Systèmes d'exploitation : Partie 2

Objectifs :

- Identifier les fonctions d'un système d'exploitation.
- **Utiliser les commandes de base en ligne de commande.**
- **Gérer les droits et permissions d'accès aux fichiers.**
- Les différences entre systèmes d'exploitation libres et propriétaires sont évoquées.

Introduction :

À la "préhistoire" des systèmes d'exploitation, ces derniers étaient dépourvus d'interface graphique (système de fenêtres "pilotables" à la souris), toutes les interactions "système d'exploitation - utilisateur" se faisaient par l'intermédiaire de "lignes de commandes" (suites de caractères, souvent ésotériques, saisies par l'utilisateur). Aujourd'hui, même si les interfaces graphiques modernes permettent d'effectuer la plupart des opérations, il est important de connaître quelques-unes de ces lignes de commandes. Pour saisir des lignes de commandes, nous allons utiliser une console (aussi appelé terminal même si ce n'est pas exactement la même chose).

Commençons donc par ouvrir un terminal sous GNU/linux et sous Windows (commande : cmd.exe), vous devriez avoir quelques chose qui ressemble à ça :

`moi@mongroupe:~$`	`C:\Users\moi>`
L'invite de commande **Ubuntu** se compose : du nom d'utilisateur (ici `moi`) ; de `@` et de son groupe ou du nom de la machine (ici `mongroupe`) ; suivi de `:` ; du répertoire courant (`~` pour le répertoire utilisateur) et termine par `$` .	L'invite de commande « MS-DOS » de **Windows** se compose du nom du lecteur (ici `C`) ; suivi de `:` ; du répertoire courant (ici le chemin est `\Users\moi`) ; et se termine par `>`

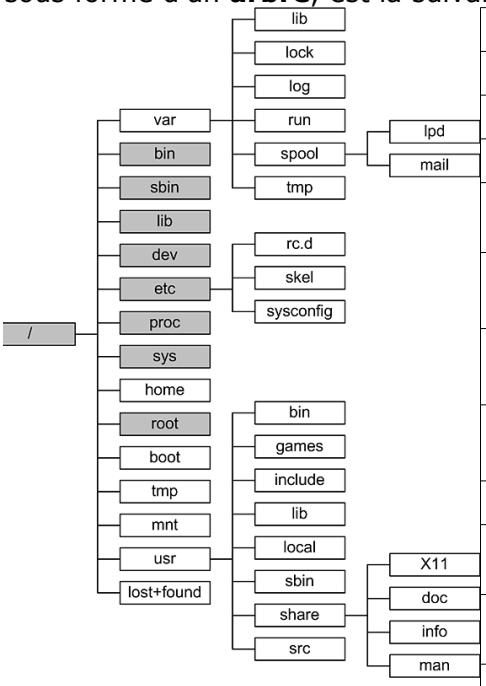
Une commande est alors composée d'un nom suivi ou non d'options. Les commandes spécifiques au système (que ce soit Ubuntu ou Windows) sont listées par la commande `help`

Pour effacer la console par exemple : `clear` (UNIX) ou `cls` (Windows).

Le système de fichiers Linux :

Dans la "philosophie UNIX" dont dérive Linux, tout est fichier : toutes les ressources de la machine, les documents mais aussi les périphériques, peuvent être accédés grâce à leur chemin d'accès dans le système de fichiers.

L'arborescence du système de fichiers, c'est à dire la hiérarchie des répertoires et fichiers présentée sous forme d'un **arbre**, est la suivante :

