

Linux Utilisation

Module 05 – Utiliser des fichiers sous Linux



1

Utiliser des fichiers sous Linux



Objectifs

- Comprendre les types de fichiers Unix/Linux
- Savoir lire du texte d'un fichier en intégralité ou partiellement
- Comprendre et créer des liens symboliques et liens physiques

*Savoir se situer, se déplacer et
utiliser fichiers et dossiers.*



2

Types de fichiers et de données



3

La commande `ls -l` indique entre autre les types de fichiers, indiqué par le premier caractère.



```
user30@deb:~$ ls -ld $HOME
-rw-r--r-- 1 user30 user30  0 juil.  2 10:47 contacts.txt
drwxr-xr-x 2 user30 user30 4096 juil.  3 11:12 dirttest01
-rw----- 1 user30 user30  586 avril 16 17:26 Edition
lrwxrwxrwx 1 user30 user30   7 juil. 20 14:55 edition1.txt -> Edition
```



4

Sous Unix/Linux, tout est fichier. Il existe plusieurs type de fichiers.

Type	Description	Type	Description
-	Fichier standard	b	Périphérique de type bloc
d	Répertoire	c	Périphérique de type caractère
l	Lien symbolique	p	Tube nommé (pipe)
		s	Socket Unix



- Sous Linux, du point de vue système il n'y a pas de notions d'extensions comme dans le monde Windows.
- Pour le système ce qui compte en premier c'est le type de fichier, puis le type de données, les permissions d'accès...
- Les extensions ne sont utilisées que pour simplifier la manipulation des fichiers par les humains.



- La commande **file** est utile pour connaître le **type de données** d'un fichier.

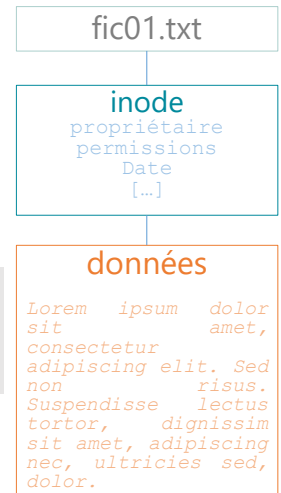
```
user30@deb:~$ file Edition
Edition: ASCII text
user30@deb:~$ file edition1.txt
edition1.txt: symbolic link to Edition
user30@deb:~$ file dirtest01/
dirtest01/: directory
user30@deb:~$ file /bin/bash
/bin/bash: ELF 64-bit LSB pie executable, x86-64, version 1 [...]
user30@deb:~$ file /usr/lib/libDeployPkg.so.0.0.0
/usr/lib/libDeployPkg.so.0.0.0: ELF 64-bit LSB shared object [...]
user30@deb:~$ file /dev/sda1
/dev/sda1: block special (8/1)
```



Un fichier sous Linux est généralement composé de trois parties.

- Nom du fichier
- Inode** / métadonnées
- Données**

```
user30@deb:~$ ls -li
10557 -rw-r--r-- 1 user30 user30 29 juil. 21 09:36 contact
35270 drwxr-xr-x 3 user30 user30 4096 juil. 20 12:38 perso
1486 -rw-r--r-- 1 user30 user30 77 juil. 21 14:40 tel2018
```



L'inode contient des informations telles que :

- Le **type** de fichier
- Les **identifiants** de l'utilisateur et du **groupe** propriétaire du fichier
- Les **permissions**
- Les **dates** et **heure** de dernière modification et d'accès
- La taille du fichier
- Les pointeurs vers les données du fichier.



L'inode **ne contient pas** le nom du fichier.

