Permissions

Permission Groups

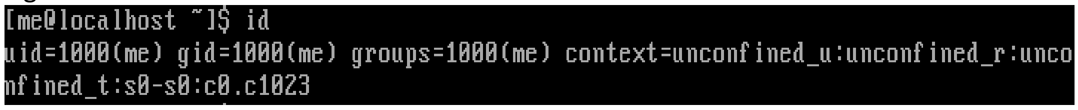
- **Owner**: It applies only to the owner of the file or directory; this will not impact the actions of other users.
- **Group**: It applies only to the group that has been assigned to the file or directory; it will not affect the actions of other users.
- **All Users**: It applies to all other users on the system; this is the permission group that you want to watch the most.

Permission Types

- **Read**: User can view the contents of the file.
- **Write**: User can modify a file or directory.
- **Execute**: User can run a file or view the contents of a directory.

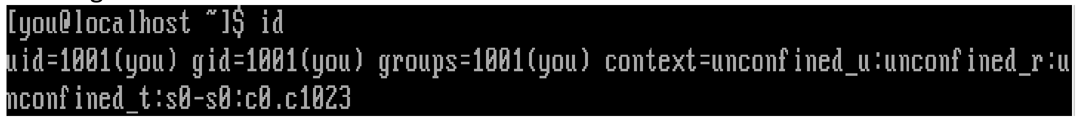
Permission Commands

- `id`: It displays user identity.
- `chmod`: It allows you to modify the access rights of a file/directory.
- `umask`: It sets the default file permissions.
- `su`: It allows you to temporarily become the superuser.
- `sudo`: It allows you to execute a command as another user.
- `chown`: It allows you to change a file's ownership.
- `chgrp`: It allows you to change a file's group ownership.
- `passwd`: It allows you to change a user's password.
- Running the `id` Command:



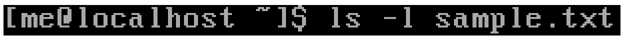
```
[me@localhost ~]$ id
uid=1000(me) gid=1000(me) groups=1000(me) context=unconfined_u:unconfined_r:unco
nfined_t:s0-s0:c0.c1023
```

- After creating new account:



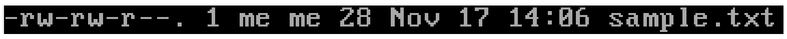
```
[you@localhost ~]$ id
uid=1001(you) gid=1001(you) groups=1001(you) context=unconfined_u:unconfined_r:u
nconfined_t:s0-s0:c0.c1023
```

- To view permission settings, use `ls` command:



```
[me@localhost ~]$ ls -l sample.txt
```

- Sample result"

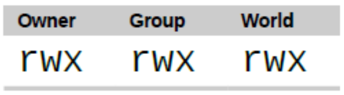


```
-rw-rw-r--. 1 me me 28 Nov 17 14:06 sample.txt
```

- Common File Types:

Attribute	File Type
-	A regular file
d	A directory
l	A symbolic link
c	A character special file This file type refers to a device that handles data as a stream of bytes, such as terminal or modem.
b	A block special file This file type refers to a device that handles data in blocks, such as hard-drive or CD-ROM drive.

- File Mode:



Owner	Group	World
rwx	rwx	rwx

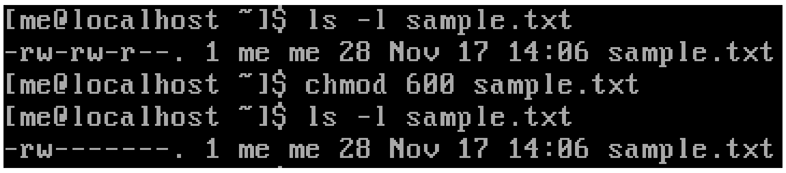
- Octal Notation used for **chmod** command:

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

- File Permission Values"

Value	Meaning
777	**(rwxrwxrwx)** It sets no restrictions on permissions. Anybody may do anything. This is not a recommended setting.
755	**(rwxr-xr-x)** The file's owner may read, write, and execute the file. All others may read and execute the file. This setting is common for programs that are used by all users.
700	**(rwx------)** The file's owner may read, write, and execute the file. Others are not given any rights. This setting is useful for programs that only the owner may use and must be kept private from others.
666	**(rw-rw-rw-)** All users may read and modify the file.
644	**(rw-r--r--)** The owner may read and modify the file, while all others may only read the file. This is a common setting for data files that everybody may read, but only the owner may change.
600	**(rw-------)** The owner may read and modify the file. All others have no rights. This is a common setting for data files that the owner wants to keep private.

