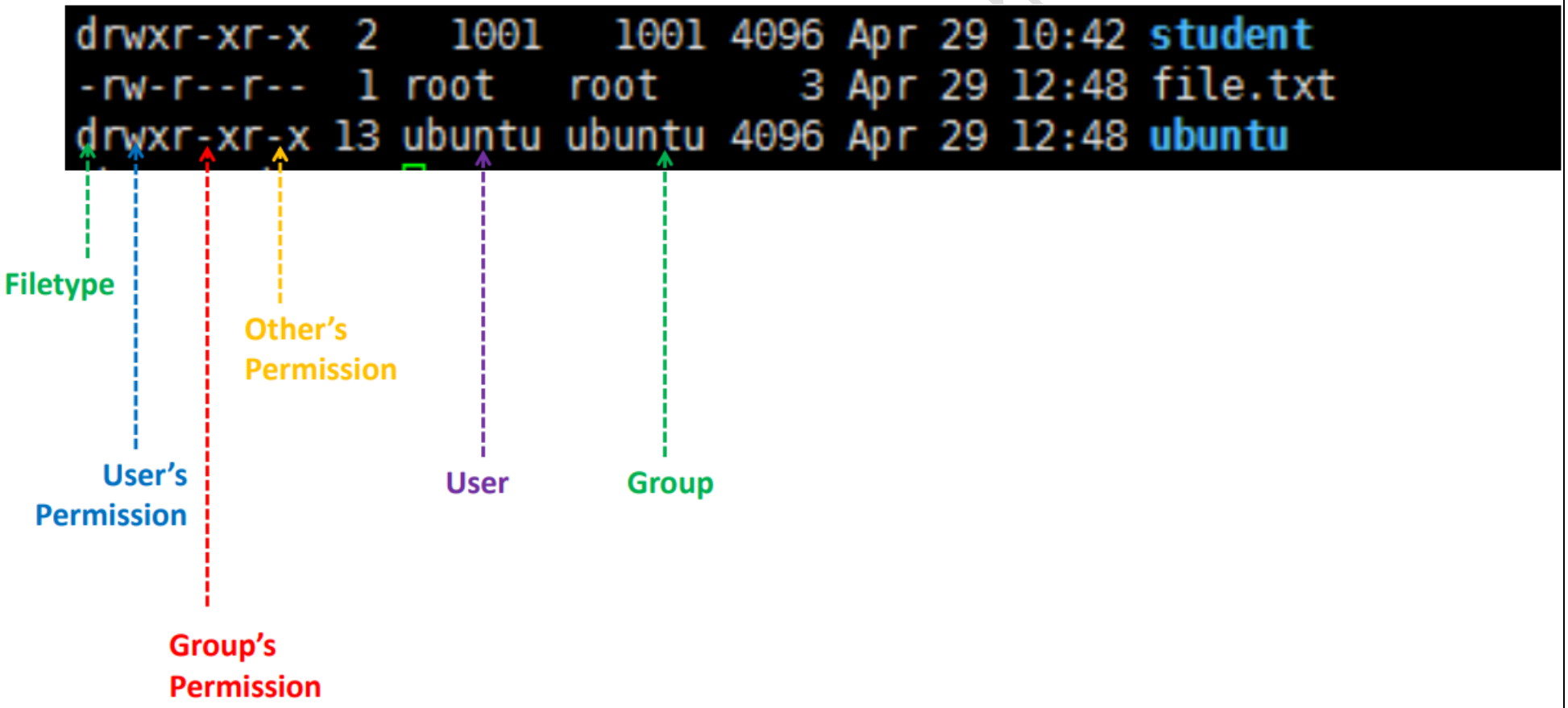
File / Directory Permission



We have 3 type of permission in Linux OS

1> Normal permission

2> Special permission

3> Access Control List (ACL)

Permission:

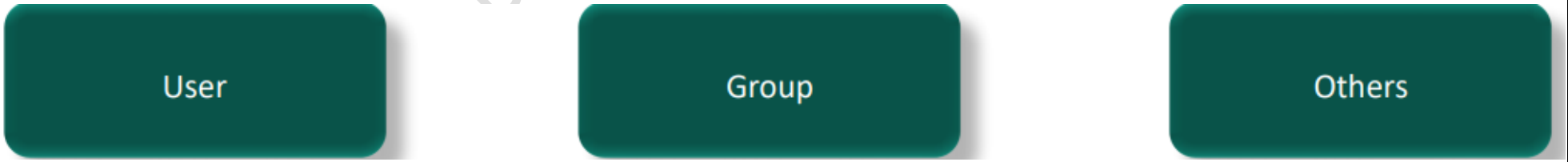
It is important for keeping our data safe and secure. The Linux file permissions system is simple but flexible, which makes it easy to understand and apply.

We have 3 categories to assign permission to a particular file or directory

a> Owner (u)

b> Group Owner (g)

c> Other Users (o)



We have 3 type of permission:

Read (r)

- Allows to view file and list content of directory.
- In binary: Just Read – 4 (100)

Write (w)

- Allows modify the file and add/delete files in directory.
- In binary: Just Write – 2 (010)

Execute (x)

- Allows to run a file and enter a directory
- In binary: Just Execute – 1 (001)

Command to modify file / dir. Permission;

#chmod *who,what,which* **Name**

Who: u g o a
What: + - =
Which: r w x

We can modify by character or by numeric:
0: Nothing
1: Execute only
2: Write only
3: Write & Execute
4: Read
5: Read & Execute
6: Read & Write
7: Read & Write & Execute

Default Permission

- The default permission for a file is 664 while it is 775 for a directory
- We can change it by setting the unmask value for the user
- To check current unmask value use command 'umask' # umask >0002
- It shows an octal value of 002
- A file gets the permission of default base_value(666) – unmask value and folder gets the permission of default base_value(777) – unmask value

Changing Ownership

We can change the user/owner of the file/folder

#chown U:G name

We can change group of the file/folder

#chgrp G name

Special Permission

Again this is also permission but it quite different from Normal permission.

We have 3 type of permission;

SPECIAL PERMISSION	EFFECT ON FILES	EFFECT ON DIRECTORIES
u+s (suid)	File executes as the user that owns the file, not the user that ran the file.	No effect.
g+s (sgid)	File executes as the group that owns the file.	Files newly created in the directory have their group owner set to match the group owner of the directory.
o+t (sticky)	No effect.	Users with write access to the directory can only remove files that they own; they cannot remove or force saves to files owned by other users.

Using same command we can assign the special permission too

#chmod	**u+s or 4664**	**File or Dir. Name**
	g+s or 2664	
	O+t or 1664	

Access Control List (ACL)

Standard Linux file permissions are satisfactory when files are used by only a single owner, and a single designated group of people. However, some use cases require that files are accessed with different file permission sets by multiple named users and groups. ***Access Control Lists (ACLs)*** provide this function.

```
[ec2-user@workstation ~]$ ll
total 4
-rw-rwxr--+ 1 ec2-user ec2-user 45 Sep  7 16:02 rana
-rw-rw-r--. 1 root     ec2-user  0 Sep  7 16:07 rana1
[ec2-user@workstation ~]$
```

+ Mark indicates rana file has ACL permission.

To get ACL permission

#getfacl filename

To set and modify ACL permission of a file

$ setfacl -m g:name:rw file

setfacl -m o::- file

setfacl -m u::rwx,g:consultants:rX,o::- file

You can use the output from getfacl as input to setfacl:

[user@host ~]$ getfacl file-A | setfacl --set-file=- file-B

For masking

setfacl -m m::r file

for recursive

setfacl -R -m u:name:rX directory

for del

setfacl -x u:name,g:name file

for default

setfacl -m d:u:name:rx directory

To delete all default ACL entries on a directory, use

$setfacl -k directory