

Systèmes avancés : Commandes principales & processus Shell sous UNIX

2016

Karim Hammoudi

Faculté des Sciences et Techniques
Université de Haute-Alsace

Plan

- Présentation des systèmes
- Modes de travail
- Systèmes de fichiers
- Script shell

Sommaire

- Présentation des systèmes
- Modes de travail
- Systèmes de fichiers
- Script shell

Présentation des systèmes

Un système est un ensemble de programmes qui permettent à un utilisateur, un programme ou une application d'accéder aux différents composants matériels et logiciels d'un ordinateur.

- **Noyau** : gère les ressources matérielles et les différents objets logiciels (les programmes en execution, les fichiers).
- **Ensemble de commandes** : pour réaliser des taches de gestion du materiel et des objets logiciels.
- **Interfaces utilisateurs** : pour la commication avec l'ordinateur (IHM)

Présentation des systèmes

- **Unix** est un système qui permet la connexion de plusieurs postes de travail à l'ordinateur -> **Multi-utilisateur**
- Chaque utilisateur peut démarrer simultanément plusieurs programmes -> **Multi-tâche**
- **Exemple de système UNIX**, une distribution Linux :
 - Linux est le noyau qui a été développé par Linus Thorwald.
 - L'ensemble des commandes proviennent du projet GNU (début de l'OS libre).
 - L'ensemble des interfaces graphiques proviennent des projets KDE ou Gnome.

Sommaire

- Présentation des systèmes
- Modes de travail
- Systèmes de fichiers
- Script shell

Modes de travail

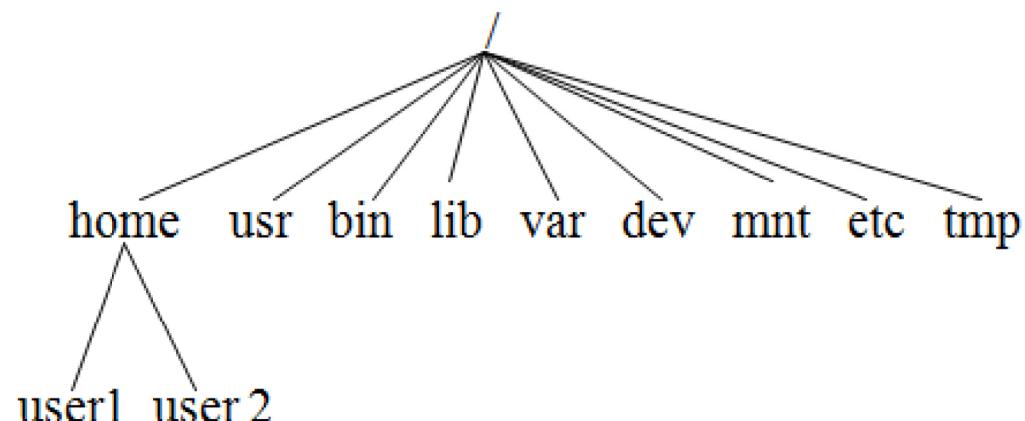
- **Dépendant du profil, des usages et des sensibilités de l'utilisateur**
 - développeur, grand public, administrateur...
- **Basculer du mode graphique au mode texte/terminal de commandes et inversement**
 - Ctrl - Alt - F7, puis Alt Fi où i est le numéro du terminal

Sommaire

- Présentation des systèmes
- Modes de travail
- Systèmes de fichiers
- Script shell

Systèmes de fichiers

- **Généralités :**
 - Se déplacer dans l’arborescence
 - Manipulation des fichiers
- Le système de fichiers se représente comme une arborescence avec la racine située au sommet.
- **Encapsulation** : fichier (feuille) vs répertoire (noeud)



