


Introduction aux systèmes d'exploitation (IS1)

TP n° 3 : liens, i-nœuds et droits

Le but de ce TP est, d'abord, d'apprendre à personnaliser l'environnement du travail, puis de vous familiariser avec les notions de *lien* et d'*i-nœud*, et de commencer à explorer le système des droits d'accès aux fichiers.

Modalités de rendu Au cours du TP, collectez dans un fichier `reponses_TP3.txt` les réponses aux questions marquées par le symbole  (et seulement à celles-ci!), ou les commandes utilisées pour y répondre (mais pas les résultats qu'elles produisent). Dans le fichier `reponses_TP3.txt` insérez en premier votre nom et prénom (ou noms et prénoms si vous travaillez à plusieurs), et déposez le fichier en fin de TP sur Moodle.



Personnaliser son environnement

Certaines lignes de commande sont longues à taper, notamment lorsqu'il y a des options. Une commande permet de pallier cet inconvénient en créant ses propres commandes :



« alias »

- avec un argument de la forme `ma_commande=commande_complète`, définit (ou redéfinit) la commande « `ma_commande` ».
- Attention, si `commande_complète` contient des espaces (par exemple, s'il y a des options), il faut alors la délimiter avec des guillemets.
- sans argument, liste tous les raccourcis qui ont été définis.

Exercice 1 – définir ses propres commandes et changer une commande

1.  Créer une commande « `li` » qui liste les fichiers d'un répertoire, un par ligne.
2.  Créer une commande « `la` » qui liste tous les fichiers d'un répertoire (y compris les fichiers cachés commençant par un point).

Si `ma_commande` existe déjà, son comportement est redéfini et remplacé par celui décrit : cela permet de rajouter systématiquement des options à une commande.

3.  Changer le fonctionnement de la commande « `ls` » pour afficher différemment les fichiers ordinaires et les répertoires.
4. Par défaut, « `rm` » ne demande pas de confirmation lorsque vous tentez de supprimer un fichier. Ceci peut se révéler assez dangereux.  Trouver l'option qui permet de demander confirmation, puis changer le fonctionnement de la commande « `rm` » pour qu'elle demande systématiquement confirmation lors d'une suppression.

5. Tester les raccourcis que vous venez de créer. Puis ouvrir un nouveau terminal et les tester dans celui-ci. Que constatez-vous ?

La commande « `alias` » n'agit que dans le shell qui l'exécute, et pas dans les shells concurrents. Pour rendre le changement pérenne, on peut utiliser des *fichiers de configuration* du shell, par exemple `.bashrc` pour le shell « `bash` ». Un tel fichier est lu à chaque ouverture d'un nouveau shell ; tout changement n'y sera pris en compte qu'en relançant le shell, via l'ouverture d'un nouveau terminal par exemple, ou en forçant la lecture par la commande « `source` » suivie du nom du fichier.

Exercice 2 – fichiers de configuration

Éditer le fichier `.bashrc` situé dans votre répertoire personnel¹ avec l'éditeur « `emacs` » pour rajouter les commandes des exercices précédents (en particulier « `rm` » avec confirmation). Redéfinir également « `cp` » et « `mv` » pour qu'elles demandent confirmation en cas d'écrasement du fichier cible.

Vérifier que les changements sont effectifs à l'ouverture d'un nouveau terminal, et dans les anciens seulement après exécution de « `source .bashrc` ». Faites attention à ne pas effacer de fichiers importants si vous voulez tester votre nouvelle version de « `rm` » !

Liens et i-nœuds

Un système UNIX identifie un fichier par son *i-nœud* (*inode* en anglais pour *index node*) et non par ses noms. Un i-nœud contient toutes les caractéristiques concernant le fichier correspondant : son type (ordinaire ou répertoire), l'adresse du ou des emplacement(s) sur le disque où se trouve son contenu, les droits accordés, la date de dernière modification, etc. Les i-nœuds sont stockés dans une table et identifiés dans cette table par un numéro.

Un répertoire est un fichier qui contient une liste de couples (nom, numéro d'i-nœud), comme dans le tableau ici à droite. Lorsqu'on dit qu'un fichier `fic` « se trouve » dans un répertoire `rep`, cela signifie en fait que `rep` contient un couple associant le nom `fic` au numéro d'i-nœud d'un fichier. On appelle cela un *lien* de nom `fic` vers cet i-nœud.