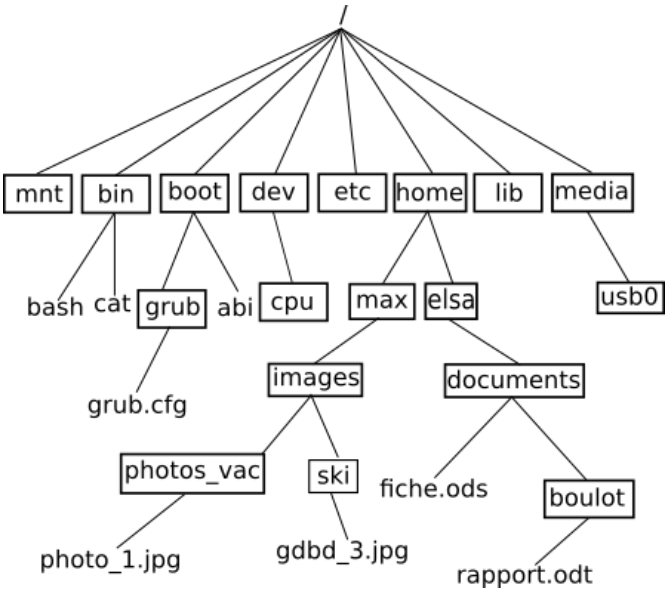


Répertoire	Description
/	Répertoire "**racine**", point d'entrée du système de fichiers
/boot	Répertoire contenant le noyau Linux et l'amorceur
/bin	Répertoire contenant les exécutables système de base
/dev	Répertoire contenant des fichiers spéciaux nommés devices qui permettent le lien avec les périphériques de la machine
/etc	Répertoire contenant les fichiers de configuration du système
/home	Répertoire contenant les fichiers personnels des **utilisateurs** (un **sous-répertoire par utilisateur**)
/media ou / run/media	Répertoire contenant les « points de montage » des médias usuels : cd, dvd, disquette, clef usb
/root	Répertoire personnel de **l'administrateur**
/tmp	Répertoire contenant des fichiers temporaires utilisés par certains programmes
/usr	Répertoire contenant les exécutables des programmes (/usr/bin et /usr/sbin), la documentation (/usr/doc).
/var	Répertoire contenant les fichiers qui servent à la maintenance du système (les fichiers de journaux notamment dans /var/log)

(Et non ici , pas de "c:\" ou "d:\" : c'est vraiment une autre façon de voir les choses...)

Germain BECKER & Sébastien POINT

Une autre représentation possible est la suivante :



Dans le schéma ci-contre on trouve des répertoires (noms entourés d'un rectangle, exemple : "home") et des fichiers (uniquement des noms "grub.cfg").

À noter : les extensions des noms de fichiers, par exemple le "cfg" de "grub.cfg", ne sont pas obligatoires dans les systèmes de type "UNIX", par exemple, "bash" est bien un nom de fichier et il n'a pas d'extension.

On parle d'arborescence, car ce système de fichier ressemble à un arbre à l'envers.

Comme vous pouvez le constater, la **base** de l'arbre s'appelle la **racine** de l'arborescence et se représente par un "**/**"

Chemin absolu ou chemin relatif ?

Pour indiquer la position d'un fichier (ou d'un répertoire) dans l'arborescence, il existe 2 méthodes : indiquer un **chemin absolu** ou indiquer un **chemin relatif**. Le chemin absolu doit indiquer "le chemin" depuis la racine. Par exemple le chemin absolu du fichier fiche.ods sera : /home/elsa/documents/fiche.ods

Remarquez que nous démarrons bien de la racine / (attention les symboles de séparation sont aussi des /)

Il est possible d'indiquer le chemin non pas depuis la racine, mais depuis un répertoire quelconque, nous parlerons alors de chemin relatif :

Le chemin relatif permettant d'accéder au fichier "photo_1.jpg" depuis le répertoire "max" est : "images/photo_vac/photo_1.jpg"

Remarquez l'absence du / au début du chemin (c'est cela qui nous permettra de distinguer un chemin relatif et un chemin absolu).

Imaginons maintenant que nous désirions indiquer le chemin relatif pour accéder au fichier "gdbd_3.jpg" depuis le répertoire "photos_vac".

Comment faire ?

Il faut "remonter" d'un "niveau" dans l'arborescence pour se retrouver dans le répertoire "images" et ainsi pouvoir repartir vers la bonne "branche". Pour ce faire il faut utiliser 2 points : ..

"../ski/gdbd_3.jpg"

Il est tout à fait possible de remonter de plusieurs "crans" : "../../" depuis le répertoire "photos_vac" permet de "remonter" dans le répertoire "max"

À faire vous-même 1 :

En vous basant sur l'arborescence ci-dessus, déterminez le chemin absolu permettant d'accéder au fichier : "cat"

et "rapport.odt"

Toujours en vous basant sur l'arborescence ci-dessus, déterminez le chemin relatif permettant d'accéder au fichier :

"rapport.odt" depuis le répertoire "elsa"

"fiche.ods" depuis le répertoire "boulot"

Le langage Bash

Et maintenant, vos première commandes (enfin presque)… Elles s'écrivent dans une syntaxe portant le nom de langage **Bash**.