- Sample Usage:



```
[me@localhost ~]$ ls -l sample.txt
-rw-rw-r--. 1 me me 28 Nov 17 14:06 sample.txt
[me@localhost ~]$ chmod 600 sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-------. 1 me me 28 Nov 17 14:06 sample.txt
```

- Directory Permission Values:

Value	Meaning
777	**(rwxrwxrwx)** It sets no restrictions on permissions. Anybody may list files, create new files in the directory, and delete files in the directory. This is not a recommended setting.
755	**(rwxr-xr-x)** The directory owner has full access. All others may list the directory, but cannot create files nor delete them. This setting is common for directories that you wish to share with other users.
700	**(rwx------)** The directory owner has full access. Others are not given any rights. This setting is useful for directories that only the owner may use and must be kept private from others.

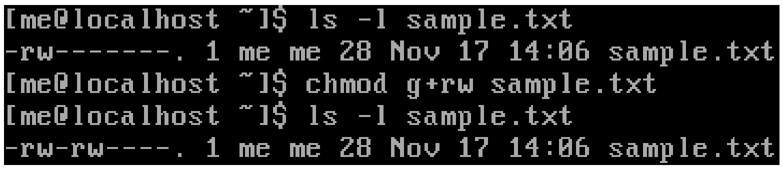
- Symbolic Notation:

Symbol	Meaning
u	The directory or file owner
g	The group owner
o	Others
a	Short for all The combination of u, g, and o

- Examples of Symbolic Notation:

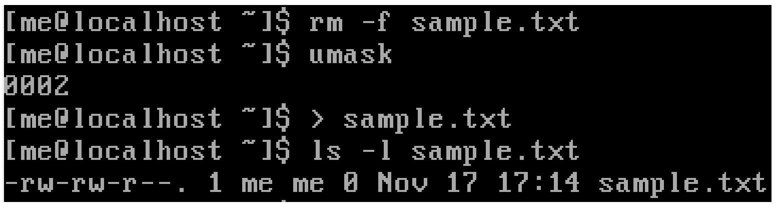
Notation	Meaning
u+x	It allows execute permission for the owner .
u-x	It removes execute permission from the owner.
+x	It allows execute permission for the owner, group, and everyone else. This is equivalent
o-rw	It removes the read and write permission from anyone besides the owner and group owner.
go=rw	It sets the group owner and anyone besides the owner to have read and write permission. It removes the execute permissions from the group owner and others.
u+x, go=rx	It adds execute permission for the owner and sets the permissions for the group and others to read and execute. Multiple specifications may be separated by commas.

- Sample Usage:



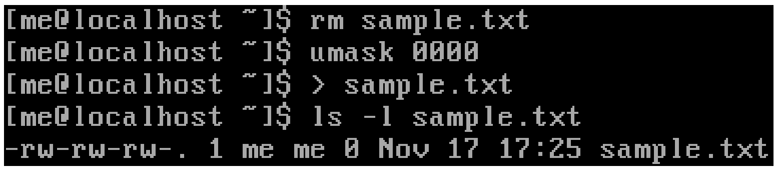
```
[me@localhost ~]$ ls -l sample.txt
-rw-------. 1 me me 28 Nov 17 14:06 sample.txt
[me@localhost ~]$ chmod g+rw sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw----. 1 me me 28 Nov 17 14:06 sample.txt
```

- The **umask** Command:



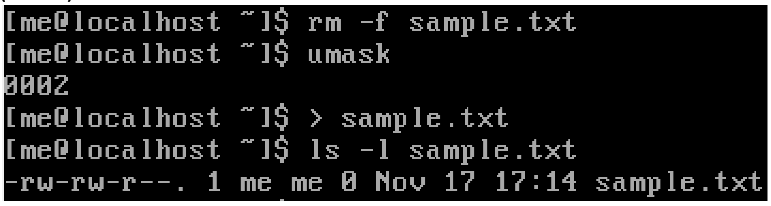
```
[me@localhost ~]$ rm -f sample.txt
[me@localhost ~]$ umask
0002
[me@localhost ~]$ > sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-r--. 1 me me 0 Nov 17 17:14 sample.txt
```