Lire du texte avec Linux



`cat <fichier> <fichier2>`



```
user30@deb:~$ cat contacts.txt
Anne-Sophie      548
Romain           351
Estelle          350
user30@deb:~$ cat entete.txt
PRENOM           TEL
-----          ---
user30@deb:~$ cat entete.txt contacts.txt
PRENOM           TEL
-----          ---
Anne-Sophie      548
Romain           351
Estelle          350
```



`cat <fichier>`



```
user30@deb:~$ cat > newcontacts.txt
Sarah            255
Mehdi            548
Fred             999
^D
```



more [-ds] <fichier>



```
user30@deb:~$ more /etc/passwd
mathieu:x:1000:1000:mathieu,,,:/home/mathieu:/bin/bash
user01:x:1031:1031:./home/user01:/bin/bash
user02:x:1032:1032:./home/user02:/bin/bash
user03:x:1033:1033:./home/user03:/bin/bash
user04:x:1034:1034:./home/user04:/bin/bash
user05:x:1035:1035:./home/user05:/bin/bash
user06:x:1036:1036:./home/user06:/bin/bash
user07:x:1037:1037:./home/user07:/bin/bash
user08:x:1038:1038:./home/user08:/bin/bash
user09:x:1039:1039:./home/user09:/bin/bash
--Plus-- (65%)
```



less [-is] <fichier>




```
user30@deb:~$ less /etc/passwd
user25:x:1055:1055:./home/user25:/bin/bash
user26:x:1056:1056:./home/user26:/bin/bash
user27:x:1057:1057:./home/user27:/bin/bash
user28:x:1058:1058:./home/user28:/bin/bash
user29:x:1059:1059:./home/user29:/bin/bash
user30:x:1060:1060:./home/user30:/bin/bash
user31:x:1061:1061:./home/user31:/bin/bash
user32:x:1062:1062:./home/user32:/bin/bash
(END)
```




Les commandes internes de **less**

[space]	faire défiler une page	gg	se déplacer en fin de fichier
e	faire défiler une ligne vers le bas	q	quitter
y	faire défiler une ligne vers le haut	/motif	chercher "motif" dans le texte à l'écran
h	accéder à l'aide interne	n	se déplacer à la prochaine occurrence de "motif"
G	se déplacer en début de fichier		



```
user30@deb:~$ less /etc/passwd
user30:x:1060:1060::/home/user30:/bin/bash
user31:x:1061:1061::/home/user31:/bin/bash
user32:x:1062:1062::/home/user32:/bin/bash
/bash
```

head **[-vn nb]** <fichier> <fichier2>



```
user30@deb:~$ head -n 4 liste.txt
Première ligne
ligne numéro 2
quatrième ligne
user30@deb:~$
```


`tail [-fvrn nb] <fichier> <fichier2>`



```
user30@deb:~$ tail -n 3 liste.txt
```

```
10
```

```
douzième et dernière ligne
```

```
user30@deb:~$
```



```
root@deb:~# tail -n2 -f /var/log/syslog
```

```
Jul 21 11:15:58 LinuxUtil systemd[1]: Started Cleanup of Temporary Directories.  
Jul 21 11:17:01 LinuxUtil CRON[498]: (root) CMD ( cd / && run-parts --report  
/etc/cron.hourly)
```



`wc [-clmw] <fichier>`



```
user30@deb:~$ wc -l liste.txt
```

```
12 liste.txt
```

```
user30@deb:~$ wc -w liste.txt
```

```
17 liste.txt
```

```
user30@deb:~$ wc -m liste.txt
```

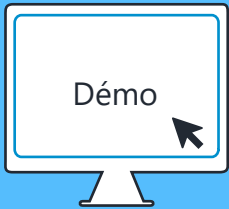
```
107 liste.txt
```

```
user30@deb:~$ wc -c liste.txt
```

```
113 liste.txt
```




