Find

Introduction

Les exemples de ce tuto sont tous à faire en *user* sauf spécification.

Cette commande permet de faire des recherches de fichier ou de dossier dans une hiérarchie de répertoires.

Par exemple, je voudrais chercher le log messages, mais je sais pas où se trouve ce fichier, faites :

```
find / -name 'messages'
```

résultat de la commande précédente

```
/var/log/messages
```

Voilà la réponse :
Il se trouve dans le répertoire `/var/log`.

Nota

Notez l'utilisation des apostrophes *afin d'éviter que l'interpréteur de commande n'étende le motif*. Elles sont inutiles dans ce cas-ci mais c'est une bonne pratique de toujours les utiliser afin d'éviter l'extension *motif*.

Quelques options

Options	Fonctions
-atime n ou +n ou -n	*trouve les fichiers auxquels on a accédé il y a strictement n jours, ou plus de n jours, ou moins de n jours*
-mtime n ou +n ou -n	*trouve les fichiers modifiés il y a strictement n jours, ou plus de n jours, ou moins de n jours*
-maxdepth n	*définit le niveau maximum de sous-répertoire à explorer*
-type l ou d ou f	*indique le type de fichier à rechercher (l pour lien symbolique, d pour répertoire (directory), f pour fichier)*

Options	Fonctions
-name	*recherche par motif en respectant la casse*
-iname	*recherche par motif sans respecter la casse*

Recherche simple par nom

Example simple : comment trouver un fichier portant le nom `note` ?

```
find / -name 'note'
```

Décomposition de la commande de l'exemple :

1. “`/`” indique que nous voulons chercher notre fichier à partir de la racine.
2. “`-name` ” est l'option qui indique ici que nous voulons spécifier le nom d'un fichier.

Après un long délai d'attente, la recherche se faisant dans toute l'arborescence de la partition, la réponse finit par venir :

résultat de la commande précédente

```
/home/martin/note
```

Si l'on n'est pas sûr de la casse (Majuscule ou minuscule) on utilise l'option `-iname`.

Règle générale, on recherche rarement un fichier depuis la racine.

Prenons un autre exemple.

Pour chercher tous les fichiers commençant par *note* et définir à partir de quel répertoire on souhaite effectuer la recherche on utilise cette syntaxe :

```
find /home/martin -name 'note*'
```

Recherche par nom simple & multiple

Maintenant, regardons, encore une fois à l'aide d'un exemple, la syntaxe de la commande **find** si l'on recherche plutôt un ou plusieurs répertoires.

Je cherche à trouver les répertoires archives dans /media/homebis. Première chose à noter, il peut-être nécessaire de se mettre en root pour avoir accès à tous les répertoires.

```
find /media/homebis -type d -name 'archives'
```

Dans ce cas-ci, je demande donc à **find** de trouver les répertoires

1. option : -type
2. argument : “d” (comme “directory”)

indiquant que l'on cherche un répertoire du nom de *archives* à partir du répertoire /media/homebis.

La réponse :

résultat de la commande précédente

```
/media/homebis/martin/textes/mes_archives/Baseball/archives
/media/homebis/martin/archives
/media/homebis/Documents_gr/Mots_croises/archives
/media/homebis/Documents_gr/archives
/media/homebis/Documents_gr/mes_fichiers/archives
```

Autre exemple un peu plus complexe cette fois.
Je désire faire une recherche de tous les fichiers audio de type .mp3 et .ogg

Il existe plus d'une façon d'y arriver.
Voyons comment on peut s'y prendre.

Première façon :

```
find /home/martin/  \( -name '*.mp3' -o -name '*.ogg' \)
```

On peut noter l'utilisation du -o qui correspond à l'opérateur ou (“or” en anglais)

Cela me donnera toute une liste de fichiers /home/martin/...

Deuxième façon :

Une autre manière d'écrire la commande ci-dessus est la suivante :

```
find -type f -name "*.mp3" -o -name "*.ogg"
```

Si je tape cette commande en étant dans mon répertoire /home/martin, le résultat sera une liste de fichiers ./....
Il est intéressant de savoir que l'on peut étendre la recherche aux fichiers mp3 et mp4 en remplaçant le 3 par un ?[2)]. La commande deviendrait donc :

```
find -type f -name "*.mp?" -o -name "*.ogg"
```

Rechercher pour supprimer

Une fonction intéressante de *find* est de supprimer en lot les fichiers trouvés.

Il n'est point rare de télécharger ou d'installer de nombreux fichiers qui ne nous servent plus, mais devant le travail pénible de devoir supprimer tous ces fichiers, on repousse au lendemain cette charge. Heureusement grâce à la fonction -delete de find, c'est un pur bonheur.

Paramètre -delete

Exemple, si dans votre home ou autre dossier vous avez beaucoup de fichier .tar.gz qui ne vous servent plus à rien. Il suffit de lancer la commande suivante :

```
find -iname "*.tar.gz" -delete
```

Attention, la fonction -delete ne vous demande pas de confirmation

Supprimer avec demande de confirmation

Pour une demande de confirmation avant suppression de chaque fichier “.tar.gz” trouvés :

```
find -iname "*.tar.gz" -ok rm {} \;
```

Merci à MicP pour cette trouvaille :)

Filtrer en fonction des droits

Une option très pratique est `-perm` qui permet de sélectionner des fichiers en fonctions de leurs droits.
Les droits peuvent être donné en forme octale, par exemple 0755 ou littérale, u=rwx,g=rw,o=rw.

Voici par exemple comment obtenir la liste de tout les fichiers dans le repertoire `/bin` qui ont le `bit setuid valant 1` :

```
find /bin -perm /5000 -user root
```

résultat de la commande précédente

```
/bin/su
/bin/mount
/bin/umount
/bin/ping
/bin/ping6
```

Cette option est intéressante pour la sécurité. Les fichiers listés dans la commande précédente sont tous exécuté avec les droits `root`.

Trois notations avec `perm`, sans préfixe, précédé du signe – ou précédé du signe /

- *sans préfixe* :

le mode du fichier doit être exactement celui passé à l'option -perm.
Par exemple, si on cherche les fichiers ayant le mode u=rwx (0700), tous les fichiers que l'on trouvera auront exactement le mode u=rwx (0700).

- *avec le signe –* :

le mode du fichier doit être au moins égal à celui passé à l'option -perm
-u=r (-0400) → u=r ou u=rw ou u=rx ou u=rwx ou u=r,g=x …

- *avec le signe /* :

un des modes (user, group ou other) doit être au moins égal à ceux passés à l'option -perm
/u=w,g=w,o=w → u=w ou g=w ou o=w ou u=w,g=w,o=w ou u=rw,g=rwx …

Recherche par motif

Pour rechercher un motif, il faut utiliser la même option, et utiliser les REGEXP.

Voici par exemple la recherche de tous les fichiers terminant par `.java` dans le dossier courant:

```
find . -name '*.java'
```

résultat de la commande précédente

```
./java/jdk1.5.0_06/demo/applets/Animator/Animator.java
./java/jdk1.5.0_06/demo/applets/ArcTest/ArcTest.java
./java/jdk1.5.0_06/demo/applets/BarChart/BarChart.java
./java/jdk1.5.0_06/demo/applets/Blink/Blink.java
./java/jdk1.5.0_06/demo/applets/CardTest/CardTest.java
...
```

Rechercher les fichiers n'appartenant pas à l'utilisateur

Il peut parfois être utile de rechercher les fichiers n'appartenant pas à l'utilisateur, en vue de corriger un problème rencontré avec une application (par exemple, un fichier peut appartenir à root au lieu d'appartenir à l'utilisateur ; ce dernier risque de ne pas avoir de droits dessus, ce que peut alors provoquer une erreur dans une application cherchant à modifier le dit fichier).

Pour ce faire, il suffit d'exécuter là commande suivante, où « utilisateur » est à remplacer par votre nom d'utilisateur :

```
find /home/utilisateur ! -user utilisateur
```

ou bien, en utilisant des variables :

```
find $HOME ! -user $USER
```

Pour avoir davantage d'informations sur les fichiers ainsi trouvés, vous pouvez ajouter l'option `ls` :

```
find $HOME ! -user $USER -ls
```

-exec - Exécuter une commande

La commande **find** permet d'effectuer toute sorte d'action avec les fichiers trouvés.

Une action très utile est “`-exec`” qui permet d'exécuter une commande sur les fichiers sélectionnés.
La syntaxe de `exec` est particulière car il faut pouvoir fournir le nom du fichier trouvé.

À la suite de la commande find habituelle, la syntaxe est :

```
 find /chemin/du/fichier/ <option> <caractéristique fichier> -exec commande {} \;
```

1. La paire d'accolade est automatiquement remplacée par le nom du fichier,
2. et le point-virgule final permet de marquer la fin de la commande.

Au cas où plusieurs fichiers sont traités *dans un même répertoire*, pour éviter une relance de la commande après chaque fichier trouvé, remplacer le `;` (point-virgule) final par le signe positif : `+`.

Par exemple ainsi :

```
find /home/mon_user/test/ -type f -exec echo {} \+
```

Si vous utilisez cette option, veillez bien à ce que votre variable d’environnement **$PATH** ne contienne pas une référence au répertoire courant « . », sinon un pirate pourrait lancer toutes les commandes qu’il souhaite en mettant un fichier au nom adéquat dans les répertoires où vous allez lancer un -execdir.
De la même manière, évitez les références vides ou les noms de répertoires exprimés en relatif

dans **$PATH**.
Cette commande est difficile à utiliser sur certains shell car ceux-ci donnent une signification particulière des caractères comme l'accolade ou le point-virgule.

Avec **Bash**, la *paire d'accolades sans espace* (`{}`) ne doit pas être protégée, au contraire du *point-virgule* qui doit être échappé à l'aide d'un backslash: `\;`.

Voici par exemple comment on peut compter le nombre de lignes de chaque fichier de code Python de ce site:

```
find developpement/django/certif -name '*.py' -exec wc -l {} \;
```

résultat de la commande précédente

```
 1 developpement/django/certif/__init__.py
 0 developpement/django/certif/acronym/__init__.py
48 developpement/django/certif/acronym/models.py
82 developpement/django/certif/acronym/tools.py
13 developpement/django/certif/acronym/urls.py
42 developpement/django/certif/acronym/views.py
.../...
```

1. Ici la commande **find** est utilisée avec l'option `-name` pour ne sélectionner que les fichiers se terminant par "`.py`" (extension de Python).
2. La commande "`wc`" (qui compte le nombre de ligne avec -l) est invoquée à l'aide "`-exec`" sur chacun de ces fichiers.

Il existe d'autres variantes de l'exécution de commande comme l'option `-execdir` qui exécute la commande à partir du répertoire du fichier.

Comme d'habitude vous avez aussi le :

```
man find
```

Entièrement disponible à votre curiosité ! ;-)

Revision #4
Created 18 December 2024 20:49:01 by Nicolas
Updated 14 February 2025 15:13:36 by Nicolas