Notes by vishalk17 (t.me/vishalk17)
Devops (06.06.2022)

groupadd command in Linux with examples

Groups in Linux refer to the user groups. In Linux, there can be many users of a single system. In a scenario where there are many users, there might be some privileges that some users have and some don't, and it becomes difficult to manage all the permissions at the individual user level. So using groups, we can group together a number of users, and set privileges and permissions for the entire group. groupadd command is used to create a new user group.

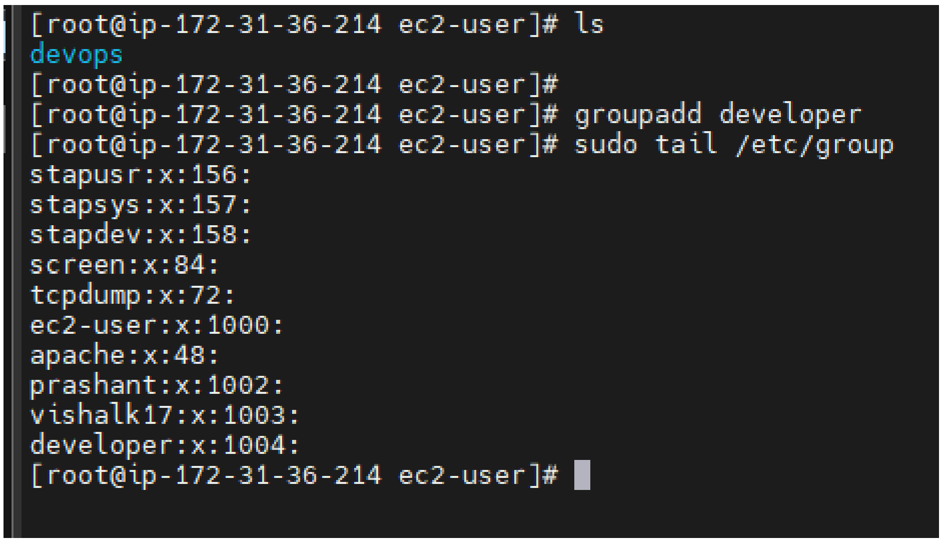
We can use either **usermod** or **gpasswd** command to add user to group in Linux operating system. And both commands are very easy to use.

Syntax

```
groupadd [option] group_name
```

Every new group created is registered in the file "/etc/group". To verify that the group has been created, enter the command

```
tail /etc/group
```



```
[root@ip-172-31-36-214 ec2-user]# ls
devops
[root@ip-172-31-36-214 ec2-user]#
[root@ip-172-31-36-214 ec2-user]# groupadd developer
[root@ip-172-31-36-214 ec2-user]# sudo tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
developer:x:1004:
[root@ip-172-31-36-214 ec2-user]#
```

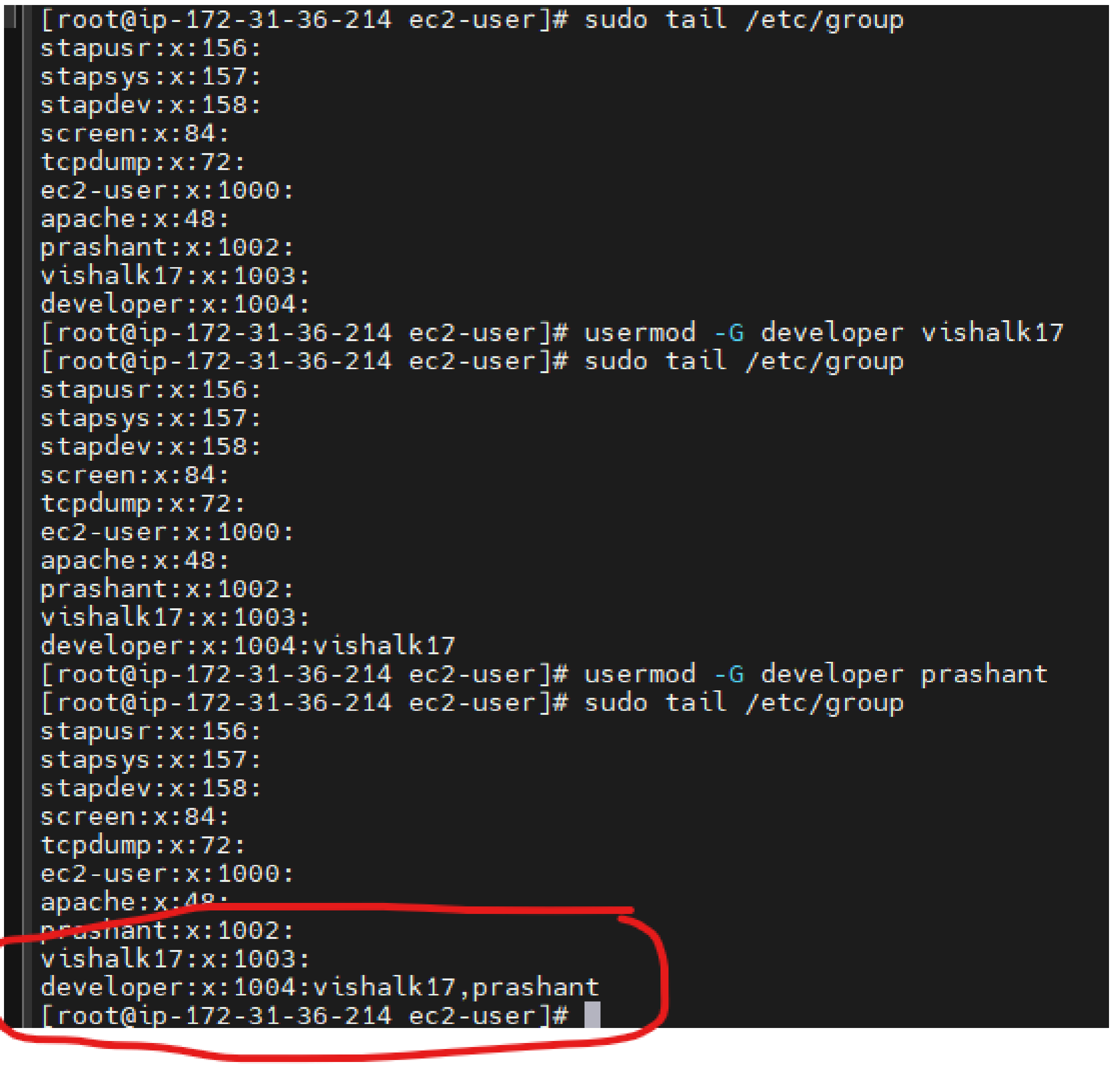
The file shows group information in the following format:

group_name : password : group-id : list-of-members

To add an existing user to a group, use the usermod command :

the group is mentioned using -g option in the command useradd

```
usermod -G groupname existing_user
```



```
[root@ip-172-31-36-214 ec2-user]# sudo tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
developer:x:1004:
[root@ip-172-31-36-214 ec2-user]# usermod -G developer vishalk17
[root@ip-172-31-36-214 ec2-user]# sudo tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
developer:x:1004:vishalk17
[root@ip-172-31-36-214 ec2-user]# usermod -G developer prashant
[root@ip-172-31-36-214 ec2-user]# sudo tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
developer:x:1004:vishalk17,prashant
[root@ip-172-31-36-214 ec2-user]#
```

The file shows group information in the following format:

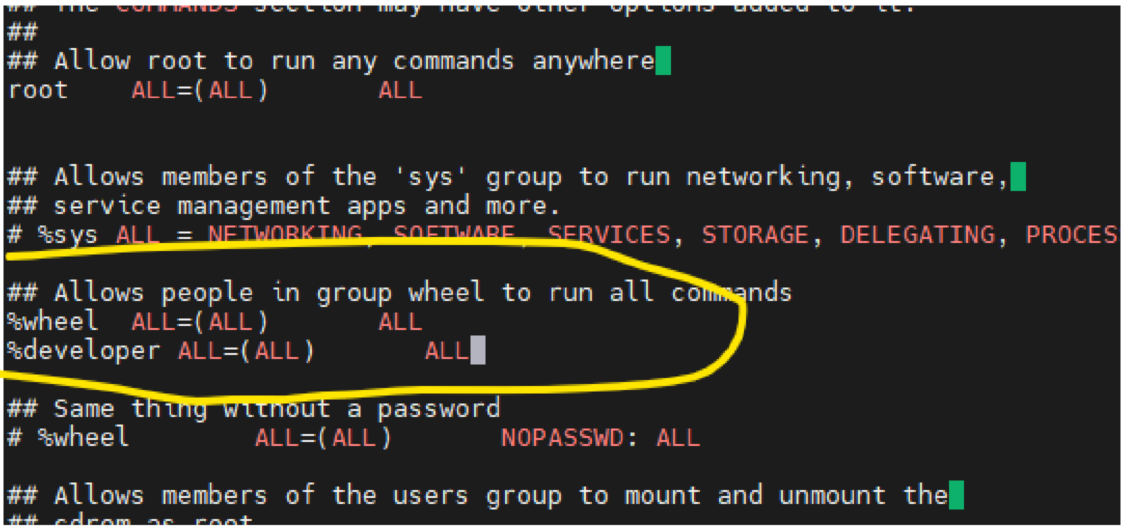
group_name : password : group-id : list-of-members-in

Every new group created is registered in the file “/etc/group“. **To verify that the group has been created, enter the command :**

```
sudo tail /etc/group
```

example case :   I want to give group “developer” root access through which members who are in same group will have root privilege automatically. So what I m going to is ,

```
visudo or use any text edior to edit /etc/sudoers
```