Lire du texte avec Linux




Démon

- Lire des fichiers
-




20

Lire du texte avec Linux



TP

- Lire des fichiers
-



21

Les liens sous Unix/Linux : Liens Symboliques



22

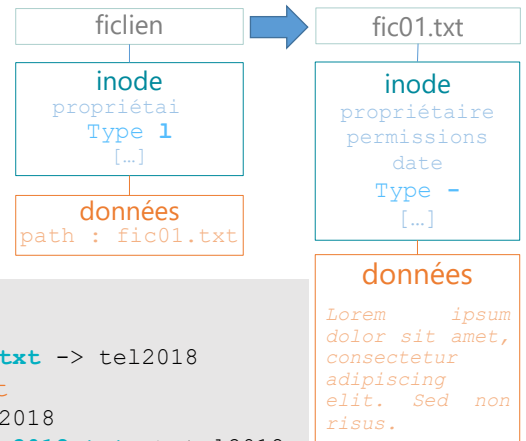
ln [-s] <source> <cible>

- Créer un lien du fichier ou dossier cible vers le fichier ou dossier source
- Il existe deux types de liens : liens **symboliques** & liens **physiques**
Les liens jouent sur les composantes des fichiers : inodes et noms



23

`ln -s <ficsource> <ficcible>`
`ln -s <ficsource> </chemin/vers/cible/>`
`ln -s <dossiersource> <dossiercible>`



```
user30@deb:~$ ln -s tel2018 Tel-2018.txt
user30@deb:~$ ls -l Tel-2018.txt
lrwxrwxrwx 1 user1 user1 7 juil. 22 Tel-2018.txt -> tel2018
user30@deb:~$ ls -li tel2018 Tel-2018.txt
1486 -rw-r--r-- 1 user1 user1 77 juil. 21 tel2018
4330 lrwxrwxrwx 1 user1 user1 7 juil. 22 Tel-2018.txt -> tel2018
```



```
user30@deb:~$ cd perso
user30@deb:~/perso$ ln -s ../tel2018 .
user30@deb:~/perso$ ls -l tel2018
lrwxrwxrwx 1 user30 user30 7 juil. 21 tel2018 -> ../tel2018
```

Ou

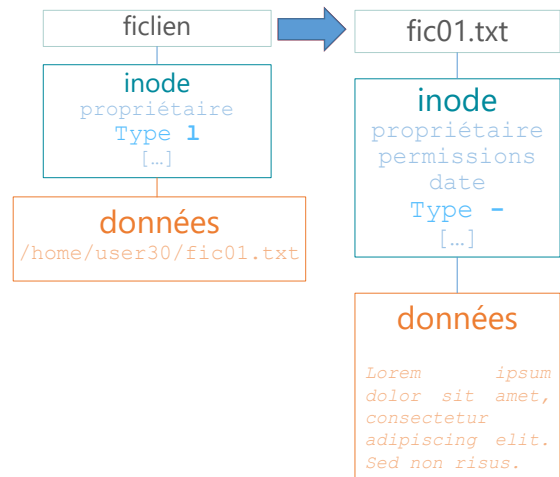
```
user30@deb:~$ ln -s $HOME/tel2018 $HOME/perso/tel2018
user30@deb:~$ ls -l perso/tel2018
lrwxrwxrwx 1 user30 user30 7 juil. 21 tel2018 -> /home/user30/tel2018
```

Équivalent à

```
ln -s /home/user30/tel2018 /home/user30/perso/tel2018
```



- La suppression d'un lien symbolique ne supprime que le mini fichier contenant le nom du fichier source. Le fichier source lui-même n'est nullement altéré.
- Inversement, si on supprime un fichier source, le lien symbolique est cassé. On peut ici faire l'analogie avec un lien mort sur un site web et la fameuse erreur 404.
- Si le fichier source est recréé, le lien refonctionne.



AVANTAGES

- Pas de contingence par rapports aux partitions (filesystem)
- Facilité à reconnaître un lien symbolique
- Lien possible sur des répertoires
- Utilisation transparente

CONTRAINTES

- Si le fichier d'origine est supprimé, le lien existe toujours mais il est invalide.
- Risque de rendre le lien invalide en cas de déplacement
- Utilisation de deux inodes
- Théoriquement plus long à accéder aux données : lecture de deux fichiers