- Setting **umask**:



```
[me@localhost ~]$ rm sample.txt
[me@localhost ~]$ umask 0000
[me@localhost ~]$ > sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me me 0 Nov 17 17:25 sample.txt
```

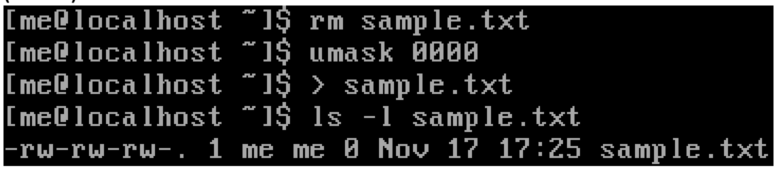
- Binary equivalent (0002):



```
[me@localhost ~]$ rm -f sample.txt
[me@localhost ~]$ umask
0002
[me@localhost ~]$ > sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-r--. 1 me me 0 Nov 17 17:14 sample.txt
```

Mask	000 000 000 010
Result	--- rw- rw- r--

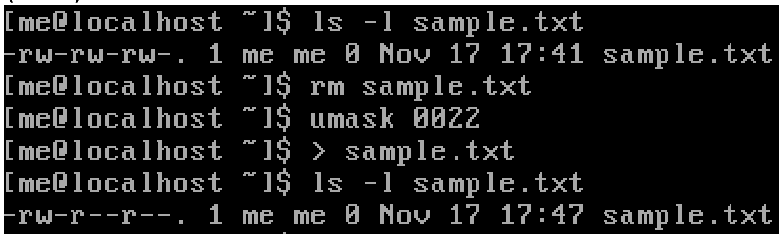
- Binary equivalent (0000):



```
[me@localhost ~]$ rm sample.txt
[me@localhost ~]$ umask 0000
[me@localhost ~]$ > sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me me 0 Nov 17 17:25 sample.txt
```

Original File Mode	--- rw- rw- r--
Mask	000 000 000 000
Result	--- rw- rw- rw-

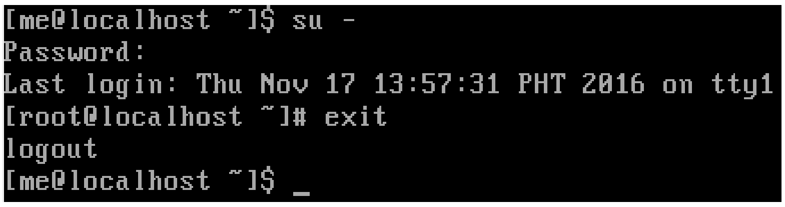
- Binary equivalent (0022):



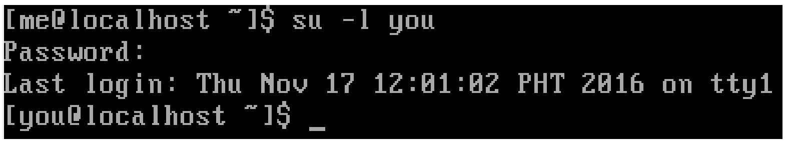
```
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me me 0 Nov 17 17:41 sample.txt
[me@localhost ~]$ rm sample.txt
[me@localhost ~]$ umask 0022
[me@localhost ~]$ > sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-r--r--. 1 me me 0 Nov 17 17:47 sample.txt
```

Original File Mode	`--- rw- rw- rw-`
Mask	`000 000 010 010`
Result	`--- rw- r-- r--`

- The **su** Command:

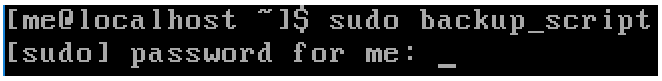


```
[me@localhost ~]$ su -
Password:
Last login: Thu Nov 17 13:57:31 PHT 2016 on tty1
[root@localhost ~]# exit
logout
[me@localhost ~]$ _
```



```
[me@localhost ~]$ su -l you
Password:
Last login: Thu Nov 17 12:01:02 PHT 2016 on tty1
[you@localhost ~]$ _
```

- The **sudo** Command:

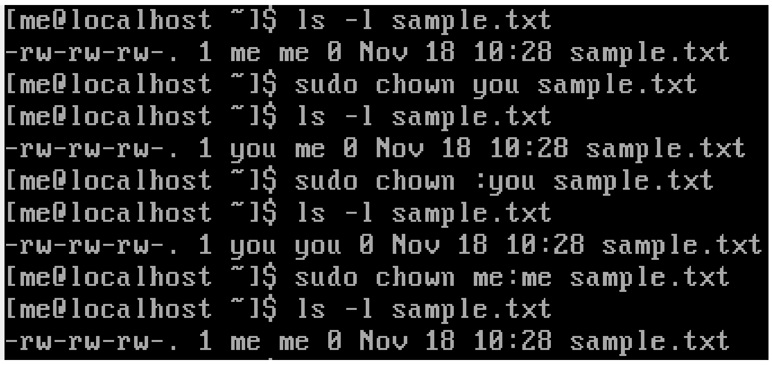


```
[me@localhost ~]$ sudo backup_script
[sudo] password for me: _
```

- Sample arguments for the **chown** command:

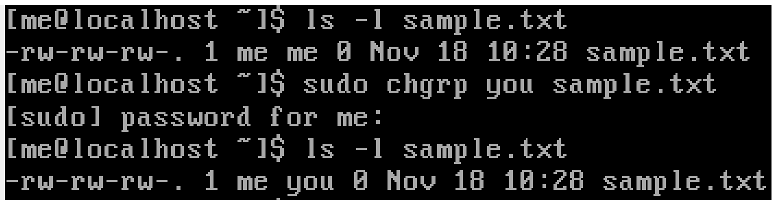
Argument	**Result**
`nika`	The ownership of the file is changed from its current owner to user "nika".
`nika:users`	The ownership of the file is changed from its current owner to user "nika" and from its current group owner to group "users".
`:admins`	The ownership of the file is changed from its current group owner to group "admins".
`nika:`	The ownership of the file is changed from its current owner to user "nika". The group owner is the login group of user "nika" too.

- Sample Usages:



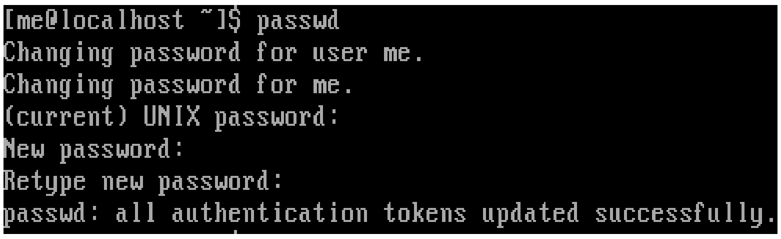
```
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me me 0 Nov 18 10:28 sample.txt
[me@localhost ~]$ sudo chown you sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 you me 0 Nov 18 10:28 sample.txt
[me@localhost ~]$ sudo chown :you sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 you you 0 Nov 18 10:28 sample.txt
[me@localhost ~]$ sudo chown me:me sample.txt
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me me 0 Nov 18 10:28 sample.txt
```

- The **chgrp** Command:

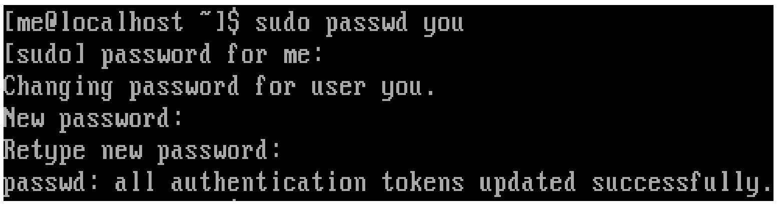


```
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me me 0 Nov 18 10:28 sample.txt
[me@localhost ~]$ sudo chgrp you sample.txt
[sudo] password for me:
[me@localhost ~]$ ls -l sample.txt
-rw-rw-rw-. 1 me you 0 Nov 18 10:28 sample.txt
```

- The **passwd** Command:



```
[me@localhost ~]$ passwd
Changing password for user me.
Changing password for me.
(current) UNIX password:
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```



```
[me@localhost ~]$ sudo passwd you
[sudo] password for me:
Changing password for user you.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

References:

Barret, Daniel J. (2016). *Linux pocket guide.* 3rd ed. USA: O'Reilly Media, Inc.

Fox, Richard. (2015). *Linux with operating system concepts.* USA: CRC Press, Taylor & Francis Group, LLC.

Shotts, William E. Jr. (2016). *The linux command line.* 3rd ed. USA: No Starch Press.