LINUX – Fonctionnement du bash / Synthèse

▪ Substitutions du shell

Le shell effectue des substitutions :

de commandes	` ou $(...)
de variables	$
de noms de fichier	* […] ?

Il interprète également les caractères ' " \ qui permettent d'inhiber ces substitutions.
Seuls $ \ et `sont interprétés entre " "
Tout caractère entre ' ' est inhibé
Le \ inhibe son voisin (et même le mot qui le suit echo *** affiche ***)

Exemple :
a=1

echo "$a * \$b ' et " '$a $b' `echo $a` affiche 1 * $b ' et $a $b 1

Si le nom de la commande comporte un / il la lance sans ambiguité (ex. /bin/ls ou ./affiche)
sinon il cherche dans l'ordre s'il s'agit : d'un alias, d'une fonction ,d'une commande interne, puis il finit par le PATH. La variable PATH permet de limiter la recherche des répertoires où peut se trouver la commande (l'ordre de recherche est important car dès que le shell a trouvé il lance la commande)

▪ Redirections et pipe

Le shell interprète également les caractères <, >, <<, >>, et |

Il redéfinit l'entrée et la sortie standard des commandes qui sont normalement le clavier et l'écran.

ps -ef | grep httpd | wc -l

▪ Processus père et fils

Lorsqu'on lance un script, un shell fils se lance pour interpréter ce script. Le processus père lui transmet les variables exportées. Si l'on ne souhaite pas qu'il y ait d'enfant il faut lancer le script avec la commande interne "source" ou la commande "." (source .bashrc)

GREP - SED - AWK / Synthèse pour les TP

cat proc

```
UID      PID      PPID     CMD      SIZE
root     1        0        init     468
root     388      1        sh       1228
root     396      388      pppd     804
root     467      1        crond    632
named    480      1        named    1780
root     749      1        xdm      996
nobody   32021    710      httpd    2540
nobody   20602    710      httpd    2540
root     493      1        xinetd   972
root     22761    493      telnetd  584
root     22762    22761    login    1108
form     22763    22762    -bash    1368
form     22764    22763    vi       860
```

▪ grep

grep in proc
```
root    1       0       init    468
root    493     1       xinetd  972
root    22762   22761   login   1108
```

grep '^n' proc
```
named   480     1       named   1780
nobody  32021   710     httpd   2540
nobody  20602   710     httpd   2540
```

grep '^n.*80$' proc
```
named   480     1       named   1780
```

grep '^[^rf]' proc
```
UID     PID     PPID    CMD     SIZE
named   480     1       named   1780
nobody  32021   710     httpd   2540
nobody  20602   710     httpd   2540
```

▪ sed

SYNTAXE GENERALE
sed 'ligne1,ligne2 ACTION paramètres' fichier

ligne1 et ligne2 peuvent être des numéros de ligne
ou des expressions régulières (entre /.../)
ligne2 est facultative

ACTION de base = **d**(elete), **p**(rint), ou **s**(ubstitute)

EDITION
sed '1,12d' proc
```
form    22763   22762   -bash   1368
form    22764   22763   vi      860
```

sed '/^root/,/8$/d' proc
```
UID     PID     PPID    CMD     SIZE
form    22763   22762   -bash   1368
form    22764   22763   vi      860
```

Explication (en encadré les 3 paquets qui correspondent à /^root/,/8$/)
```
UID     PID     PPID    CMD     SIZE
root    1       0       init    468
root    388     1       sh      1228
root    396     388     pppd    804
root    467     1       crond   632
named   480     1       named   1780
root    749     1       xdm     996
nobody  32021   710     httpd   2540
nobody  20602   710     httpd   2540
root    493     1       xinetd  972
root    22761   493     telnetd 584
root    22762   22761   login   1108
form    22763   22762   -bash   1368
form    22764   22763   vi      860
```

sed -n '/^n/p' proc
```
named   480     1       named   1780
nobody  32021   710     httpd   2540
nobody  20602   710     httpd   2540
```

