Rechercher des fichiers

1. Considérations générales

La commande **find** permet de rechercher des fichiers au sein de l'arborescence du système de fichiers à l'aide de critères et donne la possibilité d'agir sur les résultats retournés.

```
find chemin critères options
```

La commande **find** étant récursive, il suffit d'indiquer un répertoire de base pour que toute l'arborescence depuis ce répertoire soit développée. L'option de base est `-print` (souvent implicite sur la plupart des Unix) qui permet d'afficher sur écran les résultats.

```
seb@slyserver:~/Documents/slyunix> find
.
./logos-carre.tif
./logos-carre.eps
./Page 5.pdf
./logos-carre-grand.jpg
./LOGOS
./site_2.jpg
./pub_planete.pdf
./index_logon_inc.php
./logo-iceberg.eps
./flyer
./flyer/sly4.jpg
./flyer/flyerx4.sxd
./flyer/sly1.jpg
./flyer/sly2.jpg
./flyer/flyer.sxd
./flyer/sly3.jpg
./flyer/flyer.jpg
...
```

Le chemin précisé étant relatif, l'affichage est relatif. Si le chemin précisé était absolu, l'affichage aurait été absolu.

2. Critères de recherche

Les paramètres permettent de définir les critères de recherche. Ces critères, s'ils sont plusieurs, sont combinés entre eux par un ET (critère1 ET critère2).

a. -name

`-name` permet une sélection par noms de fichiers. Il est possible d'utiliser les wildcards déjà vus. Le critère est idéalement placé entre guillemets. Ici la liste de tous les fichiers depuis l'emplacement courant et commençant par « fic » est affichée.

```
$ find . -name "fic*" -print
./fic1
./fic2
./fic3
./fic4
```

b. -type

`-type` permet une sélection par type de fichier. Vous savez déjà que outre les liens, les répertoires et les fichiers simples, étaient présents d'autres types de fichiers.

Code	Type de fichier

b	Fichier spécial en mode bloc
c	Fichier spécial en mode caractère
d	Répertoire (directory)
f	Fichier ordinaire
l	Lien symbolique
p	Tube nommé (pipe)
s	Socket (Connexion réseau)

Tous les répertoires dont le nom commence par « re » sont affichés.

```
$ find . -name "re*" -type d -print
./rep1
./rep2
```

c. -user et -group

`-user` et `-group` permettent une recherche sur le propriétaire et le groupe d'appartenance des fichiers. Il est possible de préciser le nom (utilisateur, groupe) ou l'ID (UID, GID). L'exemple suivant recherche tous les fichiers ordinaires appartenant à seb et au groupe users.

```
$  find . -type f -user seb -group users -print
./fic1
./fic3
```

d. -size

`-size` permet de préciser la taille des fichiers recherchés. Sa syntaxe est particulière car elle travaille par défaut en blocs si vous ne précisez rien. C'est parfois surprenant d'autant plus que le bloc, qui a ici une taille de 512 octets, est une unité un peu virtuelle (avec certaines commandes un bloc peut faire 1 Ko ou plus).

La valeur située après le critère peut être suivie des caractères b, c, w ou k.

Caractère	Signification
b	Par défaut si non précisé, c'est un bloc de 512 octets.
c	C'est un caractère, au sens ASCII, donc 1 octet.
w	C'est un mot (au sens ancien) de 2 octets.
k	1 Ko (1024 octets).

La valeur peut être précédée d'un + ou d'un - signifiant "plus de" ou "moins de". Sans cette indication, la taille recherchée doit correspondre EXACTEMENT.

- -size 5 : recherche les fichiers d'une taille de 5 blocs (512 octets par bloc, soit ici 2560 octets).
- -size 152c : recherche les fichiers d'une taille de 152 caractères (octets).
- -size 10k : recherche les fichiers d'une taille de 10 Ko (10*1024 octets = 10240 octets).
- -size +5000k : les fichiers de plus de 5000 Ko.

- -size -100k : les fichiers de moins de 100 Ko.

```
seb@slyserver:/var/log> find -size +100k
./zypper.log-20080227.bz2
./lastlog
./zypper.log-20080302.bz2
./wtmp
./zypper.log-20080226.bz2
./zypper.log
./messages
```

Le critère de recherche **-empty** peut être utilisé en remplacement de -size 0.

e. -atime, -mtime et -ctime

- `-atime` : recherche sur la date du dernier accès (access time). Un accès peut être la lecture du fichier, mais aussi le simple fait de le lister spécifiquement.
- `-mtime` : recherche sur la date de dernière modification (modification time). C'est de la modification du contenu qu'il s'agit.
- `-ctime` : recherche sur la date de changement (change time, en fait la date de dernière modification du numéro d'inode).

La date de changement du fichier correspond à la date où les informations liées à l'inode (voir chapitre Les disques et le système de fichiers) ont été modifiées pour la dernière fois : modification du nom, déplacement, changement des droits, de la taille, etc.).

Ces trois critères ne travaillent qu'avec des jours (périodes de 24 heures). 0 est le jour même, 1 hier, 2 avant-hier, etc. La valeur n située après le critère correspond donc à n*24 heures. Cette plage n'est pas fixe car « hier » signifie il y a entre 24 et 48 heures...

