Module ITC313 - Informatique

Partie UNIX / Shell

TD1 Manipulation de fichiers et arborescences

Benoît Darties - benoit.darties@u-bourgogne.fr
Université de Bourgogne

Année universitaire 2016-2017

---

1 Gestion des fichiers sous UNIX

Les exercices de cette section sont destinés à assimiler les concepts élémentaires sur le stockage des fichiers sur un système UNIX-like, et découvrir les commandes usuelles de manipulation de fichier.

Exercice 1 : *Nom et format de fichier*

L'objectif de cet exercice est de montrer si l'extension d'un fichier est un élément déterminant dans le nom d'un fichier, si cette extension définit forcément le type du fichier ou non, et exhiber son vrai rôle.

On dispose d'un fichier nommé `image.jpg` encodé format JPEG.

1. Quelle est la commande qui permet de renommer ce fichier avec le nom `photo.gif`
2. Est-ce que renommer ce fichier en `image.gif` re-encode l'image initialement au format JPEG au format GIF ? Vérifiez votre réponse en utilisant la commande `file image.gif`
3. Quelle crédibilité peut-on alors accorder à l'extension d'un fichier ? L'extension d'un fichier est-elle alors un indicateur fiable du type d'un fichier, ou simplement un indicateur que l'on suppose valide mais sans certitude ?

Pour aller plus loin : l'extension permet essentiellement au système d'exploitation de définir le programme par défaut à utiliser pour ouvrir le fichier. A chaque extension est associé un - ou plusieurs - programmes supposés capables de lire le contenu du fichier. Lorsque l'on clique sur un fichier avec l'extension `jpg`, le programme associé (généralement un lecteur d'images) est lancé, et ce dernier tentera de lire le fichier, sans forcément se fier à l'extension, mais au contenu concret du fichier.

Exercice 2 : *Comprendre le fonctionnement du stockage d'un fichier sous forme de blocs*

L'objectif de cet exercice est de comprendre comment un fichier est stocké sous forme de blocs, mais également que le stockage des adresses de ces blocs entraine une consommation de blocs supplémentaires. Cet excédent est à l'origine de la différence entre la taille réelle d'un fichier, et la taille occupée sur le disque dur. Par ailleurs, la limitation du nombre d'adresses de blocs que l'on peut stocker entraine une limite sur la taille maximale d'un fichier sur ce système. Cette taille maximale dépend également d'autres paramètres.

Un système de fichier possède les caractéristiques suivantes :

— la table des inodes possède cinq champs d'adressage direct ;
— la table des inodes possède trois champs d'adressage à un niveaux d'indirection ;
— la table des inodes possède quatre champs d'adressage à deux niveaux d'indirection ;
— la taille des adresses de bloc est de 32 bits ;
— les blocs de données sont de 2ko.

1. Combien peut-on stocker d'adresses de blocs en utilisant seulement les adressages directs ?
2. Combien peut-on stocker d'adresses dans un bloc ?
3. Combien peut-on stocker d'adresses de blocs en utilisant seulement les adressages à un niveau d'indirection ?
4. Combien peut-on stocker d'adresses de blocs en utilisant seulement les adressages à deux niveaux d'indirection ?
5. Quelle est la taille maximale de fichier que l'on peut stocker sur ce système ?
6. On souhaite stocker un fichier de 100 Mo sur ce système de fichier.
   (a) Combien de blocs de données doivent être utilisés pour stocker ce fichier ?
   (b) Combien d'adresses de blocs de données faut-il alors stocker ?
   (c) Combien de blocs intermédiaires sont nécessaires pour stocker ces adresses de blocs ?
   (d) Quelle est la taille occupée en nombre de blocs, puis en octets, du fichier

Pour aller plus loin : désormais, les systèmes de fichiers sont définis de sorte à ce que la taille limite d'un fichier dépende d'autres facteurs que le nombre d'adresses de blocs maximal que l'on peut stocker. Il est important de comprendre que ces adresses de blocs restent les éléments primordiaux pour reconstituer un fichier. Si ces adresses disparaissent, il est quasiment impossible de reconstituer le fichier, même si les blocs de données sont encore présents. Les ransomware sont des virus qui chiffrent l'ordinateur d'une victime et obligent cette dernière à payer une rançon pour le déchiffrement : dans la pratique l'ensemble du disque n'est pas réellement chiffré (trop long), mais seul le contenu de la table d'attribut des fichiers et les blocs intermédiaires contenant les adresses des blocs de données sont chiffrés.

Exercice 3 : *Arborescence de fichiers*

L'objectif de cet exercice est de manipuler les commandes élémentaires de création et manipulation des fichiers sous UNIX, et de comprendre les principaux droits sur ces fichiers, quels sont ces droits par défaut, comment les modifier, et quelles restrictions ils engendrent. Sont également abordées quelques subtilités sur l'utilisation du caractère 'espace' dans une ligne de commande.

Trois utilisateurs `Anakin`, `Ben` et `Chewbacca` sont authentifiés sur une machine. `Anakin` et `Ben` font partie du même groupe d'utilisateurs `Jedi`, qui est également leur groupe par défaut . `Chewbacca` ne fait pas partie du même groupe, puisqu'il est dans le groupe `Wookie`. On rappelle que généralement - sauf configuration particulière - lorsqu'un fichier est créé, son propriétaire est l'utilisateur qui crée ce fichier, et que le groupe associé à ce fichier est le groupe par défaut de l'utilisateur. Lorsque anakin lance la commande `umask`, le résultat affiché est `0022`, signifiant que les fichiers réguliers seront créés avec les droits par défaut `644`, et les répertoires avec les droits par défaut `755`. `Anakin` saisit suite de commandes suivante depuis son répertoire personnel :

```
1 mkdir MP3
2 mkdir MP3/Metallica
3 mkdir "MP3/Dying Fetus"
4 cd "MP3/Dying Fetus"
5 touch "Kill Your Mother, Rape Your Dog.mp3"
6 touch "You Blood Is My Wine.mp3"
7 touch ../Metallica/One.mp3
```

1. Pourquoi `Anakin` utilise-t'il parfois des guillemets et parfois non ? Quel serait l'effet de la ligne 5 si elle avait été saisie sans les guillemets ?
2. Dessinez l'arborescence de répertoires créée par ces commandes, en prenant le répertoire `MP3` comme racine, et en précisant pour chaque fichiers les droits qu'il a.

