

# Linux Utilisation

## Module 08 – Mécanismes Linux et Bash Avancé



1

Mécanismes Linux et Bash Avancé



### Objectifs

- Comprendre Vim et ses modes de fonctionnement
- Savoir ajouter ou supprimer du texte
- Utiliser les commandes spécifiques de Vim
- Gérer les différences des fichiers texte Windows et Linux

*Savoir écrire et modifier des  
fichiers texte sous Unix Linux*



2

# Gestion des processus sous Linux



3

Processus :


- Exécution en mémoire d'un programme
- Plusieurs processus possibles pour un même programme
- Chaque processus possède un Process ID (PID) unique
- Notion de processus parents et enfants



4

• **ps [-aefux]**

- Lister les processus de l'utilisateur




```
user30@deb:~$ ps
  PID TTY          TIME CMD
   808 pts/0        00:00:00 bash
  1212 pts/0        00:00:00 ps
user30@deb:~$
```

PID	Process ID : identifiant unique d'un processus	TIME	Temps cumulé d'exécution
TTY	Nom du terminal auquel le processus est rattaché	COMMAND	Commande à l'origine du processus



Lister tous les processus en cours



```
user30@deb:~$ ps -ef
UID          PID    PPID  C  STIME TTY          TIME CMD
[...]
systemd+    504      1  0  08:57 ?        00:00:00 /lib/systemd/systemd-timesyncd
message+    510      1  0  08:57 ?        00:00:00 /usr/bin/dbus-daemon --system
root        512      1  0  08:57 ?        00:00:00 /lib/systemd/systemd-logind
root        513      1  0  08:57 ?        00:00:00 /usr/sbin/rsyslogd -n -iNONE
root        591      1  0  08:57 ?        00:00:00 /usr/sbin/sshd -D
user30      798      1  0  15:08 ?        00:00:00 /lib/systemd/systemd --user
user30      799     798  0  15:08 ?        00:00:00 (sd-pam)
user30      807     795  0  15:08 ?        00:00:00 sshd: user30@pts/0
user30      808     807  0  15:08 pts/0    00:00:00 -bash
```



- Visualiser les processus de l'utilisateur sous forme d'arbre

user30@deb:~\$ **ps -uf**

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
user30	808	0.0	0.2	7652	4424	pts/0	Ss	15:08	0:00	-bash
user30	1309	0.0	0.1	10916	3204	pts/0	R+	16:10	0:00	\_ ps -uf

- Visualiser tous les processus sous forme d'arbre

user30@deb:~\$ **ps -faux**

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
[...]										
root	591	0.0	0.3	15852	6676	?	Ss	08:57	0:00	/usr/sbin/sshd -D
root	795	0.0	0.3	16632	7892	?	Ss	15:08	0:00	\_ sshd: user30 [priv]
user30	807	0.0	0.2	16916	6016	?	S	15:08	0:00	\_ sshd: user30pts/0
user30	808	0.0	0.2	7652	4424	pts/0	Ss	15:08	0:00	\_ -bash
user30	1348	0.0	0.1	11076	3620	pts/0	R+	16:17	0:00	\_ ps -faux

- **kill [-sig] <pid> (<pid> <pid>)**
- Arrêter (tuer) un processus en précisant le type d'arrêt.
- Sans précision, le signal envoyé est 15, ou SIGTERM

Nom	ID Num	Effet
SIGTERM	15	Envoi d'une instruction au processus pour qu'il se termine proprement. C'est la valeur par défaut et celle à privilégier
SIGINT	2	Envoi d'un signal d'interruption (équivalent de [ctrl]+[c] dans un terminal)
SIGHUP	1	Envoi d'un signal d'arrêt de session terminal (HangUp)
SIGKILL	9	Arrêt incontionnel. Tuer directement le processus (dangereux)

```
user30@deb:~$ ps -f
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
user30	745	744	0	15:31	pts/0	00:00:00	-bash
user30	749	745	0	15:32	pts/0	00:00:00	sleep 600

```
user30@deb:~$ kill 749
```

```
user30@deb:~$ ps -f
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
user30	745	744	0	15:31	pts/0	00:00:00	-bash

```
user30@deb:~$ ps -f
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
user30	634	633	0	10:13	pts/0	00:00:00	-bash
user30	954	634	0	15:16	pts/0	00:00:00	/bin/bash ./erp-install.sh

```
user30@deb:~$ kill -2 954
```



9

```
pskill [-sig] <processname>
```

Arrêter les processus correspondant au nom <processname>


```
user30@deb:~$ pskill firefox
```



10

**time** <commande>

Visualiser le temps d'exécution d'une commande.



```
user30@deb:~$ time wc -l Edition
31 Edition
real    0m0,007s
user    0m0,002s
sys     0m0,000s
```



- Lancer une commande en arrière-plan se fait simplement en ajoutant le caractère **&** à la fin de la ligne.

**commande** [opt] <arg> **&**

- Visualiser les processus en arrière-plan

**jobs** [-1]

**-1**    Afficher les PID des processus.



```
user30@deb:~$ ./sauvegarde.sh &
[1] 841
user30@deb:~$ ./erp-install.sh &
[2] 843
user30@deb:~$ sleep 300 &
[3] 845
user30@deb:~$ jobs
[1]  En cours d'exécution  ./sauvegarde.sh &
[2]- En cours d'exécution  ./erp-install.sh &
[3]+ En cours d'exécution  sleep 300 &
```



```
user30@deb:~$ ./sauvegarde.sh &
[1] 841
user30@deb:~$ ./erp-install.sh &
[2] 843
user30@deb:~$ sleep 300 &
[3] 845
user30@deb:~$ jobs
[1]  En cours d'exécution  ./sauvegarde.sh &
[2]- En cours d'exécution  ./erp-install.sh &
[3]+ En cours d'exécution  sleep 300 &
```



```
user30@deb:~$ nohup ./erp-install.sh &  
[1] 867
```



```
user30@deb:~$ nohup: les entrées sont ignorées et la sortie est  
ajoutée à 'nohup.out'
```

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



Gestion des processus





# Les mécanismes de redirection



17



18

Les Pipelines : `commande1 | commande2`



19

- Enregistrer la sortie standard dans un fichier `commande > fichier.txt`



- Écraser le contenu du fichier : `commande > fichier.txt`
- Ajouter le contenu au fichier : `commande >> fichier.txt`



20

- Enregistrer la sortie d'erreur dans un fichier `commande2 > fichier.txt`



- Écraser le contenu du fichier : `commande 2> fichier.txt`
- Ajouter le contenu au fichier : `commande 2>> fichier.txt`



- Rediriger le flux d'erreur vers le flux de sortie standard : `commande 2>&1`

```
user30@deb:~$ head -3 Edition dir1/fic2 >> result.log 2>&1
user30@deb:~$ cat result.log
==> Edition <==
Aboaf Maurice 244748
Adda Jen 239234
Allo Jean-Pierre 255398
head: impossible d'ouvrir 'dir1/fic2' en lecture: Aucun fichier ou dossier de ce type
```

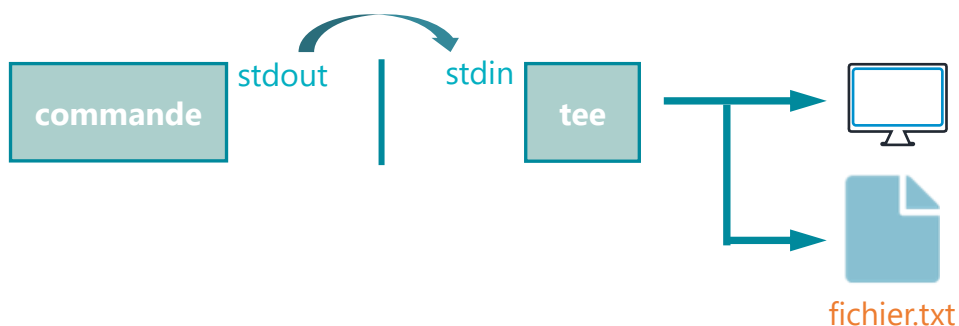


- Spécificité de Bash, une syntaxe plus courte pour le même effet : `commande &>`



Doubler la sortie standard à l'écran ET dans un fichier :

`commande | tee -a fichier.txt`



Rediriger le flux d'entrée depuis un fichier `commande < ficsource.txt`