Les signes + ou - permettent de préciser les termes « de plus » et « de moins » :

- -mtime 1 : fichiers modifiés hier (entre 24 et 48 heures).
- -mtime -3 : fichiers modifiés il y a moins de trois jours (72 heures).
- -atime +4 : fichiers modifiés il y a plus de 4 jours (plus de 96 heures).

```
seb@slyserver:/var/log> find . -mtime -1
./kdm.log
./vmware
./vmware/vmware-serverd-0.log
./vmware/vmware-serverd.log
./mail.info
./Xorg.0.log
./lastlog
./Xorg.0.log.old
./warn
...
```

Vous pouvez jeter un oeil aux critères **-newer**, **-anewer** et **-cnewer** qui prennent comme paramètre un fichier. Dans ce cas find recherche les fichiers plus récents que celui précisé.

f. -perm

-perm permet d'effectuer des recherches sur les autorisations d'accès (droits, SUID, SGID, Sticky). Les droits doivent être précisés en base 8 (valeur octale) et complets. Le caractère - placé devant la valeur octale signifie que les fichiers recherchés doivent au moins avoir les droits désirés. Le + indique que le fichier doit avoir au moins l'un des droits spécifiés, d'où une nuance. Dans l'exemple suivant sont recherchés les répertoires où tout le monde (user, group, others) a le droit de pénétrer (droit x, soit 1).

```
seb@slyserver:/var/log> find -type d -perm -111
.
./vmware
./vmware/vmsd-xaction
./cups
```

g. -links et -inum

Bien que ces critères fassent appel à des notions plus avancées du système de fichier, il est bon de les présenter dès maintenant. Vous pourrez y revenir dès que le chapitre Les disques et le système de fichiers vous aura présenté le fonctionnement interne d'un système de fichiers.

L'option -links permet une recherche par nombre de hard links. Vous pouvez préciser les signes + ou - (plus de n liens et moins de n liens). Un fichier normal seul possède 1 lien. Un répertoire 2 liens (l'entrée dans le catalogue dont il fait partie et dans le point). Pour une recherche de liens symboliques il faudra utiliser l'option -type l.

```
$ find . -type f -links +2 -print
./fic2
./hardlink3_fic2
./hardlink_fic2
./hardlink2_fic2
```

-inum permet une recherche par numéro d'inode. Elle est utile dans le cas d'une recherche de tous les liens portant un même numéro d'inode. Le numéro d'inode est visible par l'option -i de la commande **ls**.

```
seb@slyserver:/var/log> ls -i
491891 acpid                491793 mail.info   491860 Xorg.0.log
491791 boot.log             491794 mail.warn   490686 Xorg.0.log.old
491729 boot.msg             492046 mcelog      492060 Xorg.1.log
seb@slyserver:/var/log> find . -inum 491791 -print
./boot.log
```

3. Commandes

Outre l'option -print on trouve d'autres options permettant d'effectuer une action sur les fichiers trouvés.

a. -ls

Le critère affiche des informations détaillées sur les fichiers trouvés correspondant au critère au lieu du simple nom de fichier. La sortie correspond à une commande **ls** avec les paramètres d, i, l et s (taille en blocs de 1 Ko).

```
seb@slyserver:~> find -size +500000k -ls
2342935 584388 -rw-r--r--   1 seb      users    597817344 fév 24
11:52 ./eeexubuntu-7.10.3-desktop-i386.iso
```

b. -exec

Le critère -exec va exécuter la commande située juste après pour chaque occurrence trouvée. Quelques remarques s'imposent :

- -exec doit obligatoirement être la dernière option de la commande **find**.
- La commande exécutée par -exec doit se terminer par un « ; ». Ce caractère spécial doit s'écrire \; pour ne pas être interprété par le shell.

- Pour passer comme paramètre pour la commande le fichier trouvé par find, il faut écrire {} (substitution du fichier).

Exemple pour effacer tous les fichiers finissant par « .mp3 » :

```
$ find . -type f -name "*.mp3" -exec rm -f {} \;
```

La commande **find** n'attend pas d'avoir trouvé tous les fichiers avant d'exécuter la commande précisée. Elle la lance dès qu'un fichier est trouvé. Aussi si la commande précédente vous a affiché n fichiers avant que vous ne pensiez à l'interrompre, alors ces n fichiers sont déjà perdus.

---

c. -ok

Le critère -ok est identique à l'option -exec mais, pour chaque occurrence, une confirmation est demandée à l'utilisateur.

```
$ find . -inum 95 -ok rm -f {} \;
< rm ... ./fic1 > (yes)?   n
< rm ... ./lien_fic1 > (yes)?   y
```

4. Critères AND / OR / NOT

Il est possible de combiner les options de critère de sélection. Sans aucune précision c'est le ET logique qui est implicite.

