions can be represented by 8 numbers starting from number 0 to number 7, Which means by Octal number system having a base of 8.

Octal Number System → $(0\text{-}7)_8$

We need three binary bits to represent Octal numbers. These three bits can be used to represent permission, here first bit represent read (r), second bit represent write (w) and third bit represents execute (x) permissions.

Octal Number	**Binary**	**Permissions**
0	000	---
1	001	--x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx