Commande Sed

La commande sed fonctionne en mode flux ligne par ligne, le flux d'entrée peut-être aussi bien un fichier, qu'un pipe.
Sed peut s'utiliser de deux façons :
- La méthode "classique", qui consiste à appliquer la commande sur le flux d'entrée, et de récupérer le flux de sortie. Par exemple, on applique sed sur un fichier et on redirige la sortie sur un autre fichier
- La méthode "directe", avec l'option sed -i, qui applique la commande directement sur le fichier passé en entrée.
Il existe encore deux manières de passer un script à sed :
- Ecrire le script directement dans la ligne de commande, avec l'option sed -e. On séparera les commandes avec des ";" c'est la façon "uniligne", souvent très pratique
- On peut passer à sed un fichier externe "monscript.sed" par exemple, qui contient le script, grâce à sed -f script_files. Cela assure une meilleure lisibilité pour les gros scripts et permet aussi de réutiliser un script.

Adressage par ligne

Prenons un fichier d'exemple suivant :

QCMa9JuqAnaOH8gY-image-1560345499040.png

Si l'on utilise un sed '4d; 7d' test.txt, on obtiendra

rd2Na5TLRrURBkDd-image-1560345519340.png

On a supprimé la ligne 4 et la ligne 7. (d=delete)
Si l'on a utilisé une "," avec la commande sed '4,7 d' test.txt

WERGl8mOnrtme3LF-image-1560345544452.png

On a supprimé les lignes comprise entre la ligne 4 et la ligne 7
Adressage par motif
ex : sed '/system/!d' test
Notre regex contient le schéma de recherche 'system', et les "/" permettent de délimiter le schéma. Cette regex ne sert pas à grand chose, car on recherche un mot précis, mais nous allons voir qu'avec, on peut chercher et récupérer des syntaxes beaucoup plus complexes.
Le langage contient bon nombre de caractères avec un sens spécifique qui permettent des recherches très pointues. Tout d'abord nous allons parler des fonctions qui permettent de définir des traitements de recherche pour la commande sed. Si l'on utilise, par exemple, l'option '!d', le sed ne nous sortira que les ligne contenant le pattern recherché, nous placerons les options en fin de regex.

Pour notre fichier de test qui contient la liste des utilisateurs :

tBSsi6KWnlvPMaNy-image-1560345569548.png

sed ‘/system/!d’ nom_fichier
donnera la sortie suivante :

4ulqpVerDIPvvyou-image-1560345588711.png

Les différentes fonctions de la commande sed où l’on peut utiliser des motifs sont :
(Attention, sed peut être utilisé sans motif pour sélectionner des lignes ou des ensembles de ligne, je vous laisse vous reporter au man, pour plus de détails.)

Fonction	Exemple	Description
d	sed ‘/system/d’ fichier	Supprime de la liste toutes les lignes qui correspondent au pattern de recherche
!d	sed ‘/system/!d’ fichier	Supprime de la liste toutes les lignes qui ne correspondent pas au pattern de recherche
-n et p	sed -n ‘/system/p’ fichier	Le mode silencieux -n permet de dire à la commande sed de n’afficher aucune lignes, seules les lignes intéressantes seront affichées grâce à l’option print “p”
-n et =	sed -n ‘/system/=’ fichier	Permet d’obtenir les numéros de lignes
-l et l	sed -n ‘/system/l’ fichier	Affiche la ligne sélectionnée avec en plus les caractères de contrôles en clair avec leur code ASCII
,	sed -n ‘/network/,/resolve/’ fichier	On recherche un interval, donc toutes les lignes comprises entre les 2 motifs qui sont séparés par la virgule
q (quit)	sed ‘/system/q’ fichier	Va lister le fichier jusqu’à trouver le pattern et s’arrêter
-r	sed -r	Permet d’utiliser la syntaxe étendue
-e	sed -e	Permet d’écrire un script directement en ligne de commande
s (substitution)	sed -re 's/^# *//' fichier	Permet de substituer le premier motif par le deuxième, ici, on remplace les # suivis d’espace en début de ligne par rien
y (translitération)	sed -re ‘y/éèê/eee/’ fichier	Permet de remplacer certains caractères par d’autres caractère

Revision #1
Created 2019-10-31 12:07:41 UTC by Cécile
Updated 2019-10-31 12:08:08 UTC by Cécile