SUBSTITUTION

```
sed 's/o/<O>/g' proc
UID     PID     PPID    CMD     SIZE
r<O><O>t    1       0       init    468
r<O><O>t    388     1       sh      1228
r<O><O>t    396     388     pppd    804
r<O><O>t    467     1       cr<O>nd   632
named   480     1       named   1780
r<O><O>t    749     1       xdm     996
n<O>b<O>dy  32021   710     httpd   2540
n<O>b<O>dy  20602   710     httpd   2540
r<O><O>t    493     1       xinetd  972
r<O><O>t    22761   493     telnetd 584
r<O><O>t    22762   22761   l<O>gin   1108
f<O>rm    22763   22762   -bash   1368
f<O>rm    22764   22763   vi      860
```

```
sed -e 's/[0-9]//g' proc
UID     PID     PPID    CMD     SIZE
root                  init
root                sh
...
```

INSERTION

```
sed -e '/root/i\
      voici un nouveau texte\
      sur deux lignes ' proc
UID     PID     PPID    CMD     SIZE
voici un nouveau texte
sur deux lignes
root    1       0       init    468
voici un nouveau texte
sur deux lignes
root    388     1       sh      1228
voici un nouveau texte
sur deux lignes
root    396     388     pppd    804
voici un nouveau texte
sur deux lignes
root    467     1       crond   632
named   480     1       named   1780
...
```

```
cat fcommandes
1d
4q
sed  -f  fcommandes proc
root    1       0       init    468
root    388     1       sh      1228
root    396     388     pppd    804
```

▪ awk

```
awk '{print $2,$5}' proc
PID SIZE
1 468
388 1228
396 804
```

```
awk '/^n/   {print $0}' proc
named   480     1       named   1780
nobody  32021   710     httpd   2540
nobody  20602   710     httpd   2540
```

```
awk '/vi/ {print $1,$2,$5}' proc
form 22764 860

awk '/vi/ {print $1,$2,$5}' proc | read a b c
echo "$a => $b => $c"
form => 22764 => 860

awk '/vi/ {print $1 " => " $2 " => "$5}' proc
form => 22764 => 860


awk ' $4 ~ /d$/ && $1 ~/^n/  && $2 > 21000 {print $0}' proc
nobody  32021   710    httpd   2540
```

Autres exemples avec GREP - SED - AWK

```
cat proc
UID      PID      PPID     CMD      SIZE
root     1        0        init     468
root     388      1        sh       1228
root     396      388      pppd     804
root     467      1        crond    632
named    480      1        named    1780
root     749      1        xdm      996
nobody   32021    710      httpd    2540
nobody   20602    710      httpd    2540
root     493      1        xinetd   972
root     22761    493      telnetd  584
root     22762    22761    login    1108
form     22763    22762    -bash    1368
form     22764    22763    vi       860
```

Avec grep

Extraire les lignes de ce fichier qui :

1 contiennent le mot nobody

```
grep nobody proc                                    # 1
nobody   32021    710      httpd    2540
nobody   20602    710      httpd    2540
```

2 commencent par un n

```
grep '^n' proc                                      # 2
named    480      1        named    1780
nobody   32021    710      httpd    2540
nobody   20602    710      httpd    2540
```

3 commencent par une majuscule

```
grep '^[[:upper:]]' proc                                  #
3
UID      PID      PPID     CMD      SIZE
```

4 contiennent un a en deuxième position

```
grep '^.a' proc                                     # 4
```

```
named   480     1       named   1780
```

5 se terminent par un 8

```
grep '8$' proc                                          # 5
root    1       0       init    468
root    388     1       sh      1228
root    22762   22761   login   1108
form    22763   22762   -bash   1368
```

6 commencent par un 'f' et se termine par un '8'

```
grep '^f.*8$' proc                                      # 6
form    22763   22762   -bash   1368
```

7 contiennent uniquement des blancs et des majuscules

```
grep  '^[A-Z∪][A-Z∪]*$' proc                            # 7
UID     PID     PPID    CMD     SIZE
```