Systèmes de fichiers

Répertoire	description
/	Répertoire "racine", point d'entrée du système de fichiers
/boot	Répertoire contenant le noyau Linux et l'amorceur
/bin	Répertoire contenant les exécutables de base, comme par exemple cp, mv, ls, etc.
/dev	Répertoire contenant des fichiers spéciaux nommés <i>devices</i> qui permettent le lien avec les périphériques de la machine
/etc	Répertoire contenant les fichiers de configuration du système
/home	Répertoire contenant les fichiers personnels des utilisateurs (un sous-répertoire par utilisateur)
/lib	Répertoire contenant les librairies et les modules du noyau (/lib/modules)
/media	Répertoire contenant les « points de montage » des médias usuels : CD, DVD, disquette, clef USB
/root	Répertoire personnel de l'administrateur
/sbin	Répertoire contenant les exécutables destinés à l'administration du système
/tmp	Répertoire contenant des fichiers temporaires utilisés par certains programmes
/usr	Répertoire contenant les exécutables des programmes (/usr/bin et /usr/sbin), la documentation (/usr/doc), et les programmes pour le serveur graphique (/usr/X11R6).
/var	Répertoire contenant les fichiers qui servent à la maintenance du système (les fichiers de journaux notamment dans /var/log)

Systèmes de fichiers

- **Manipulation des fichiers :**
 - mkdir, rmdir
 - touch
 - pwd
 - cp
 - mv
 - rm
 - ls
 - clear

Systèmes de fichiers

- **Manipulation des fichiers :**

- man
- --help
- cat

Ex : affichage de fichier(s concaténés)

> cat filename1 filename2

- grep

Ex : recherche d'un motif dans un fichier

>grep "motif" filename

- ...

Systèmes de fichiers

- **Exercice “se repérer et s’organiser” :**
 1. Listez les répertoires /bin et /home
 2. Cherchez dans le manuel les explications concernant l'option -t de la commande ls
 3. Que réalise la commande « ls /bin/b* » ?
 4. Donner une suite de commandes pour organiser votre "répertoire de connexion" en sous-répertoires d'après vos "thèmes" de travail.
 5. Créez un fichier contenant un petit texte via un éditeur. Comment visualiser son contenu ?

Systèmes de fichiers

- **Exercice “gérer l'accès au contenu” :**
 - Créer un répertoire « workspace » et déplacez vous dans celui-ci
 - Créer un fichier vide « mon_fichier.txt » et examiner ses permissions.
 - Les droits d'accès apparaissent alors comme une liste de 10 symboles. Expliquer les différences de symbole entre ceux du répertoire créé et ceux du fichier créé.

En complément : <https://doc.ubuntu-fr.org/permissions>

- Donnez successivement au fichier créé les droits nécessaires pour que vous (le propriétaire puissiez:
 - Lire, modifier et exécuter votre fichier
 - Lire, modifier mais pas exécuter votre fichier
 - Lire mais pas modifier ou exécuter votre fichier
- Accorder toutes les permissions au propriétaire et assigner la lecture seule pour le groupe

Sommaire

- Présentation des systèmes
- Modes de travail
- Systèmes de fichiers
- **Script shell**

Script shell

Le shell est :

- l'interpréteur de commande ;
- l'interface entre l'utilisateur et les commandes ;
- un langage de programmation (interpréteur), il permet donc de réaliser de nouvelles commandes ;
- un gestionnaire de processus ;

Pour afficher le shell utilisé : echo \$SHELL

Script shell

Plusieurs variantes :

- Le Thompson shell (1971)
- Le Bourne shell (bsh)
- Le C shell (csh) -> la syntaxe s'inspire du C
- Le Korn shell (ksh)
- ...
- Le Bourne again shell again (bash) est le shell par défaut de la plupart des distributions Linux mais aussi celui du terminal de Mac OS X

Script shell

- Un script shell permet d'automatiser une série d'opérations.
- Il se présente sous la forme d'un fichier contenant une ou plusieurs commandes qui seront exécutées de manière séquentielle.
- Dans le cas de commandes placées dans un fichier ascii, celui-ci doit être rendu exécutable, ce fichier exécutable est également appelé script shell.
- Un script shell doit avoir des droits d'exécution pour être employé.