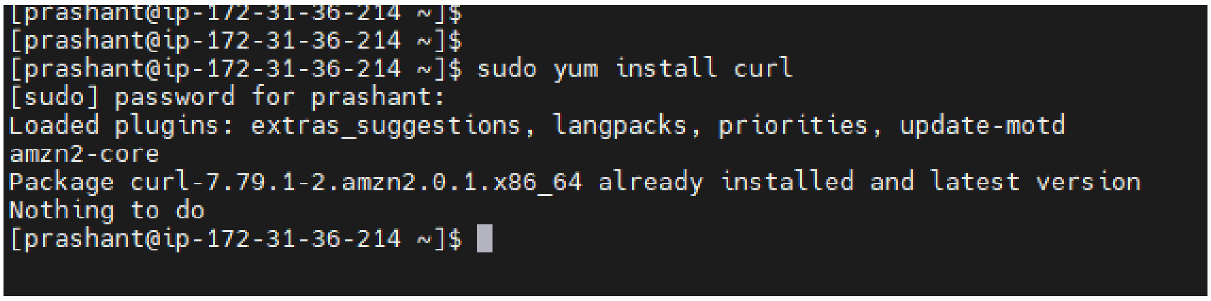
```
## The COMMANDS section may have other options added to it.
##
## Allow root to run any commands anywhere
root    ALL=(ALL)       ALL

## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCES

## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
%developer ALL=(ALL)       ALL

## Same thing without a password
# %wheel        ALL=(ALL)       NOPASSWD: ALL

## Allows members of the users group to mount and unmount the
## cdrom as root
```

Image shows developer group added. And given permission of root

Save /etc/sudoers file and exit.

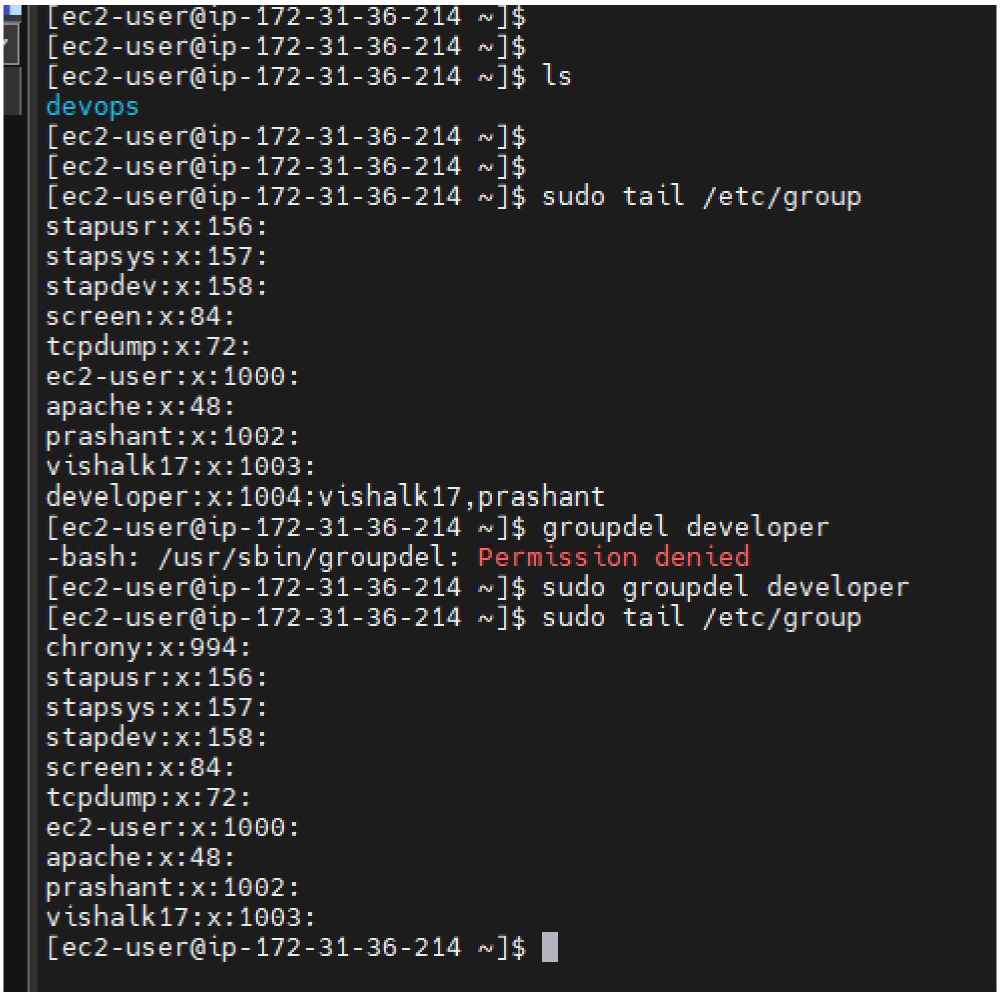
Now, members of group developer will have root privilege.

Testing:



```
[prashant@ip-172-31-36-214 ~]$
[prashant@ip-172-31-36-214 ~]$
[prashant@ip-172-31-36-214 ~]$ sudo yum install curl
[sudo] password for prashant:
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
Package curl-7.79.1-2.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
[prashant@ip-172-31-36-214 ~]$
```

How to delete group :



```
[ec2-user@ip-172-31-36-214 ~]$
[ec2-user@ip-172-31-36-214 ~]$
[ec2-user@ip-172-31-36-214 ~]$ ls
devops
[ec2-user@ip-172-31-36-214 ~]$
[ec2-user@ip-172-31-36-214 ~]$
[ec2-user@ip-172-31-36-214 ~]$ sudo tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
developer:x:1004:vishalk17,prashant
[ec2-user@ip-172-31-36-214 ~]$ groupdel developer
-bash: /usr/sbin/groupdel: Permission denied
[ec2-user@ip-172-31-36-214 ~]$ sudo groupdel developer
[ec2-user@ip-172-31-36-214 ~]$ sudo tail /etc/group
chrony:x:994:
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
[ec2-user@ip-172-31-36-214 ~]$
```

How to Manage Groups using “gpasswd” Command

The gpasswd command is used to administer /etc/group, and /etc/gshadow. Every group can have administrators, members and a password in linux to remove user from group, add users to group, list group members and set password for a group.