Les systèmes de type "UNIX" sont des systèmes "multi-utilisateurs" : chaque utilisateur possède son propre compte. Chaque utilisateur possède un répertoire à son nom, ces répertoires personnels se situent traditionnellement dans le répertoire "home". Dans l'arborescence ci-dessus, nous avons 2 utilisateurs : "max" et "elsa". Par défaut, quand un utilisateur ouvre une console, il se trouve dans son répertoire personnel. Dans l'image de la première console , nous avons un "moi@mongroupe ~ $" (au passage, on

appelle cela "l'invite de commande"), le "~" (caractère "**tilde**") signifie que l'on se trouve **actuellement** dans le **répertoire personnel de l'utilisateur courant**, autrement dit dans le répertoire de chemin absolu "/home/moi" (puisque l'utilisateur courant est "moi"). Le répertoire "où l'on se trouve actuellement" est appelé "répertoire courant". L'invite de commande vous indique à tout moment le répertoire courant : "moi@mongroupe ~/Documents $" vous indique que vous êtes dans le répertoire "Documents" qui se trouve dans le répertoire "moi" qui se trouve dans le répertoire "home" (chemin absolu : "/home/moi/Documents")

Comme son nom l'indique, on invite donc l'utilisateur à entrer une commande en sachant qu'il se "trouve" actuellement dans tel ou tel répertoire.

Chaque ligne de commande doit être suivie de l'appui sur la touche **ENTRÉE** pour être exécutée.

Attention : les systèmes de type "UNIX" sont "sensibles à la casse" (il faut différencier les caractères majuscules et les caractères minuscules) : le répertoire "Documents" et le répertoire "documents" sont 2 répertoires différents.

En faisant suivre le nom d'une commande par `-h` ou `--help` on obtient des détails sur l'utilisation et les options de la commande. (On peu t également faire précéder la commande par `man` pour manuel)

La commande cd

La commande "cd" permet de changer le répertoire courant. Il suffit d'indiquer le chemin (relatif ou absolu) qui permet d'atteindre le nouveau répertoire :

Par exemple (en utilisant l'arborescence ci-dessus) :

si le répertoire courant est le répertoire "elsa" et que vous "voulez vous rendre" dans le répertoire "documents", il faudra saisir la commande : "cd documents" (relatif) ou "cd /home/elsa/documents" (absolu)

si le répertoire courant est le répertoire "photos_vac" et que vous "voulez vous rendre" dans le répertoire "ski", il faudra saisir la commande : "cd ../ski" (relatif) ou "cd /home/max/images/ski" (absolu)

si le répertoire courant est le répertoire "boulot" et que vous "voulez vous rendre" dans le répertoire "documents", il faudra saisir la commande : "cd .." (relatif) ou "cd /home/elsa/documents" (absolu)

Pour revenir à son répertoire personnel quelque soit l'endroit de l'arborescence où l'on se trouve, il faudra saisir la commande : "cd"

À faire vous-même 2 :

Toujours en utilisant l'arborescence ci-dessus, quelle est la commande à saisir si le répertoire courant est le répertoire "home" et que vous "voulez vous rendre" dans le répertoire "boulot" (vous utiliserez d'abord un chemin absolu puis un chemin relatif)

La commande ls

La commande "ls" permet de lister le contenu du répertoire courant.

À faire vous-même 3 :

Après avoir ouvert une console, utilisez la commande ls depuis votre répertoire personnel.

Pas de Linux chez vous ? Pas de problème : vous pouvez faire de la ligne de commande...en ligne !

Voila un lien vers un émulateur de ligne de commande Linux fonctionnant dans un navigateur (et codé en JavaScript) :

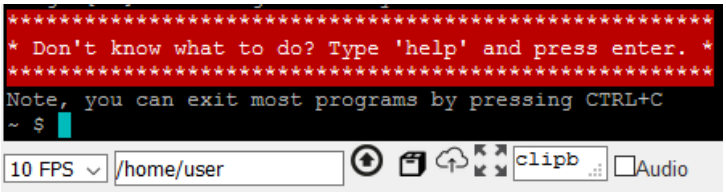
http://s-macke.github.io/jor1k/demos/main.html?user=XjgQbmJywo&cpu=asm&n=1&relayURL=wss%3A%2F%2Frelay.widgetry.org%2F

(vous pouvez aussi faire une recherche avec les mots suivants :

jor1k: OpenRISC OR1K Javascript Emulator Running Linux With Network Support

Par défaut, vous vous connectez en tant qu'administrateur (compte root), mais vous pouvez créer un compte utilisateur, même si ce n'est pas indispensable.

Vous devriez voir quelque chose comme ceci :



vous remarquez que vous êtes dans votre répertoire courant d'utilisateur indiqué par :" `~ $` "
(qui correspond en chemin absolu à " `/home/user` "
vérifier le en utilisant la commande" `pwd` ")

En utilisant la commande "ls" sur l'émulateur il ne doit rien se passer …

Normal votre répertoire ici est vide…

Remontez alors d'un cran dans l'arborescence et utilisez la commande "`ls`".

Remontez encore d'un cran dans l'arborescence et réutilisez la commande "ls" (vous êtes normalement à la racine "/"de l'arborescence) et vous devez avoir quelque chose qui ressemble à ceci :

```
~ $ ls
~ $
~ $ cd ..
/home $ ls
user
/home $ cd ..
/ $ ls
bin      etc      lib      proc     sbin     tmp      var
dev      home     linuxrc  root     sys      usr
/ $
```

La commande "mkdir"

La commande "mkdir" permet de créer un répertoire dans le répertoire courant. La commande est de la forme "mkdir nom_du_répertoire"

Remarque : il est préférable de ne pas utiliser de caractères accentués dans les noms de répertoire (ou de fichier). Il en est de même pour les espaces (à remplacer par des caractères tirets bas "_")

À faire vous-même 4 :

Après avoir ouvert une console (ou être revenu dans votre répertoire personnel commande "cd"), utilisez la commande "mkdir" afin de créer un répertoire "test_nsi" dans votre répertoire personnel.

Vérifier que le répertoire à bien été créer.

Vous devez avoir quelque chose qui ressemble à ceci :

```
/ $ cd home
/home $ cd user
~ $ mkdir test_nsi
~ $ ls
test_nsi
~ $
```

La commande "rm"

La commande "rm" permet de supprimer un fichier ou un répertoire. La commande est de la forme

"rm nom_du_répertoire_ou_nom_du_fichier"

La plupart des commandes UNIX peuvent être utilisées avec une ou des options. Par exemple, pour supprimer un répertoire non vide, il est nécessaire d'utiliser la commande "rm" avec l'option "-r" : "rm -r nom_du_répertoire"

La commande "touch"

La commande "touch" permet de créer un fichier vide. La commande est de la forme

"touch nom_du_fichier_à_créer"

La commande "cp"

La commande "cp" permet de copier un fichier. La commande est de la forme

"cp /répertoire_source/nom_fichier_à_copier /répertoire_destination/nom_fichier"

À noter : le nom du fichier "destination" n'est pas obligatoirement le même que le nom du fichier "source" (on peut avoir "cp fic.txt info/fiche.txt")

À faire vous-même 5 :

Placez-vous dans le répertoire "test_nsi" créé au "À faire vous-même 4". Créez un fichier "test.txt". Créez un répertoire "doc". Copiez le fichier "test.txt" dans le répertoire "doc". Effacez le répertoire doc (et son contenu).

Gestion des utilisateurs et des groupes :

Les systèmes de type "UNIX" sont des systèmes **multi-utilisateurs**, plusieurs utilisateurs peuvent donc partager un même ordinateur, chaque utilisateur possédant un environnement de travail qui lui est propre.

Chaque utilisateur possède certains **droits** lui permettant d'effectuer certaines opérations et pas d'autres. Le système d'exploitation permet de gérer ces droits très finement. Un utilisateur un peu particulier est autorisé à modifier tous les droits : ce "**super utilisateur**" est appelé "**administrateur**" ou "**root**".

L'administrateur pourra donc attribuer ou retirer des droits aux autres utilisateurs. Au lieu de gérer les utilisateurs un par un, il est possible de créer des groupes d'utilisateurs. L'administrateur attribue des droits à un groupe au lieu d'attribuer des droits particuliers à chaque utilisateur.

Comme nous venons de le voir, chaque utilisateur possède des droits qui lui ont été octroyés par le "super utilisateur". Nous nous intéresserons ici uniquement aux droits liés aux fichiers, mais vous devez savoir qu'il existe d'autres droits liés aux autres éléments du système d'exploitation ((imprimante, installation de logiciels...).