`Anakin` saisit ensuite les commandes suivantes depuis son répertoire personnel :

```
1 chmod 000 "./MP3/Dying Fetus/Kill Your Mother, Rape Your Dog.mp3"
2 chmod 735 "./MP3/Dying Fetus"
3 chmod o-r "./MP3/Dying Fetus/Your Blood Is My Wine.mp3"
4 chmod ug-w,o+w ./MP3/Metallica
```

On suppose dans les questions suivantes que les répertoires personnels de chaque utilisateur ont les droits d'exécution `755`.

3. Depuis son répertoire personnel, `Ben` peut-il exécuter chacune des commandes suivantes ? Chaque question est ici indépendante, et ne tient pas compte de la commande précédente qu'elle ait été correctement exécutée ou non.
   (a) `cd "~anakin/MP3/Metallica"`
   (b) `cd "~anakin/MP3/Dying Fetus"`
   (c) `ls "~anakin/MP3/Dying Fetus/*.mp3"`
   (d) `cp "~anakin/MP3/Dying Fetus/Kill Your Mother, Rape Your Dog.mp3" ./`
   (e) `cp "~anakin/MP3/Dying Fetus/Your Blood Is My Wine.mp3" ./`
   (f) `cp "~anakin/MP3/Metallica/one.mp3" ./`
   (g) `rm "~anakin/MP3/Dying Fetus/Kill Your Mother, Rape Your Dog.mp3"`
   (h) `rm "~anakin/MP3/Dying Fetus/Your Blood Is My Wine.mp3"`
   (i) `rm "~anakin/MP3/Metallica/one.mp3"`
   (j) `mv "~anakin/MP3/Dying Fetus/Kill Your Mother, Rape Your Dog.mp3" ./`
   (k) `mv "~anakin/MP3/Dying Fetus/Your Blood Is My Wine.mp3" ./`
   (l) `mv "~anakin/MP3/Metallica/one.mp3" ./`
   (m) `mv "~anakin/MP3/Dying Fetus/Kill Your Mother, Rape Your Dog.mp3" ~anakin/MP3/Metallica/one.mp3`
4. Même question avec l'utilisateur `Chewbacca`.

Exercice 4 : *Situations de blocage*

Dans cet exercice, nous poussons la gestion des droits dans des configurations insolites, qui peuvent présenter certains comportement étonnants. Il est possible de reproduire les configurations sur machine par binôme, en utilisant deux comptes sur une même machine.

`Jacen` et `Jaina` sont deux utilisateurs standards d'un système. Ils possèdent chacun un compte utilisateur avec un espace personnel avec les droits `0755`. Pour chacune des situations suivantes, indiquez s'il est possible de créer une telle situation, et si oui quels enchaînement de commandes peuvent mener à cette situation :

1. `Jacen` peut consulter et modifier le contenu du fichier nommé `coralliens.txt` dont il est propriétaire, mais ne peut pas le supprimer sans l'aide de l'administrateur ou de `jaina`
2. `Jacen` ne peut pas supprimer (sans l'aide de l'administrateur ou de `jaina`) le répertoire nommé `centerpoint`, alors que ce dernier se situe dans son propre répertoire personnel et qu'il est le propriétaire de ce fichier.
3. `Jacen` possède dans son répertoire personnel un sous-répertoire nommé `Tenel Ka` qui lui appartient et dont les droits sont `0700`. Dans ce sous-répertoire est contenu un fichier `Allana.txt` sur lequel il n'a aucun droit de lecture ou d'écriture, mais que `Jaina` peut arriver à modifier sans avoir à demander à `Jacen` de modifier les droits du répertoire `Tenel Ka`

Pour aller plus loin : bien que peu communes, les situations mises en évidences dans ces exemples sont pourtant nécessaires d'être comprises, dans la mesure où elles permettraient d'exploiter une ressource au delà des limites fixées par un administrateur, par exemple si chaque utilisateur a un quota d'espace disque sur sa session.

2 Commandes Shell

Exercice 5 : *Utilisation de caracteres Joker*

En utilisant les caractères joker `*` (n'importe quelle suite de caractères, même vide), `?` (exactement un caractère), et `[..]` (un caractère parmi ceux de l'ensemble), écrivez la commande qui permet d'afficher :

1. la liste des fichiers dont l'extension est `.pdf`
2. la liste des fichiers dont l'extension est `.gif` ou `.GIF`
3. la liste des fichiers qui commencent par une voyelle
4. la liste et le détail (option `-l`) des fichiers du répertoire `/usr/bin` qui commencent par la lettre c
5. la liste des fichiers du répertoire `/usr/bin` dont le nom est composé d'au moins trois caractères, le premier faisant parmi de l'ensemble les lettres a, t et s, et le troisième compris entre les lettres a et f.
6. la liste des fichiers dont l'extension ramenée à des minuscules est `.gif` (par exemple : `.gif`, `.GIF`, `.Gif`, `.gIF`, ...)

note : la commande `ls` devra être utilisée avec l'option `-d` pour éviter que l'on liste le contenu des sous-répertoires si ces derniers répondent aux critères de filtres.