We can use either **usermod** or **gpasswd** command to add user to group in Linux operating system. And both commands are very easy to use.

Syntax

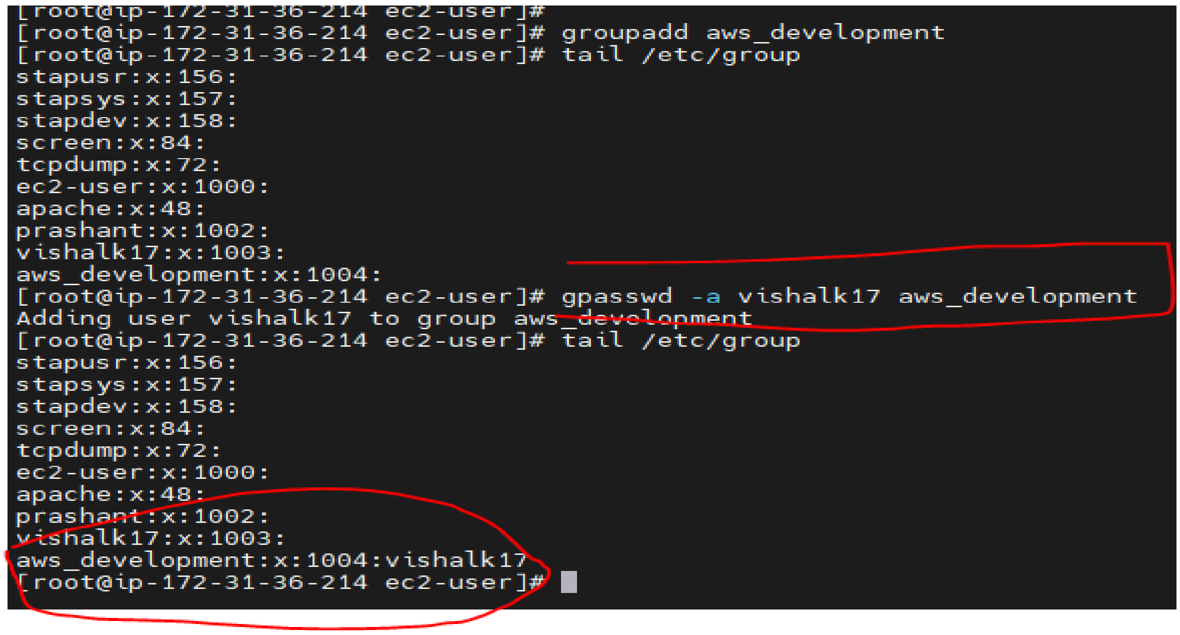
```
gpasswd [options] group
```

```
-a, --add user Add the user to the named group.

-d, --delete user Remove the user from the named group.

-A, --administrators user,... Set the list of administrative users.

-M, --members user,... Set the list of group members.
```

Add a User to a group using gpasswd comand

Let check how to add a 'existing_user' to a group 'group_name'

```
$ gpasswd -a existing_user group_name
```



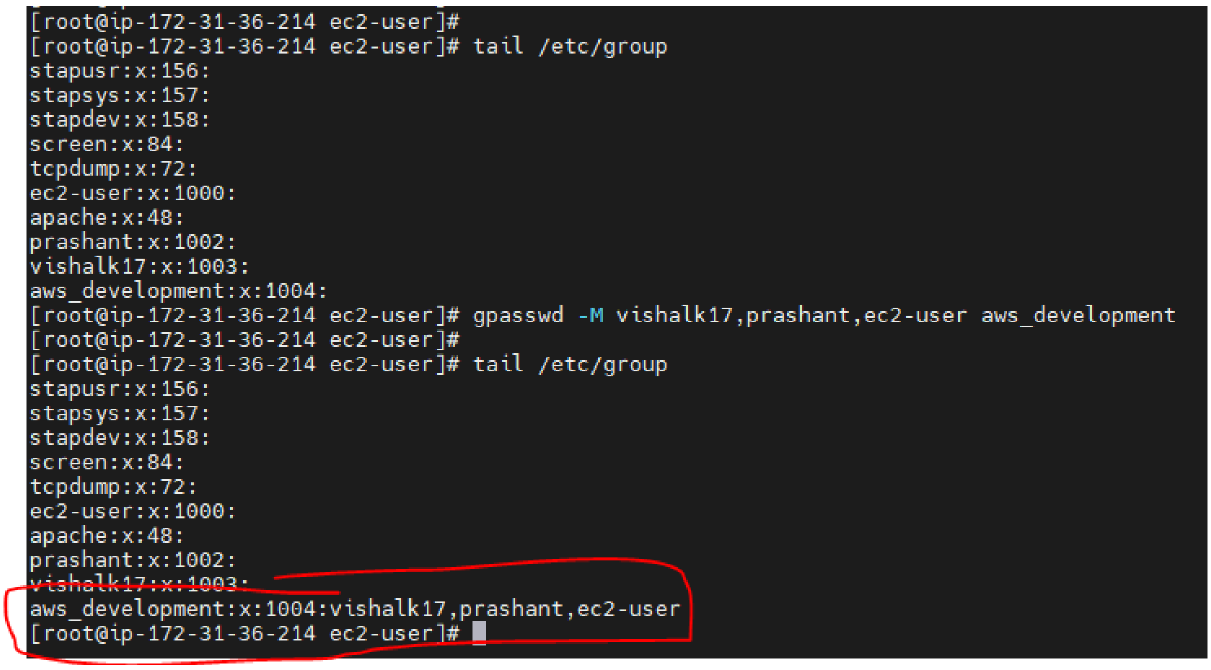
```
[root@ip-172-31-36-214 ec2-user]#
[root@ip-172-31-36-214 ec2-user]# groupadd aws_development
[root@ip-172-31-36-214 ec2-user]# tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
aws_development:x:1004:
[root@ip-172-31-36-214 ec2-user]# gpasswd -a vishalk17 aws_development
Adding user vishalk17 to group aws_development
[root@ip-172-31-36-214 ec2-user]# tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
aws_development:x:1004:vishalk17
[root@ip-172-31-36-214 ec2-user]#
```