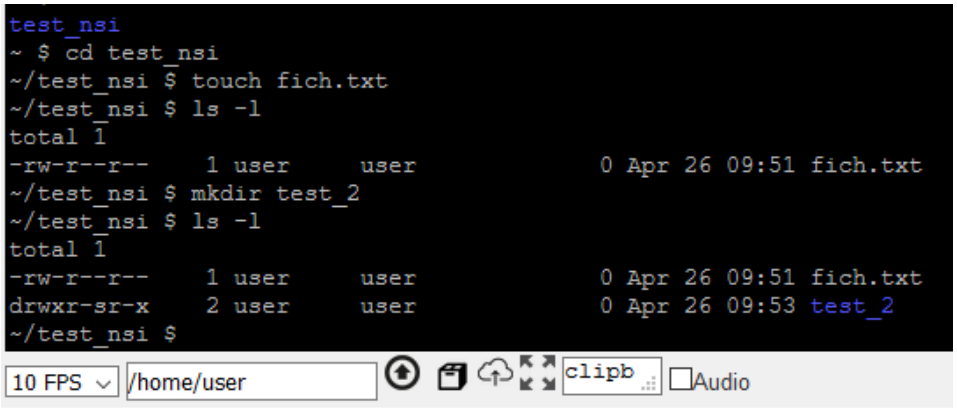
Les fichiers et les répertoires possèdent **3 types de droits** :

- les droits en lecture (symbolisés par la lettre "**r**") : est-il possible de lire le contenu de ce fichier
- les droits en écriture (symbolisés par la lettre "**w**") : est-il possible de modifier le contenu de ce fichier
- les droits en exécution (symbolisés par la lettre "**x**") : est-il possible d'exécuter le contenu de ce fichier (quand le fichier du code exécutable)

Il existe **3 types d'utilisateurs** pour un fichier ou un répertoire :

- le propriétaire du fichier (par défaut c'est la personne qui a créé le fichier), il est symbolisé par la lettre "**u**"
- un fichier est associé à un groupe, tous les utilisateurs appartenant à ce groupe possèdent des droits particuliers sur ce fichier. Le groupe est symbolisé par la lettre "**g**"
- tous les autres utilisateurs (ceux qui ne sont pas le propriétaire du fichier et qui n'appartiennent pas au groupe associé au fichier). Ces utilisateurs sont symbolisés la lettre "**o**"

Il est possible d'utiliser la commande "ls" avec l'option "-l" afin d'avoir des informations supplémentaires.



Ici après s'être placé dans le répertoire test_nsi un fichier fich.txt (vide) a été créé et un nouveau répertoire test_2 aussi.

Prenons la première ligne (suivant la dernière commande "ls -l":

```
-rw-r--r--    1 user    user          0 Apr 26 09:51 fich.txt
```

Lisons cette ligne de gauche à droite :

le premier symbole "-" signifie que l'on a affaire à un fichier, dans le cas d'un répertoire, nous aurions un "d" (voir la 2ème ligne)

les 3 symboles suivants "rw-"donnent les droits du propriétaire du fichier : lecture autorisée (r), écriture autorisée (w), exécution interdite (- à la place de x)

les 3 symboles suivants "r--"donnent les droits du groupe lié au fichier : lecture autorisée (r), écriture interdite (- à la place de w), exécution interdite (- à la place de x)

les 3 symboles suivants "r--"donnent les droits des autres utilisateurs : lecture autorisée (r), écriture interdite (- à la place de w), exécution interdite (- à la place de x)

le caractère suivant "1" donne le nombre de liens (nous n'étudierons pas cette notion ici)

le premier "user" représente le nom du propriétaire du fichier

le second "user" représente le nom du groupe lié au fichier

le "0" représente la taille du fichier en octet (ici notre fichier est vide)

"Apr 26 09:51" donne la date et l'heure de la dernière modification du fichier

"fich.txt" est le nom du fichier

Prenons la deuxième ligne :

```
drwxr-sr-x    2 user    user          0 Apr 26 09:53 test_2
```