```
user30@deb:~$ wall < modeles/warning.txt
```

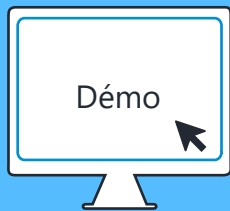
```
Diffusion de message de user30@deb (pts/0) (Tue Sep 8 16:11:45 2020) :
```



```
/!\ Attention /\!  
Redemarrage du serveur  
dans dix minutes
```

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





TP



# Archivage et compression



27

- Archiver : créer un fichier conteneur dans lequel on pourra en stocker plusieurs autres.
- Exemple : un fichier iso
- Compresser : réduire la taille d'un fichier dans le but de minimiser l'espace de stockage occupé ainsi que réduire les temps de transfert.
- Exemple : un fichier zip



28

- Sous Linux : une commande pour tout faire : tar
- **tar [ctxzjvf] archive.tar**
  - **c** Créer une archive
  - **t** Lister le contenu
  - **x** Extraire le contenu
  - **z** Préciser l'utilisation de la compression avec **gzip**
  - **j** Préciser l'utilisation de la compression avec **bzip2**
  - **u** Mettre à jour une archive existante
  - **v** Activer le mode verbeux
  - **f** Préciser le nom du fichier



- Extraire le contenu d'une archive à l'emplacement actuel

```
user30@deb:~$ tar -xf archive.tar.gz
user30@deb:~$
```

- Extraire le contenu d'une archive en utilisant le mode verbeux, dans le dossier restau.d

```
user30@deb:~$ tar xvf archive.tar.gz -C restau.d
contacts.txt
edition1.txt
err.txt
Fait.txt
```



- Créer un fichier **archive.tar** qui contient le fichier **Edition** et tous les fichiers **txt**



```
user30@deb:~$ tar -cvf archive.tar Edition *.txt
Edition
contacts.txt
edition1.txt
err.txt
Fait.txt
```

- Créer le fichier **archive.tar.gz** qui sera **compressé** au format **gzip**



```
user30@deb:~$ tar -czf archive.tar.gz Edition *.txt
user30@deb:~$
```

- On rencontre régulièrement l'extension **.tgz**, une version raccourcie de **.tar.gz**



Créer le fichier **archive.tar.bz2** qui sera **compressé** au format **bzip2**



```
user30@deb:~$ tar cjvf archive.tar.bz2 Edition *.txt
Edition
contacts.txt
edition1.txt
err.txt
Fait.txt
```



**Bzip2** est un autre format de compression disponible sous Linux.  
Il est plus efficace en terme de compression que Gzip mais demande plus de calcul au processeur.





- Ajouter les nouveaux fichiers .txt à une archive existante



```
user30@deb:~$ cd /
user30@deb:~$ tar uvf archive.tar home/user30/*.txt
user30@deb:~$
```

- Pour éviter les soucis dus à la suppression du "/" représentant la racine par la commande `tar`, on se déplace à la racine du système avec la commande `cd /`



### Lister le contenu d'une archive

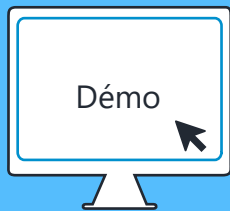


```
user30@deb:~$ tar -tf archive.tar.gz
Edition
contacts.txt
edition1.txt
err.txt
Fait.txt
```



```
user30@deb:~$ tar -tvf archive.tar.gz
-rw----- user30/user30  586 2020-04-16 17:26 Edition
-rw-r--r-- user30/user30   71 2020-07-21 14:16 contacts.txt
lrwxrwxrwx user30/user30    0 2020-07-20 14:55 edition1.txt ->
Edition
-rw-r--r-- user30/user30   86 2020-09-08 15:06 err.txt
-rw-r--r-- user30/user30   39 2020-08-13 17:31 Fait.txt
```





•  
La commande TAR  
•



TP

- 
- 



# Gestion des alias avec Bash



37

- Un alias est un raccourci, un synonyme, une abréviation d'une commande.
- Il permet de créer des séquence plus courtes ou plus sécurisées d'une commande.