Add multiple user to a group

To add multiple users to sales group following below command:

To add multiple users to “group_name” group following below command:

```
$ gpasswd -M user1,user2,user3 group_name
```



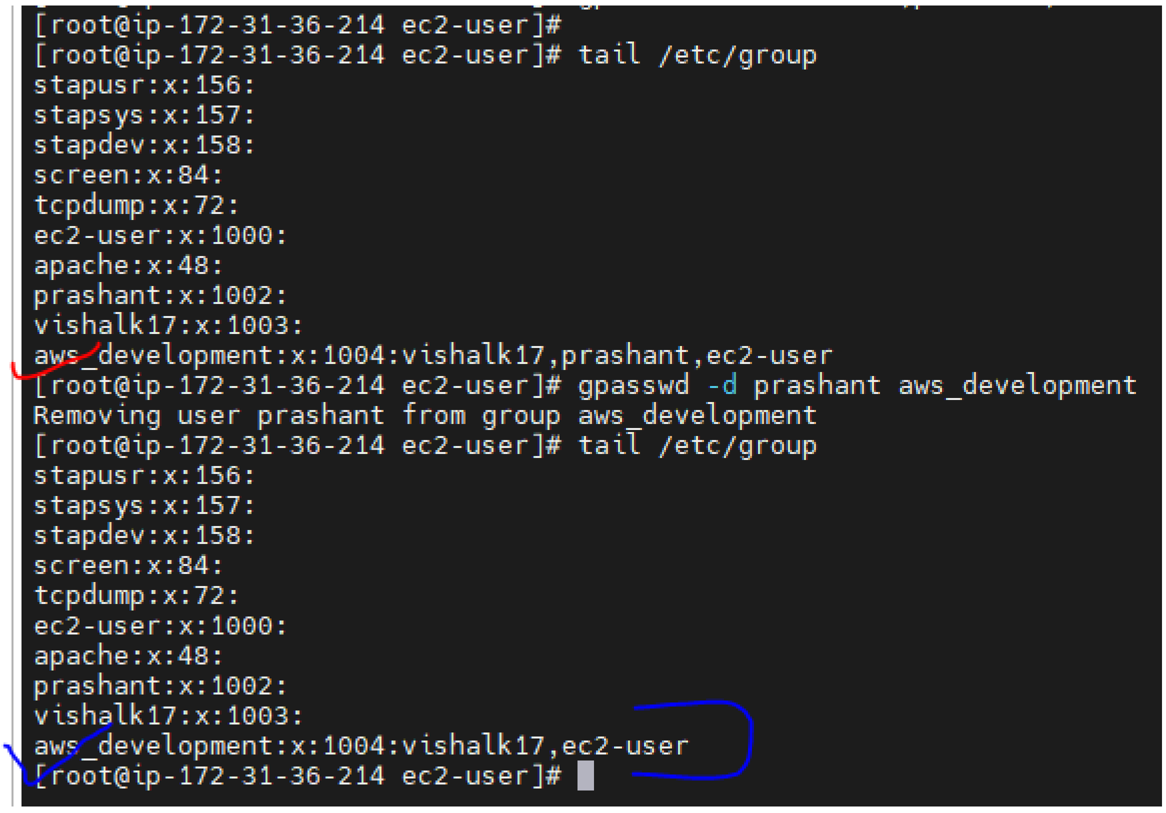
```
[root@ip-172-31-36-214 ec2-user]#
[root@ip-172-31-36-214 ec2-user]# tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
aws_development:x:1004:
[root@ip-172-31-36-214 ec2-user]# gpasswd -M vishalk17,prashant,ec2-user aws_development
[root@ip-172-31-36-214 ec2-user]#
[root@ip-172-31-36-214 ec2-user]# tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
aws_development:x:1004:vishalk17,prashant,ec2-user
[root@ip-172-31-36-214 ec2-user]#
```

Remove a user from group

We can use -d option to remove a user from a group.