Lisons cette ligne de gauche à droite :

le premier symbole "d" signifie que l'on a un répertoire

les 3 symboles suivants "rwx"donnent les droits du propriétaire du répertoire : lecture du contenu du répertoire autorisée (r), modification du contenu du répertoire autorisée (w), il est possible de parcourir le répertoire (voir le contenu du répertoire) (x)

les 3 symboles suivants "r-x"donnent les droits du groupe lié au répertoire : modification du contenu du répertoire interdite (- à la place de w)

les 3 symboles suivants "r-x"donnent les droits des autres utilisateurs : modification du contenu du répertoire interdite (- à la place de w)

le caractère suivant "2" donne le nombre de liens (nous n'étudierons pas cette notion ici)

le premier "user" représente le nom du propriétaire du répertoire

le second "user" représente le nom du groupe lié au répertoire

le "0" représente la taille du répertoire en octets

"Apr 26 09:53" donne la date et l'heure de la dernière modification du contenu du répertoire

"test_2" est le nom du répertoire

À faire vous-même 6 :

Analysez les lignes du résultat de la commande "ls -l" ci-dessous

```
-rw-r--r--    1 user    user          0 Apr 25 08:41 photo.jpg

-rwxr-xr-x    1 user    user          0 Apr 27 10:51 test.exe
```

La commande "chmod"

Il est important de ne pas perdre de vu que l'utilisateur "root" a la possibilité de modifier les droits de tous les utilisateurs.

Le propriétaire d'un fichier peut modifier les permissions d'un fichier ou d'un répertoire à l'aide de la commande "**chmod**". Pour utiliser cette commande, il est nécessaire de connaître certains symboles :

- les symboles liés aux utilisateurs : "u" correspond au propriétaire, "g" correspond au groupe lié au fichier (ou au répertoire), "o" correspond aux autres utilisateurs et "a" correspond à "tout le monde" (permet de modifier "u", "g" et "o" en même temps)
- les symboles liés à l'ajout ou la suppression des permissions : "+" on ajoute une permission, "-" on supprime une permission, "=" les permissions sont réinitialisées (permissions par défaut)
- les symboles liés aux permissions : "r" : lecture, "w" : écriture, "x" : exécution.

La commande "chmod" à cette forme :

chmod [u g o a] [+ - =] [r w x] nom_du_fichier

par exemple

```
chmod o+w toto.txt
```

attribuera la permission "écriture" pour le fichier "toto.txt" "aux autres utilisateurs"

Il est possible de combiner les symboles :

