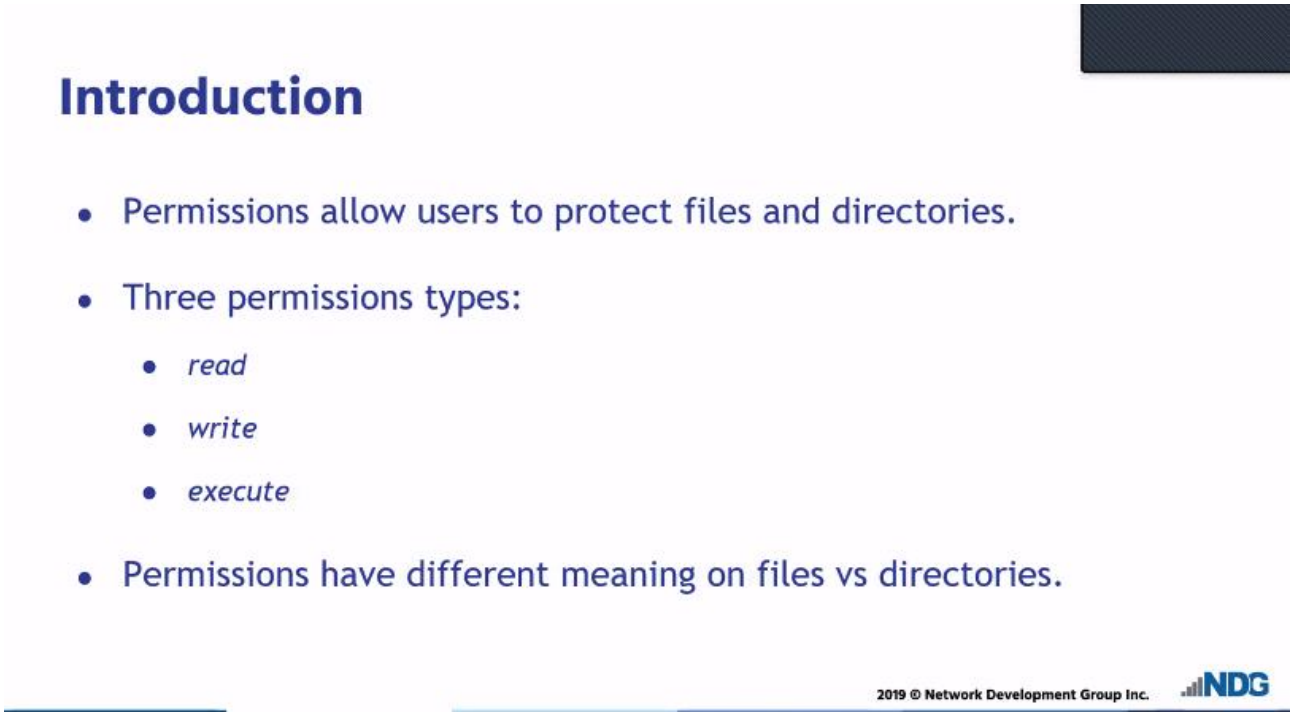
Comandi:

- bg (background): manda il processo da foreground a background
- fg (foreground): manda il processo da background a foreground

MODULO 13: FILE PERMISSIONS



Changing File User Owner

- When a file or directory is created, the owner is automatically assigned using the effective user ID at the time of creation
- The `chown` command is used to change the ownership of a file or directory
- File owner can only be changed by a user with root privileges:

```
sysadmin@localhost:~$ touch nullfile
sysadmin@localhost:~$  ls -l nullfile
-rw-rw-r-- 1 sysadmin sysadmin 0 Mar 21 17:30 nullfile
sysadmin@localhost:~$ sudo chown root nullfile
[sudo] password for sysadmin:
sysadmin@localhost:~$ ls -l nullfile
-rw-rw-r-- 1 root sysadmin 0 Mar 21 17:30 nullfile
```


Changing File User Owner

- When a file or directory is created, the owner is automatically assigned using the effective user ID at the time of creation
- The `chown` command is used to change the ownership of a file or directory
- File owner can only be changed by a user with root privileges:

```
sysadmin@localhost:~$ touch nullfile
sysadmin@localhost:~$  ls -l nullfile
-rw-rw-r-- 1 sysadmin sysadmin 0 Mar 21 17:30 nullfile
sysadmin@localhost:~$ sudo chown root nullfile
[sudo] password for sysadmin:
sysadmin@localhost:~$ ls -l nullfile
-rw-rw-r-- 1 root sysadmin 0 Mar 21 17:30 nullfile
```

2019 © Network Development Group Inc.

Switching Groups

- To see what groups you belong to, then you can execute the groups command:

```
sysadmin@localhost:~$ groups
sysadmin adm sudo
```

- To create a file or directory that will be owned by a group different from your current primary group use the `newgrp` command.

```
newgrp [GROUP]
```

- For example, any new files or directories created will be group owned by the adm group (instead of the sysadmin group):

```
sysadmin@localhost:~$ newgrp adm
sysadmin@localhost:~$ ls -l newfile.txt
-rw-rw-r-- 1 sysadmin adm 0 Mar 21 20:00 newfile.txt
```


Changing File Group Owner

- If you forget to use the `newgrp` command to switch to the user's primary group before creating a file, you can use the `chgrp` command:

```
sysadmin@localhost:~$ touch newfile2.txt
sysadmin@localhost:~$ ls -l newfile2.txt
-rw-rw-r-- 1 sysadmin sysadmin 0 Mar 21 18:03 newfile2.txt
sysadmin@localhost:~$ sudo chgrp nogroup
newfile2.txt
sysadmin@localhost:~$ ls -l newfile2.txt
-rw-rw-r-- 1 sysadmin nogroup 0 Mar 21 18:03 newfile2.txt
```

- Only the owner of the file and the root user can change the group ownership of a file.