Following command will remove user 'prashant' from the 'aws_development' group.

```
$ gpasswd -d user group_name
```



```
[root@ip-172-31-36-214 ec2-user]#
[root@ip-172-31-36-214 ec2-user]# tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
aws_development:x:1004:vishalk17,prashant,ec2-user
[root@ip-172-31-36-214 ec2-user]# gpasswd -d prashant aws_development
Removing user prashant from group aws_development
[root@ip-172-31-36-214 ec2-user]# tail /etc/group
stapusr:x:156:
stapsys:x:157:
stapdev:x:158:
screen:x:84:
tcpdump:x:72:
ec2-user:x:1000:
apache:x:48:
prashant:x:1002:
vishalk17:x:1003:
aws_development:x:1004:vishalk17,ec2-user
[root@ip-172-31-36-214 ec2-user]#
```

Linux permissions: chown, chgrp and chmod

In Linux EVERYTHING **is a file: folders, files themselves, programs, even hard disks! Knowing that, every file has permissions.**

Read, Write and Execute

Read, Write and Execute are the three basic permissions, you won't need anything else to get started, but before we get ahead and understand how to use them, it's important to understand what they do.

- **Read**: will enable you to read the file (or the folder's table of contents), as the name suggest.
- **Write**: with this you can modify the file (or create a new file in the case of a folder).
- **Execute**: enables you to run the program; if it is set on a folder enables you to access that folder.

Read, Write and Execute have different forms

We read them as words, but for computers it's not that simple, to tell the truth… read, write and execute are actually… numbers! (bits to be more precise). Since the entire word "read" or "execute" is way too big to be listed and repeated many times, you won't usually find permissions in the form of read/write/execute, instead you will find one of these two forms:

- **Read**: either **r** or **4**
- **Write**: either **w** or **2**
- **Execute**: either **x** or **1**

User, Group and Others

Each file will have these three *subjects*. Each *subject* will have its own permissions.

The **user** *subject* is the owner of the file, he usually has the highest level of permissions.

The **group** *subject* is the group assigned to that file, meaning that if the group has writing permission (w) each user of that group will have writing permissions (so long they are still in that group).

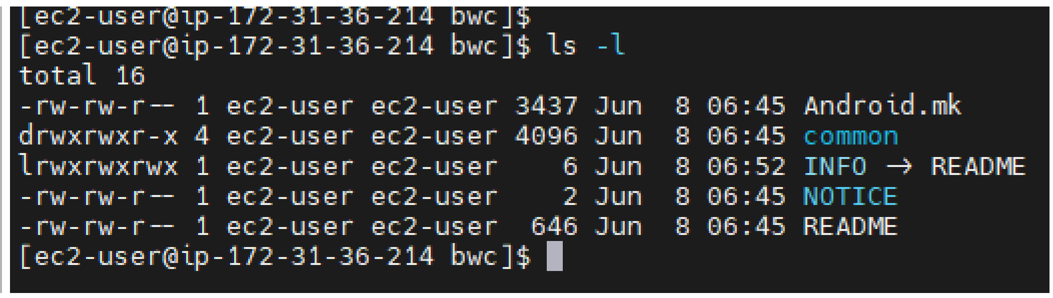
The **other** *subject* is simply ANYONE else. Be careful when assigning permissions to these three *subjects*, since they are the most basic and most permissive form of control.

****

Before we dive into the world of permissions you must learn how to check them in the terminal. This can be achieved using the ls command.

ls -l

Ls is a basic Linux command which lists the files in your current folder, using the -l flag you can learn what permissions are associated with the files. Let's give it a look!



```
[ec2-user@ip-172-31-36-214 bwc]$
[ec2-user@ip-172-31-36-214 bwc]$ ls -l
total 16
-rw-rw-r-- 1 ec2-user ec2-user 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user ec2-user 4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user ec2-user    6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user ec2-user    2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user ec2-user  646 Jun  8 06:45 README
[ec2-user@ip-172-31-36-214 bwc]$
```

Okay, now let's try to understand. Here's how ls shows its output:

file (-) /directory(d) / soft linked(l), permissions, number **of** links, user, group, size, date, time, name

let's focus only on permissions, user and group. In this case the user is ec2-user and the group is ec2-user. The permission column is a little bit more difficult, but fear not and let's analyze it:

```
-rw-rw-r—
```