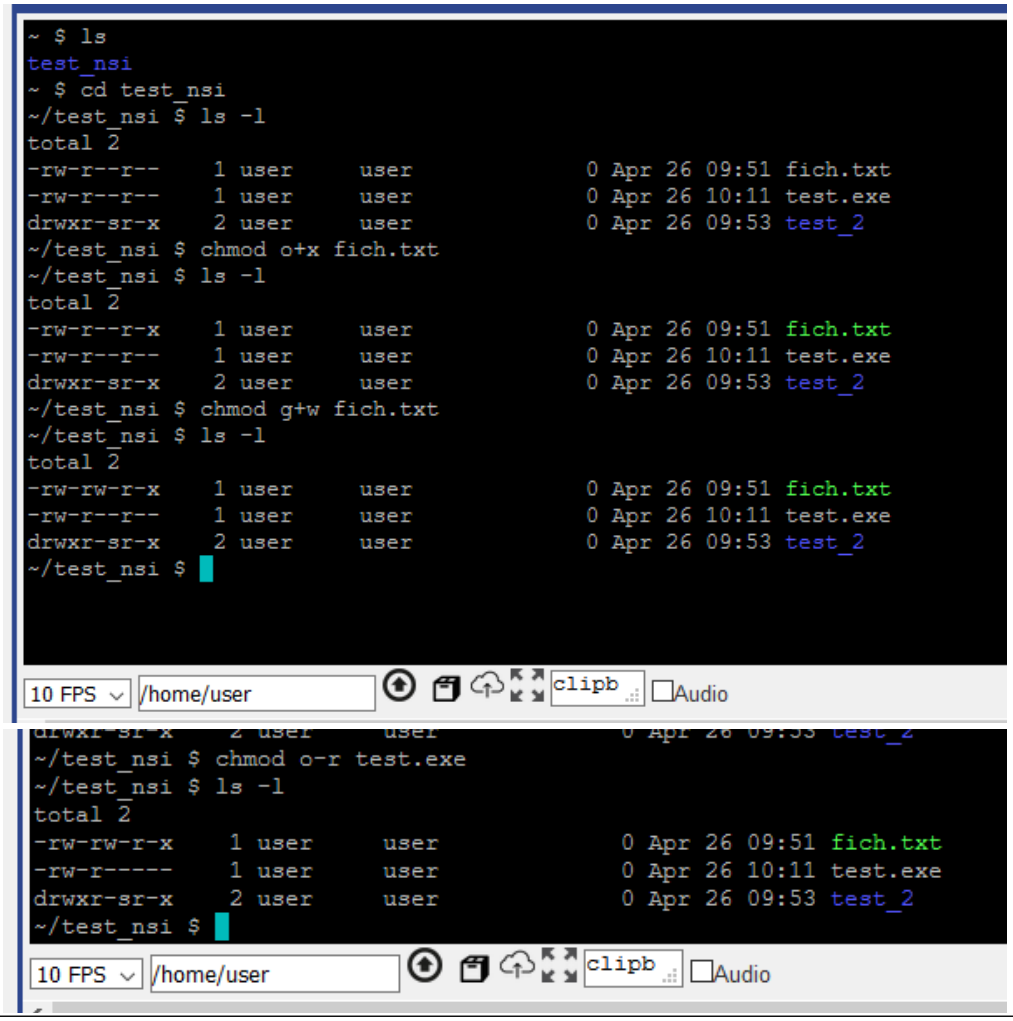
```
chmod g-wx toto.txt
```

La commande "chmod" ci-dessus permet de supprimer la permission "écriture" et la permission "exécution" pour le fichier "toto.txt" "au groupe lié au fichier"

Une fois de plus, "root" a tous les droits sur l'ensemble des fichiers et des répertoires, il peut donc utiliser la commande "chmod" sur tous les répertoires et tous les fichiers.

À faire vous-même 7 :

Analysez attentivement l'enchaînement de commandes suivantes :



À faire vous-même 8 :

Créez un répertoire "test_nsi2" dans votre répertoire personnel. Placez-vous dans le répertoire "test_nsi2". Créez un fichier "titi.txt", vérifiez les permissions associées à ce fichier. Modifiez les permissions associées au fichier "titi.txt" afin que les "autres utilisateurs" aient la permission "écriture"

Pour aller plus loin : Lire et écrire dans un fichier :

Créer dans votre zone personnelle un fichier nommé test.txt

pour envoyer du contenu dans un fichier, on utilise la commande :

`echo "blablabla" > nom du fichier`

Utilisez cette commande pour placer quelques mots dans le fichier test.txt

à l'aide de la commande adéquate, afficher et noter les droits dont vous disposez sur le fichier test.txt

ouvrer ce fichier avec la commande `nano test.txt`

Nano est un éditeur de texte accessible dans la console. Pour le lancer, il suffit de taper "`nano file1.txt`" pour ouvrir le fichier file1.txt. Certains raccourcis sont notés en bas de la fenêtre.

Le symbole ^ correspond à la touche contrôle. Le symbole M correspond à la touche alt.

En particulier : ^X ou la combinaison des touches ctrl + x permet de quitter Nano

entrez la commande qui permet de vous retirer les droits en lecture et en écriture sur le fichier test.txt

essayez maintenant d'afficher le contenu du fichier test.txt; essayez également de modifier son contenu. Que constate-t-on ?

Redonnez-vous les droits de lecture et d'écriture sur le fichier test.txt, et vérifiez le changement.

supprimez enfin le fichier test.txt ...

Formulaire et informations complémentaire.

Comparaison Unix et Windows :

Les commandes spécifiques au système (que ce soit Ubuntu ou Windows) sont listées par la commande `help` . Pour effacer la console par exemple : `clear` (UNIX) ou `cls` (Windows).

UNIX	Windows	Action
`pwd`	`cd`	Affiche le répertoire courant (pratique si on ne sait plus ou on est...)
`cd`	`cd %HOMEPATH%`	Positionne au répertoire utilisateur
`cd ..`	`cd ..`	Positionne au répertoire parent
`cd /`	`cd \`	Positionne à la racine

De manière générale : `cd` suivi d'un chemin positionne au bout du chemin.

Le chemin peut être relatif (au répertoire courant) ou absolu (commençant par `/` ou `C:\`).

`..` est le répertoire précédent, `.` le répertoire courant.

La touche de tabulation permet une complétion automatique des noms de fichiers.

UNIX	Windows	Action
`ls`	`cdir`	Liste les fichiers du répertoire courant

En faisant suivre le nom d'une commande par `-h` ou `--help` ou `-man` pour manuel (UNIX) ou `/?` (Windows), on obtient des détails sur leurs utilisations et options. Par exemple, pour obtenir une liste récursive des fichiers du répertoire courant et de ses sous-répertoires : `ls -R` ou `dir /S`.