Critère	Action
-a, -and	AND, ET logique, par défaut
-o, -or	OR, OU logique
!	Négation du critère

Exemple avec tous les fichiers ne contenant pas fic dans leur nom, et tous les fichiers n'étant ni normaux ni des répertoires.

```
$ find . ! -name "*fic*" -print
.
./rep1
./liste
./mypass
./users
./liste2
./ls.txt
./toto.tar.gz
./nohup.out
./liste_ls
./rep2
./seb1
./seb2
$ find . ! \( -type f -o -type d \) -ls
     409    0 lrwxrwxrwx  1 oracle   system          4 Aug 14 15:21
./lien_fic1 -> fic1
     634    0 lrwxrwxrwx  1 oracle   system          4 Aug 14 15:21
./lien_fic2 -> fic2
```

5. Retrouver des exécutables

a. whereis

La commande **whereis** recherche dans les chemins de fichiers binaires, du manuel et des sources les fichiers correspondant aux critères fournis.

```
$ whereis date
date: /bin/date /usr/share/man/man1/date.1.gz
/usr/share/man/man1p/date.1p.gz
```

Vous pouvez préciser quelques paramètres :

- `-b` uniquement pour les binaires,
- `-m` uniquement pour les manuels,
- `-s` uniquement pour les sources.

Les fichiers sont recherchés par défaut dans :

```
/{bin,sbin,etc}
/usr/{lib,bin,old,new,local,games,include,etc,src,man,sbin,X386,TeX,
g++-include}
/usr/local/{X386,TeX,X11,include,lib,man,etc,bin,games,emacs}
```

Donc ne soyez pas étonné d'obtenir ceci :

```
$ whereis -b passwd
passwd: /usr/bin/passwd /etc/passwd /etc/passwd.old
/etc/passwd.YaST2save /etc/passwd.vipwKSnTgH /usr/bin/X11/passwd
```

b. which

La commande **which** recherche une commande dans le PATH (chemin des exécutables) et vous fournit la première qu'elle trouve :

```
$ which date
/bin/date
```

Il arrive que des commandes de même nom existent dans plusieurs chemins, vous pouvez dès lors préciser le paramètre `-a` pour que which continue sa recherche. Sachez cependant que c'est la première qui sera exécutée par défaut si vous la lancez.

```
$ which -a passwd
/usr/bin/passwd
/usr/bin/X11/passwd
```

6. locate

La commande **locate** recherche un fichier selon le modèle donné dans une base de données de fichiers construite par la commande **updatedb**.

La commande **updatedb** prend une série de chemins dans laquelle elle va effectuer un **find** et stocker tous les résultats dans une base indexée. Cela évite donc pour les recherches classiques d'effectuer de nouveau un find. Dans la pratique, il suffit de préciser à updatedb la liste des chemins ou ne pas mettre en base les fichiers.

updatedb est généralement lancé par la crontab de manière quotidienne. Les paramètres de la commande sont parfois placés dans un fichier **`/etc/sysconfig/locate`**.

```
$ cat /etc/sysconfig/locate | grep -Ev "^(#|$)"
RUN_UPDATEDB=yes
RUN_UPDATEDB_AS=nobody
UPDATEDB_NETPATHS=""
UPDATEDB_PRUNEPATHS="/mnt /cdrom /tmp /usr/tmp /var/tmp /var/spool
/proc /media /sys"
```

```
UPDATEDB_NETUSER=""
UPDATEDB_PRUNEFS=""
```

La commande lancée dans ce cas est la suivante :

```
# updatedb --localuser=nobody --prunepaths=/mnt /cdrom /tmp /usr/tmp
/var/tmp /var/spool /proc /media /sys
```

Si vous lancez ainsi la commande, elle risque de consommer toutes les ressources du processeur de votre machine, aussi en réalité updatedb est lancé avec une priorité basse.

La base est placée dans **/var/lib/locatedb**.

```
$ locate toto
/opt/kde3/share/apps/ksgmltools2/docbook/xsl/html/autotoc.xsl
/opt/kde3/share/apps/ksgmltools2/docbook/xsl/params/autotoc.label.se
parator.xml
/usr/share/gnome/help/gnome-doc-xslt/C/db2html-autotoc.xml
/usr/share/xml/docbook/stylesheet/nwalsh/1.73.1/fo/autotoc.xsl
/usr/share/xml/docbook/stylesheet/nwalsh/1.73.1/html/autotoc.xsl
/usr/share/xml/docbook/stylesheet/nwalsh/1.73.1/params/autotoc.label
.in.hyperlink.xml
/usr/share/xml/docbook/stylesheet/nwalsh/1.73.1/params/autotoc.label
.separator.xml
/usr/share/xml/docbook/stylesheet/nwalsh/1.73.1/xhtml/autotoc.xsl
/usr/share/xml/gnome/xslt/docbook/html/db2html-autotoc.xsl
/usr/src/linux-2.6.22.17-0.1/drivers/mtd/maps/omap-toto-flash.c
/usr/src/linux-2.6.22.17-0.1/drivers/mtd/nand/toto.c
/usr/src/linux-2.6.24.4/drivers/mtd/maps/omap-toto-flash.c
/usr/src/linux-2.6.24.4/drivers/mtd/nand/toto.c
```