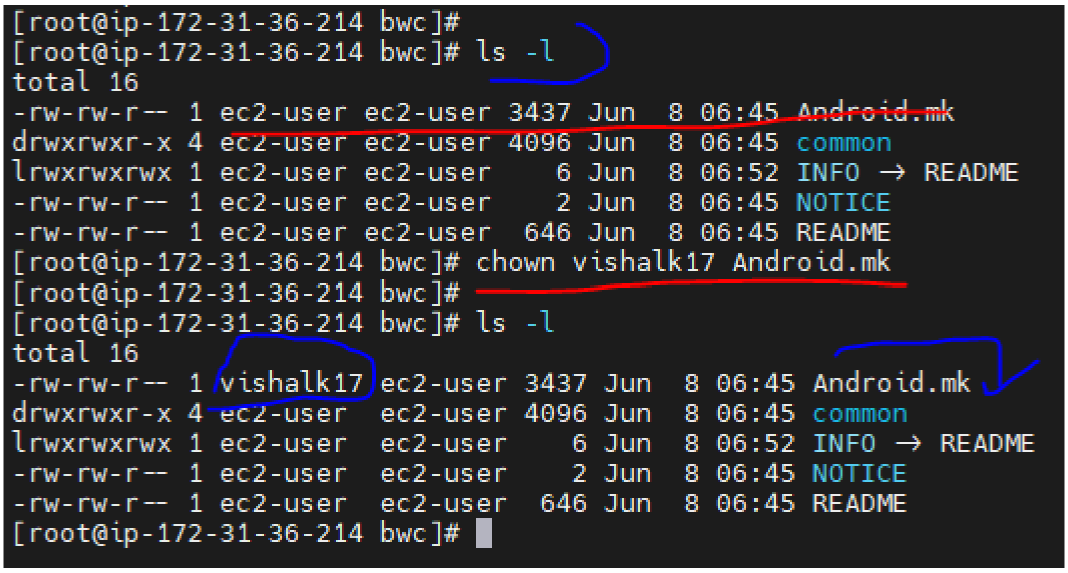
file (-) /directory(d) / soft linked(l), permissions,

The first bit isn't important for this lesson, **let's look at the remaining 9**.
The first three are the **user**'s permissions,
the second three are **group**'s permissions
the remaining three are **others** permissions

chown

It is used to change the owner (or user *subject*). Its syntax is:

```
chown owner filename
```

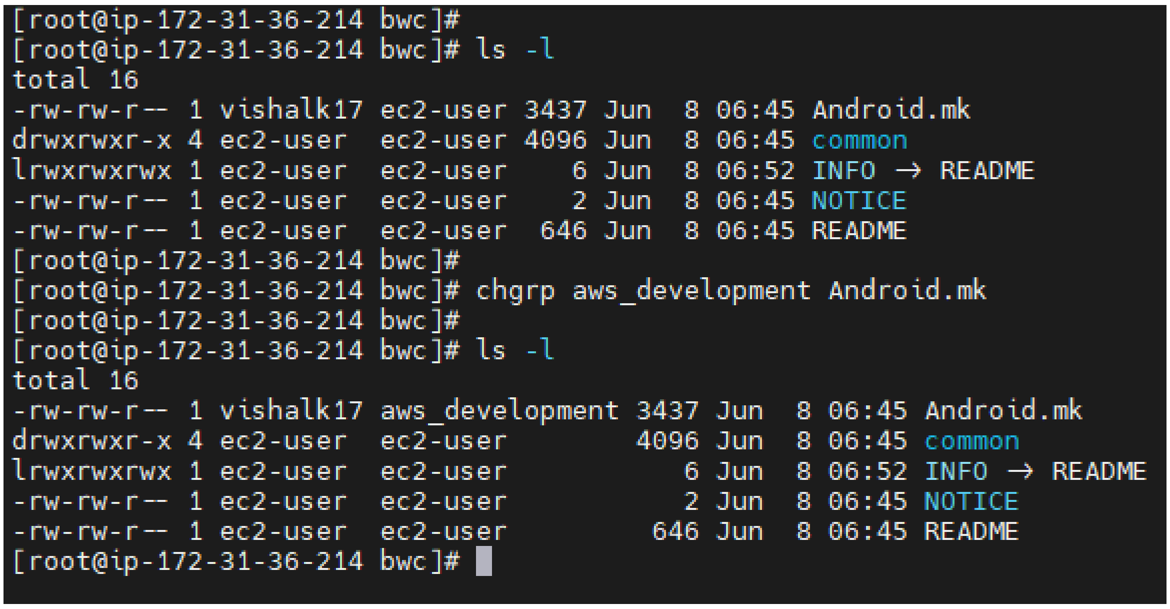


```
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rw-rw-r-- 1 ec2-user ec2-user 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user ec2-user 4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user ec2-user    6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user ec2-user    2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user ec2-user  646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]# chown vishalk17 Android.mk
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rw-rw-r-- 1 vishalk17 ec2-user 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user 4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user    6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user    2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user  646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
```

chgrp

Chown changes file's **user**, while **chgrp** changes file's **group**. Its syntax is:

```
chgrp group_name filename
```



```
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rw-rw-r-- 1 vishalk17 ec2-user 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user 4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user    6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user    2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user  646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# chgrp aws_development Android.mk
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rw-rw-r-- 1 vishalk17 aws_development 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user        4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user           6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user           2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user         646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
```

chmod

It is used to change the permissions of the three *subjects* and its syntax is:

```
$ chmod permission filename/foldername
```

Now the problem is what goes into the permission field?

I mentioned you can use two forms to represent permissions: **r w x** or **4 2 1**, in this case we'll be using the numerical form.

