CS130 – Introduction au système Unix – TP n°2

ESISAR – 2014/2015

1 Complétion du shell

Créez un répertoire Q1 dans votre répertoire, et entrez-y. Créez un fichier titi. Tapez ensuite la commande

`$ cat ti` TAB

(où TAB représente la touche TAB du clavier). Comment le shell a-t-il pu réaliser la complétion ?

Créez un fichier `tito`. Tapez la même commande, que se passe-t-il ? Appuyez à nouveau plusieurs fois sur TAB. Expliquez.

Tapez maintenant

`$ ./ti` TAB

Que se passe-t-il ? Pourquoi ? Comment faire pour que la complétion fonctionne la aussi ?

Résumez comment fonctionne la complétion mise en œuvre par le shell.

2 Liens et droits

Créez dans votre répertoire privé :

– un répertoire secret avec aucun droit,

– un fichier toto lisible par tous dans ce répertoire,

– dans votre répertoire privé, un lien nommé tutu vers secret/toto, lisible par tous.

Expliquez le résultat des commandes suivantes :

```
$ cat secret/toto
$ cat tutu
```

La complétion du shell fonctionne-t-elle pour la première commande ? Pour la deuxième ? Pourquoi ?

3 La commande tar

Récupérez le fichier sudoku.tgz depuis la page chamilo du cours CS110. Listez le contenu du fichier, sans puis avec l'option v.

Extrayez les fichiers, retirez le droit d'écriture sur tous les fichiers du répertoire Sudoku, puis créez un fichier `sudoku_nw.tgz` contenant tous ces fichiers. Effacez le répertoire Sudoku et tout son contenu puis extrayez les fichiers de `sudoku_nw.tgz`. Vérifiez les droits.

Dans la page de manuel de tar, cherchez (à l'aide de `/`) les options `-j` et `-a` et expérimentez les.

4 Commandes head et tail

Parcourez les pages de manuel des commandes `head` et `tail`. Utilisez-les pour afficher les lignes 20 à 37 du fichier gsudoku.alg.

5 Variables

Quelle est la valeur de la variable LANG ? Utilisez là pour obtenir la page de manuel de `passwd` en français et en anglais.

6 Commande find

1. Recherchez, en une seule commande, tous les fichiers qui contiennent dans leur nom la chaîne de caractères « gre » dans les répertoires /bin et /usr/bin.
2. Recherchez, en une seule commande, tous les fichiers sous l'arborescence `/usr/share/man` dont le nom commence par `passwd.1`. Pouvez-vous ouvrir ces fichiers avec `less` ? Que dit la commande `type` pour ces fichiers ? Ouvrez-les avec `zless`. Que contiennent ces fichiers ?
3. Recherchez dans votre dossier personnel tous les fichiers qui ont été modifiés depuis moins de 2 jours.
4. Recherchez, dans les répertoires /bin et /usr/bin, en une seule commande, tous les fichiers normaux exécutables par tous et affichez les informations relatives à ceux-ci sous cette forme :

   ```
   N° d'inoeud        droit en octal       nom du fichier avec son chemin
   ```

5. Recherchez dans votre répertoire personnel les fichiers qui ont une taille inférieure à 250 ko.

7 Commande grep

1. Recherchez, dans tous les fichiers .h situés dans le répertoire /usr/include, toutes les lignes qui contiennent une succession de trois caractères identiques compris entre a et e.

2. Récupérez le fichier `algo_linux64-20140923.tgz` depuis la page chamilo du cours CS110. Extrayez **uniquement** le répertoire `Algo/support` de l'archive.
   Pour que les caractères accentués des fichiers alg s'affichent correctement, travaillez dans un nouveau terminal que vous créerez avec la commande xterm -en iso8859-1.
   - Affichez la liste des fonctions définies dans tous les fichiers .alg du répertoire support. Supprimez l'affichage du nom de fichier en début de ligne.
   - Affichez la liste des fichiers .alg dans lequel on utilise une pile.
   - Que fait la commande `grep -l file Algo/support/*.alg | xargs grep -l pour` ? (l'option est "tiret elle", pas tiret un)

8 Analyse d'un fichier de log

Avant de commencer, téléchargez le fichier chat.log depuis la page chamilo du cours. Pour chaque question, la réponse doit être trouvée en une et une seule ligne ! Donnez les commandes pour connaître :

1. le nombre de lignes de ce fichier de log
2. le nombre de messages échangés (`MESSAGE_SENT`)
3. le nombre de lignes de chaque catégorie (`LOGIN_SUCCESS`, `LOGIN_FAIL` etc)
4. le nombre de messages de chaque type, avec affichage par nombre de messages décroissant
5. qui écoute le plus les autres
6. le hit parade des trois plus gros bavards, avec les lignes numérotées 1,2 et 3.
7. la minute durant laquelle il y a eu le plus de messages échangés
8. qui se parle le plus à lui-même
9. qui a le plus grand score nombre de messages envoyés + nombre de messages reçus
   - version 1 : en comptant deux fois le message si la personne se parle à elle-même
   - version 2 : en comptant une seule fois le message si la personne se parle à elle-même
10. Transformer les lignes `MESSAGE_SENT` pour obtenir des messages de la forme :

```
20120930 15:00 MESSAGE_SENT mcgee --> dinozzo
20120930 15:00 MESSAGE_SENT dinozzo se parle à lui-même
```

9 Bonus : manipulations sur les processus

1. La commande ps permet d'afficher l'état des processus en cours. Exécutez cette commande. Expliquez le résultat.
2. Lancez la commande cat, écrivez "coucou", puis appuyez sur Return. Que se passe-t-il ? Interrompez cette commande. Relancez cat et suspendez cette commande. Pour chaque cas, listez les processus et vérifiez que le processus cat est suspendu ou qu'il n'existe plus.

3. Lancez la commande gedit en arrière plan depuis un terminal. Créez un autre terminal et lancez à nouveau la commande gedit en arrière plan. L'option `-x` permet de voir tous vos processus.

   La commande kill permet d'envoyer un signal à un processus. Lisez le manuel de cette commande. À partir de chaque terminal, détruisez les processus gedit lancés depuis l'autre terminal.

4. Dans un terminal, identifiez le processus correspondant au shell qui tourne dans ce terminal. Détruisez-le avec kill. Que se passe-t-il ?

5. Le deuxième étudiant du binome se connecte dans un nouveau terminal comme lors du premier TP (ssh etu2@localhost). Le premier lance la commande cat dans un terminal, puis obtient son numéro (PID) depuis un autre terminal. Connaisant ce PID, etu2 peut-il détruire le processus ? Pourquoi ?