Autres actions possibles sur les dossiers :

UNIX	Windows	Action
`mkdir monRep`	`md monRep`	Crée le répertoire monRep
`mv monRep rep`	`ren monRep rep`	Renomme le répertoire monRep en rep
`rmdir monRep`	`rd monRep`	Supprime le répertoire vide monRep

Visualisation et manipulation des fichiers

Quelques actions possibles sur les fichiers :

UNIX	Windows	Action
`touch fichier.txt`	Pas d'équivalent?	Crée (le fichier fichier.txt dans le répertoire courant)
`cat fichier.txt`	`type fichier.txt`	Affiche le contenu
`mv fichier.txt test.txt`	`ren fichier.txt test.txt`	Renomme fichier.txt en text.txt
`mv test.txt monRep`	`move test.txt monRep`	Déplace test.txt dans monRep (répertoire)
`cp test.txt monRep`	`copy test.txt monRep`	Copie test.txt dans monRep (répertoire)
`rm test.txt`	`del test.txt`	Supprime test.txt

On peut lancer des applications depuis la ligne de commande.

Exemples : `gedit fichier.txt` (UNIX) ou `notepad fichier.txt` (Windows)

Droits et permissions d'accès aux fichiers (pour aller plus loin)

Sous Windows, il n'existe pas vraiment d'équivalent aux lignes de commandes proposées ci-dessous. On ne considère donc ici que les fichiers d'un système de type UNIX. On les distingues par :

4 types de fichier :	3 catégories d'utilisateurs :	3 types de droit :
- Ordinaire	u Propriétaire (user)	r Lecture (read)
d Répertoire (directory)	g Groupe (group)	w Écriture (write)
l Lien symbolique (link)	o Autres (others)	x Exécution (execute)
c ou b Spécial		

On peut visualiser ces informations par un listing détaillé : `ls -l`.

Les trois types de droit pouvant ou non être accordés indépendamment, ils génèrent 23 = 8 possibilités, que l'on numérote comme s'ils écrivaient un nombre binaire :

---	--x	-w-	-wx	r--	r-x	rw-	rwx
0	1	2	3	4	5	6	7

En juxtaposant les droits des 3 catégories d'utilisateurs, on obtient alors un nombre écrit en octal.

Exemple : Un répertoire d, pour lequel le propriétaire a tous les droits rwx, que le groupe n'a juste pas le droit de modifier r-x et qui ne laisse le droit qu'en lecture aux autres r-- est de type drwxr-xr--. Ces droits se lisent en octal : 754.

On peut changer les droits en donnant sa valeur en octal (nouveauté):

`chmod 777 fichier.txt` donne tous les droits à tout le monde sur le fichier.

On peut modifier les droits relativement à certains utilisateurs :

`chmod o - w fichier.txt` enlève le droit d'écriture aux autres utilisateurs.

(o peut être g ou u, - peut être + et w peut être r ou x)

Enfin si vous êtes arrivés jusqu'ici petite surprise sur l'émulateur que nous avons utilisés :

http://s-macke.github.io/jor1k/demos/main.html?user=XjgQbmJywo&cpu=asm&n=1&relayURL=wss%3A%2F%2Frelay.widgetry.org%2F

Ce type d'émulateur en JS n'a pas forcément beaucoup d'utilité en programmation, mais il permet de faire fonctionner des applications ou logiciels obsolètes en implémentant une version compatible.

Essayer la commande `help` et testez certaines des applications proposées (attention l'implémentation peu simuler des systèmes datant des années 1990 (voire 80…) …

Sources :

https://doc.ubuntu-fr.org/arborescence

https://cache.media.eduscol.education.fr/file/NSI/76/7/RA_Lycee_G_NSI_arch_systemes_approfondis_1170767.pdf

https://cache.media.eduscol.education.fr/file/NSI/76/8/RA_Lycee_G_NSI_arch_systemes_utilisateurs_1170768.pdf

http://nsivaugelas.free.fr/premiere/archi_s_e.php

https://pixees.fr/informatiquelycee/n_site/nsi_prem_os_intro.html

Synthèse de cours de NSI :Mickaël BARRAUD

https://interstices.info/a-quoi-sert-un-systeme-dexploitation/