Les liens sous Unix/Linux : Liens Physiques



28

Lire du texte avec Linux
Liens physiques : fonctionnement

`ln <ficsource> </chemin/vers/ficcible>`



```

user30@deb:~$ ln tel2020 Tel-2020.txt
user30@deb:~$ ls -l tel2020 Tel-2020.txt
-rw-r--r-- 2 user30 user30 81 juil. 21 tel2020
-rw-r--r-- 2 user30 user30 81 juil. 21 Tel-2020.txt

```



```

user30@deb:~$ ln tel2020 Tel-2020.txt
user30@deb:~$ ls -li tel2020 Tel-2020.txt
1487 -rw-r--r-- 2 user30 user30 81 juil. 21 tel2020
1487 -rw-r--r-- 2 user30 user30 81 juil. 21 Tel-2020.txt

```

inode
N° : 1487
nb lien : 2
propriétaire
permissions
Type -
[...]

données
*Lorem ipsum dolor
sit amet,
consectetur
adipiscing elit.
Sed non risus.*



29

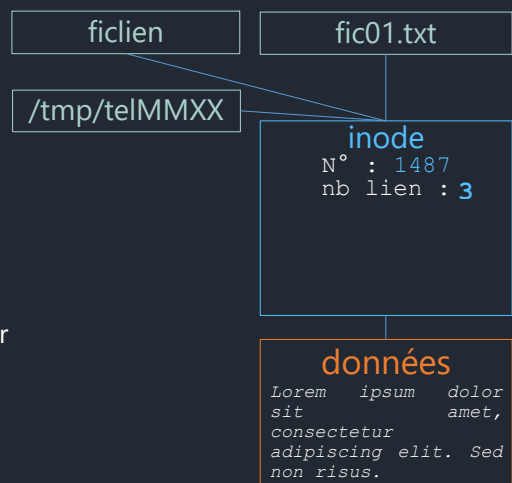
```
user30@deb:~$ ln Tel-2020.txt /srv/telMMXX
user30@deb:~$ ls -li tel2020 Tel-2020.txt /srv/telMMXX
1487 -rw-r--r-- 3 user30 user30 81 juil. 21 tel2020
1487 -rw-r--r-- 3 user30 user30 81 juil. 21 Tel-2020.txt
1487 -rw-r--r-- 3 user30 user30 81 juil. 21 /srv/data/telMMXX
user30@deb:~$ mv Tel-2020.txt srv/perso/tel2020.txt
user30@deb:~$ ls -li /srv/perso/Tel-2020.txt
1487 -rw-r--r-- 3 user30 user30 81 juil. 21 /data/perso/tel2020.txt
```



```
user30@deb:~$ ln tel2020 /media/usbkey/
'tel2020.txt': Lien physique inter-périphérique invalide
```



- La suppression d'un lien physique n'a aucune répercussion sur les autres.
- Tant qu'un nom de fichier existe, les données restent accessibles.
- À la suppression du dernier lien, les données sont effacées et le nombre de lien de l'inode passe à 0.
- L'inode n'ayant plus de lien actif ; il pourra être réutilisé par le système pour un autre fichier.



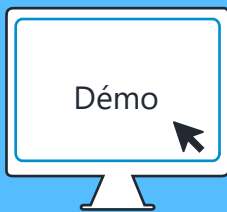
Liens physiques : avantages et contraintes

AVANTAGES

- Pas de problème de chemins lors d'un déplacement de fichiers
- Pas de perte d'inodes
- Théoriquement, accès plus rapide aux données qu'un lien symbolique

CONTRAINTES

- Limité à un espace de stockage, à un filesystem
- Les permissions, utilisateurs, groupes, [...], sont toutes les mêmes pour chaque lien puisque que contenus dans l'inode
- Pas de liens physiques sur un répertoire
- Visibilité moins évidente avec la commande ls.



Les liens sous Linux





TP

•
Les liens sous Linux
•



- Vous savez différencier les différents types de fichiers
- Vous savez lire des fichiers, dans leur totalité ou partiellement
- Vous comprenez le mécanisme des liens sous Linux et savez les manipuler