Displaying permissions

- Permissions are displayed with `ls -l` command:

```
sysadmin@localhost:~$ ls -l /bin/ls
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

  - File Type Field; indicates the *type of a file* (i.e. file, directory, link etc.)

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

  - Permission Field

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

  - User Owner Field

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

  - Group Owner Field

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```


Understanding Permission

- Permissions are broken into three sets:
  - - The first set for the user who owns the file displayed:

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

  - - The second set for the group that owns the file displayed:

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

  - - The last set for everyone else:

```
-rwxr-xr-x 1 root root 133792 Jan 18 2018 /bin/ls
```

- The term "everyone else" means anyone who is not the user that owns the file or a member of the group that owns the file.

Files vs directories

Permission (Symbol)	Effect on File	Effect on Directory
read (r)	Allows for file contents to be read and copied.	File names in the directory can be listed, but other details are not available.
write (w)	Allows for contents to be written to by the process, so changes to a file can be saved.	Files can be added to or removed from the directory.
execute (x)	Allows for a file to be executed or run as a process.	Allows for the user to use the cd command to "get into" the directory and use the directory in a pathname to access files and subdirectories.

Changing Basic File Permissions

- The chmod command is used to change the permissions of a file or directory.

```
chmod MODE FILE…
```

- To change the permissions of a file, you must either be the user who owns the file or the root user.
- There are two methods used to change permissions:
  - Symbolic (relative) method - uses a combination of letters and symbols to add or remove
  - Octal (numeric) method - uses three numbers to represent file permissions for owner, group, everyone else

The symbolic Method

- First, specify who (u, g, o, a):

Symbol	Meaning
u	The user who owns the file
g	The group who owns the file
o	People other than the user owner or member of the group owner (others)
a	To refer to the user, group and others (all)

- Next, specify an operator (+, =, -):

Symbol	Meaning
+	Add the permission, if necessary
=	Specify the exact permission
-	Remove the permission, if necessary

- Lastly, specify the permissions (r, w, x).

The Symbolic method

- Examples:

Example	Meaning
`chmod u+x myscript`	Adds the *execute* permission for the user owner.
`chmod g-w file1`	Removes the *write* permission from the group owner.
`chmod o=r,g-w,u+x myscript`	Assigns others the *read* permission, removes *write* from the group owner and adds *execute* to the user owner.
`chmod a=- file1`	Assign everyone no permissions.

The octal method

- Uses numeric values for permissions

Permission	Octal Value
read (r)	4
write (w)	2
execute (x)	1

- Must always specify 3 values (owner, group, everyone else)

Example	Meaning
chmod 764 myscript	Results in rwxrw-r--
chmod 644 myfile	Results in rw-r--r--
chmod 744 myscript	Results in rwxr--r--
chmod 000 myfile	Results in ---------

Changing Advanced File Permissions

Permission	Symbol	Octal Value	Purpose
setuid on a file	An `s` where you normally see the `x` for the user owner permissions , set with `u+s`	4000	Causes an executable file to execute under user owner identity instead of user running command.
setgid on a file	An `s` where you normally see the `x` for the group owner permissions , set with `g+s`	2000	Causes an executable file to execute under group owner identity instead of user running command.

Default File Permissions

- The `umask` command is used to set default permissions
- Only affects the permissions placed on new files and directories at the time they are created
- Does not affect the special advanced permissions of setuid, setgid or sticky bit

Default File Permissions

- The `umask` command uses same numeric values as `chmod`:

Octal Value	Permission
4	read
2	write
1	execute
0	none

- The umask is a value subtracted from the following maximum permissions:

File Type	Octal	Symbolic
Files	666	`rw-rw-rw-`
Directories	777	`rwxrwxrwx`

Default File Permissions

- The `umask` command is automatically executed when a shell is started.
- A custom `umask` command can be added to the `~/.bashrc` file.
- The `umask` command can be used to display the current umask value:

```
sysadmin@localhost:~$ umask
0002
```

- In the output above:
  - The first `0` is for special permissions (setuid, setgid and sticky bit).
  - The second `0` (`---`) indicates which permissions to subtract from the default user owner
  - The third `0` (`---`) indicates which permissions to subtract from the default group owner.
  - The last number `2` (`-w-`) indicates which permissions to subtract from others.

Understanding Umask for Files

- By default, the maximum permissions that will be placed on a brand-new file are `rw-rw-rw-` (or `666`).
- To set a umask value for new files that would result in:
  - Default permissions for the user owner.
  - Remove (mask) write permission for the group owner.
  - Remove all permissions from others.
- Calculate the umask value as follows:

File Default	666	rw-rw-rw-
Umask	-026	----w-rw-
Result	640	rw-r-----

Understanding Umask for Directories

- For directories, the execute permission is critical to access the directory properly so the default permissions are `rwxrwxrwx` (or `777`).
- To set a umask value for new directories that would result in:
  - Full permissions for the user owner.
  - Remove (mask) write permission for the group owner.
  - Remove all permissions from others.
- Calculate the umask value as follows:

File Default	777	rwxrwxrwx
Umask	-027	----w-rwx
Result	640	rwxr-x---

MODULO 15

MODULO 18