Avec **egrep**

8 Extraire les lignes qui contiennent le mot xdm ou login

```
egrep 'xdm|login' proc                                  # 8
root    749     1       xdm     996
root    22762   22761   login   1108
```

Avec **sed**

9 Extraire les lignes qui contiennent le mot nobody

```
sed -n '/nobody/p' proc                                 #9
```

10 Supprimer les lignes qui contiennent le mot nobody

```
sed '/nobody/d' proc                                    #10
```

11 Remplacer le mot root par ROOT

```
sed 's/root/ROOT/g' proc                                #11
```

12. Substituer les blancs consécutifs par 1 seul blanc

```
sed 's/  */ /g' proc                                    #12
```

13. Substituer les chiffres par des *

```
sed 's/[0-9]/*/g' proc                                  #13
```

14 Substituer les chiffres par rien du tout (les retirer)

```
sed 's/[0-9]//g' proc                                   #14
```

15 Ajouter des commentaires devant chaque ligne (substitution de ^.*$ par #&)

```
sed 's/^.*$/# &/' proc                                  #15
```

16 Ajouter la ligne "Util Proc Pere Commande Taille" derrière la ligne qui se termine par SIZE

```
sed '/SIZE$/a\                                                  # 16
Utilisateur       Processus Pere Commande  Taille' proc
UID      PID      PPID     CMD      SIZE
Util     Proc Pere Commande  Taille
root     1        0        init     468
...
```

17 Pour chaque tronçon délimité par une ligne commençant par 'root' jusqu'à une ligne contenant un 'h' remplacer les chiffres par des *

```
sed '/^root/,/h/s/[0-9]/*/g' proc                               # 17
UID      PID      PPID     CMD      SIZE
root     *        *        init     ***
root     ***      *        sh       ****
root     ***      ***      pppd     ***
root     ***      *        crond    ***
named    ***      *        named    ****
root     ***      *        xdm      ***
nobody   *****    ***      httpd    ****
nobody   20602    710      httpd    2540
root     ***      *        xinetd   ***
root     *****    ***      telnetd  ***
root     *****    *****    login    ****
form     *****    *****    -bash    ****
form     22764    22763    vi       860
```

Avec **awk**

18 Extraire les lignes qui contiennent le mot nobody

```
awk '/nobody/ {print $0}' proc                                  # 18a
awk '/nobody/ {print }' proc                                    # 18b
awk '/nobody/' proc                                             # 18c
```

19 Extraire les colonnes UID et CMD

```
awk '{print $1,$4}' proc                                        # 19
```

20 Insérer un # devant chaque ligne (encadrer le # par " " et non ' ')

```
awk '{print "#", $0}' proc                                      #20
# UID      PID      PPID     CMD      SIZE
# root     1        0        init     468
# root     388      1        sh       1228
# root     396      388      pppd     804
# root     467      1        crond    632
```

Synthèse des caractères spéciaux du bash

Pour des raisons pédagogiques, ils sont présentées dans le "sens des aiguilles d'une montre" par rapport à un clavier AZERTY classique :

&	arrière plan
~	home directory
~user	home directory de user
"..."	inhibe tout sauf \ , $, et `
#	commentaires
'...'	inhibe tout (attention à la parité)
{...}	limite la portée du $ comme dans ${var}-suite
(...)	créer un sous shell
[`...]	opérateur de test
|	séparateur de commande (pipe)
`...	substitution de commande comme dans rm `cat liste`
\	inhibe le caractère qui le suit
$	substitution par la valeur de la variable (echo $PATH)
$ (...)	substitution par la valeur de la commande comme dans rm $(cat liste)
*	tous les fichiers du répertoire courant sauf ceux qui commencent par .
!	négation dans les tests et dans les métacaractères (rm *.[!co])
:	null operator
;	séparateur de commandes comme dans sleep 60; init 0
?	joker pour un caractère dans un nom de fichier
<	redirection de l'entrée standard
>	redirection de la sortie standard