Example,

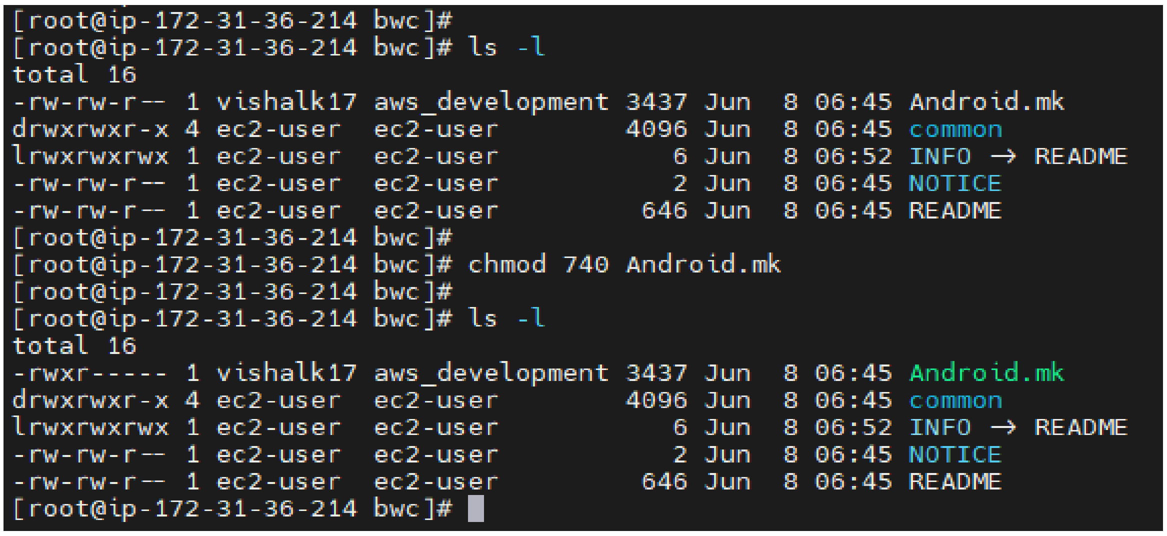
let's suppose we want to assign :

read+write+execute to **owner**,
read to **group** and
none to **others**.

- **Read**: either **r** or **4**
- **Write**: either **w** or **2**
- **Execute**: either **x** or **1**

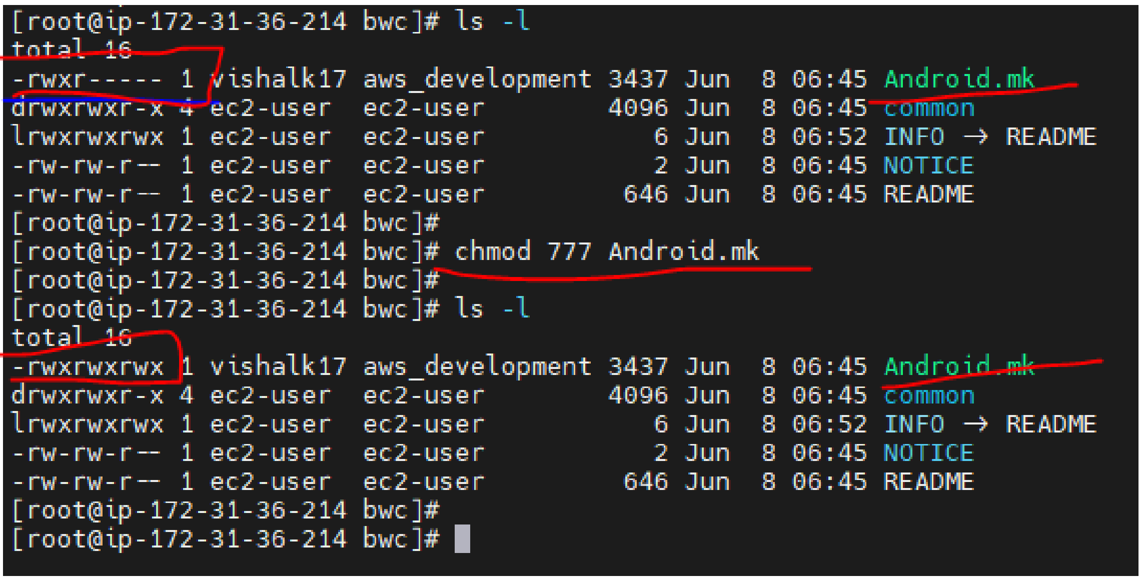
It's time to calculate:
- owner will be read+write+execute=4+2+1=7
- group will be read=4
- others will be = 0

So, number is 740! Let's use it on Android.mk



```
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rw-rw-r-- 1 vishalk17 aws_development 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user        4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user           6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user           2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user         646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# chmod 740 Android.mk
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rwxr----- 1 vishalk17 aws_development 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user        4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user           6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user           2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user         646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
```

Notice how it changed. Now suppose we want to full permissions to everyone! Can you guess which number I will use?



```
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rwxr----- 1 vishalk17 aws_development 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user        4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user           6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user           2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user         646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# chmod 777 Android.mk
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]# ls -l
total 16
-rwxrwxrwx 1 vishalk17 aws_development 3437 Jun  8 06:45 Android.mk
drwxrwxr-x 4 ec2-user  ec2-user        4096 Jun  8 06:45 common
lrwxrwxrwx 1 ec2-user  ec2-user           6 Jun  8 06:52 INFO -> README
-rw-rw-r-- 1 ec2-user  ec2-user           2 Jun  8 06:45 NOTICE
-rw-rw-r-- 1 ec2-user  ec2-user         646 Jun  8 06:45 README
[root@ip-172-31-36-214 bwc]#
[root@ip-172-31-36-214 bwc]#
```

If you said 777, then you’ve answered correctly!