```

user30@deb:~$ alias ll="ls -l"
user30@deb:~$ ll
-rw----- user30/user30    586 2020-04-16 17:26 Edition
-rw-r--r-- user30/user30     71 2020-07-21 14:16 contacts.txt
lrwxrwxrwx user30/user30     0 2020-07-20 14:55 edition1.txt ->
Edition
-rw-r--r-- user30/user30     86 2020-09-08 15:06 err.txt
-rw-r--r-- user30/user30     39 2020-08-13 17:31 Fait.txt
  
```




38



```
user30@deb:~$ alias rm="rm -i"  
user30@deb:~$
```



```
user30@deb:~$ alias ipa="ip address show dev ens33 | grep 'inet '  
user30@deb:~$ ipa  
inet 192.168.1.73/24 brd 192.168.1.255 scope global dynamic ens33
```




```
user30@deb:~$ alias  
alias grep='grep --color=auto'  
alias ipa='ip address show dev ens33 |grep '\''inet '\'''  
alias ll='ls -l'  
alias ls='ls --color=auto'  
alias rm='rm -i'  
user30@deb:~$
```



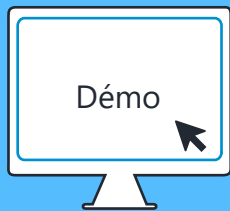


```
user30@deb:~$ vim ~/.bashrc
[...]  
# some more ls aliases  
alias ll='ls -l'  
#alias la='ls -A'  
#alias l='ls -CF'  
[...]  
### Mes alias ###  
alias rm="rm -i"  
alias ipa="ip address show dev ens33 |grep 'inet '  
user30@deb:~$ . .bashrc
```



```
user30@deb:~$ unalias rm  
user30@deb:~$ alias  
alias grep='grep --color=auto'  
alias ipa='ip address show dev ens33 |grep '\"inet '\"'  
alias ll='ls -l'  
alias ls='ls --color=auto'
```





- Les alias



# Plus loin avec les variables



- Chaque **variable** porte un nom.
- Ce nom permet de récupérer, modifier, le contenu de cette mémoire.
- La donnée n'existe que dans la **mémoire vive** du système.
- Pour lire une variable, on accole un caractère **\$** avant le nom de la variable.



```
var="Ceci est le contenu de ma variable"
echo $var
Ceci est le contenu de ma variable
```



- Créer une variable locale



```
user30@deb:~$ prenom="Romain"
user30@deb:~$ echo $prenom
Romain
```

- Transformer (exporter) une variable locale en variable d'environnement



```
user30@deb:~$ export prenom
user30@deb:~$ bash
user30@deb:~$ echo $prenom
Romain
```




- Créer une variable d'environnement



```
user30@deb:~$ export departement="Comptabilite"
user30@deb:~$ bash
user30@deb:~$ echo $departement
Comptabilite
```

- Détruire une variable




```
user30@deb:~$ unset departement
user30@deb:~$ echo $departement

user30@deb:~$
```




- Créer une variable à partir d'une commande



```
user30@deb:~$ heure=$(date +%Hh%M)
user30@deb:~$ echo $heure
11h35
```

- Version compatible avec d'anciens shell. Considérée obsolète



```
user30@deb:~$ heure=`date +%Hh%M`
user30@deb:~$ echo $heure
11h35
```






- Rappel : les variables d'environnement systèmes sont saisies en majuscules.

**PATH** chemins des exécutable (séparés par des ":")

- Cette variable permet au shell de savoir où trouver les programmes (ou scripts) sans devoir en préciser la localisation.



```
user30@deb:~$ which ls
/usr/bin/ls
user30@deb:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```



```
user30@deb:~$ ls -l Edition
```


Commande présente dans  
les chemins de \$PATH ?

```
$PATH /usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
/usr/bin/ls -> OK
```

Transformation de la  
commande

```
/usr/bin/ls -l Edition
```

Résultat



```
-rw----- 1 user30 user30 586 avril 16 17:26 Edition
```





```
user30@deb:~$ Get-ChildItem Edition
```

Commande présente dans les  
chemins de \$PATH ?

```
$PATH /usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games  
-> KO
```

Message d'erreur



```
-bash: Get-ChildItem : commande introuvable
```



Manipuler les variables





TP

•

•



- Vous savez visualiser et gérer les processus sous Linux
- Vous savez utiliser les flux du shell et les rediriger
- Vous savez manipuler les archives avec Tar
- Vous savez manipuler les alias et les variables du shell