Script shell

- Création et exécution d'un script shell
 - Créer et éditer un nouveau fichier.sh
 - `$ gedit essai.sh`
 - Indiquer le type du shell utilisé par le script (bash)
 - `#!/bin/bash`
 - Écrire les commandes à exécuter
 - Enregistrer et quitter gedit
 - Donner les droits d'exécution au script
 - `chmod +x essai.sh`
 - Exécution du script
 - `./essai.sh`

Script shell

- Afficher et manipuler des variables :
 - Créer une variable :
 - `Nom_variable='valeurdelavariable'`
 - Afficher une variable :
 - `echo $Nom_variable`
 - Afficher un message puis la valeur d'une variable :
 - `echo « message » $Nom_variable`
 - Demander une saisie par l'utilisateur :
 - `read variable1 variable2 variable3`
 - Afficher les variables saisies :
 - `echo $variable1 $variable2 $variable3`
 - Dans les scripts shell, toutes les lignes commençant par `#` sont des commentaires.

Script shell

- Exemple d'entrée/sortie :

```
#!/bin/bash  
echo "Bonjour. Comment vousappelez-vous ?"  
read nom  
echo " très bien. Je vous souhaite, $nom, de passer une  
bonne journée."
```

- Obtenir une trace de l'exécution (ce qui peut être très utile en cas de bug). Il faut lancer le script en utilisant l'option "-x" :

```
bash -x essai.h
```

Script shell

- Exemple d'opérations arithmétiques :

- Utiliser l'instruction « let »

```
#!/bin/bash
```

```
a=5
```

```
b=3
```

```
let « c=a*b »
```

```
echo $c
```

```
let « a*=c »
```

```
echo $a
```

- Que vaut a?
 - La puissance : **

Script shell

- Structure de contrôle alternative « if »

```
if [ test ]
then
    Traitement 1
elif [ autre test ]
then
    Traitement 2
elif [ encore_autre _test ]
then
    Traitement 3
else
    Traitement 4
fi
```

- Ne pas coller les crochets aux tests
- Comparaison du contenu d'une variable avec un entier
 - Utiliser l'instruction « **-eq** » (égal à), « **-ne** » (différent de)

Script shell

- Structure de contrôle itérative « while »

while [[test]]

do

traitement

done

Script shell

- Structure de contrôle itérative (déterministe)

« for » :

- **for i in `seq 0 4`;**
do
echo \$i
done
- « ` » touche 7

Script shell

- Structure de contrôle itérative (déterministe)
« for » proche du C avec les shell modernes :
 - **for ((i=0;\$i<5;i++))**
 - do**
 - echo \$i**
 - done**

Script shell

- **Exercice “Simuler un système d’authentification”:**
 - Objet : Reproduisez le système d’authentification de votre téléphone mobile
 - Prédéfinir un code PIN dans votre script
 - Prédéfinir l’adresse email de l’admin « vous »
 - Demander la saisie du code PIN à l’utilisateur
 - Effectuer la vérification
 - Bloquer l’authentification au bout de 3 saisies invalides
 - Indiquer à l’utilisateur d’envoyer une demande de PIN à « l’email de l’admin »
 - Ouvrir un navigateur qui pointe sur <https://e-partage.uha.fr/>
 - Donner le choix à l’utilisateur de choisir d’ouvrir un client de messagerie au lieu de sa boîte en ligne

Script shell

- **Exercice “Composition d’un script”:**
 - En composant avec les structures de contrôles et les mécanismes précédemment présentées, proposer un script fonctionnel et pertinent permettant d’illustrer :
 - Soit une commande, un système ou une application existante (ex : authentification d’un mobile)
 - Soit un système « original » pouvant répondre à de nouveaux besoins identifiés.

- Maher Rebai. *Généralités sur le système UNIX.*
- *Introduction aux scripts shell*, https://doc.ubuntu-fr.org/tutoriel/script_shell
- *Reprenez le contrôle à l'aide de Linux !*,
<https://openclassrooms.com/courses/reprenez-le-controle-a-l-aide-de-linux>
- Sylvain Cherrier. *Unix : Principes généraux*
- Jean-Marie Rifflet et Jean-Baptiste Yunès. *Programmation et communication UNIX*, Dunod, 774p., 2003.
- *La documentation en ligne de l'interpréteur bash :*
<http://www.gnu.org/software/bash/manual/bashref.html>