.	13767923
..	13766888
tp3.sh	13768284
tp3.tex	13768275

Exercice 3 – liens et i-nœuds



Depuis Moodle, télécharger le fichier `tp3.sh` dans le répertoire IS1, puis exécuter dans ce répertoire la commande « `bash tp3.sh` ».

1. À l'aide d'une option de « `ls` », afficher les numéros d'i-nœud des fichiers contenus dans IS1/TP3.

1. Rappel : ce fichier doit déjà contenir la ligne « `export PAGER=less` » depuis le TP n° 1



2.  Renommer IS1/TP3/grosminet en sylvestre dans le même répertoire. Qu'est devenu le numéro d'i-nœud ?
3.  Copier IS1/TP3/sylvestre sous le nom gatto_silvestro dans le même répertoire. Est-ce que les numéros d'i-nœud de deux fichiers sont les mêmes ?
4. Afficher les numéros d'i-nœud des fichiers contenus dans IS1/TP3/Canards. Que remarquez-vous ?

« ln » (**link**) permet de créer un nouveau nom (lien) vers un i-nœud existant : si *ancien* est un nom valide et *nouveau* un nouveau nom, « ln ancien nouveau » crée un nouveau lien de nom *nouveau* vers le même i-nœud que *ancien*.

5.  Créer dans IS1/TP3/Canards un lien loulou vers le même i-nœud que FiFi.
6.  Copier RiRi sous le nom hihihi dans le même répertoire, puis afficher le contenu des fichiers RiRi, FiFi, loulou et hihihi. Les quatre fichiers ont-ils tous le même contenu ? Ont-ils tous le même numéro d'i-nœud ?

On peut modifier le contenu d'un fichier *fic* pour y placer le texte *nouveau_contenu* par la commande suivante :

```
echo nouveau_contenu > fic
```

7.  Remplacer le contenu du fichier hihihi par le texte « *Tout est en ordre, Donald.* ». Ensuite, afficher à nouveau le contenu des fichiers RiRi, FiFi, loulou et hihihi.
8. Toujours de cette manière, remplacer maintenant le contenu du fichier FiFi par le texte « *Hélas, le "Manuel des Castors Juniors" dit exactement la même chose.* » (les guillemets sont-ils affichés ? Que se passe-t-il si on utilise « \ " » ?).  Ensuite, afficher à nouveau le contenu des fichiers RiRi, FiFi, loulou et hihihi. Expliquer.
9. Supprimer FiFi et lister le contenu de IS1/TP3/Canards. Expliquer. Pouvez-vous afficher le contenu de FiFi ? De RiRi ? De loulou ? De hihihi ?



Comment recréer le nom FiFi en gardant son i-nœud original ?

Groupes d'utilisateurs, droits et leurs effets

Chaque fichier (ou répertoire) est la propriété d'un utilisateur particulier. Par défaut, celui-ci appartient à l'utilisateur qui a créé le fichier. Les utilisateurs sont réunis en groupes (par exemple : le groupe des étudiants de L1 informatique, celui des enseignants, celui des administrateurs systèmes). Un utilisateur pouvant faire partie de plusieurs groupes, pour chaque fichier est spécifié le *groupe propriétaire* (c'est-à-dire en tant que membre de quel groupe le propriétaire détient le fichier).

« id » indique vos numéro d'utilisateur, nom et numéro de groupe principal (qui sera par défaut le groupe propriétaire de vos fichiers) ainsi que la liste des groupes auxquels vous appartenez (plus complet que « whoami », donc).

Exercice 4 – connaître ses groupes

🔗 Déterminez votre (ou vos) groupe(s) d'appartenance. Faire de même pour le super-utilisateur root.

Les droits accordés sur un fichier sont définis en séparant les utilisateurs en trois types : le propriétaire, les (autres) membres du groupe propriétaire, et les autres. Il y a trois types de droits : le *droit en lecture* (r), le *droit en écriture* (w) et le *droit en exécution* (x).

l'option « -l » de « ls » permet d'afficher au format dit *long* les méta-données de chaque fichier listé – plus précisément, le contenu de la ligne correspondante de la table des i-nœuds, à l'exception des adresses des blocs utilisés sur le disque. Pour chaque nom de fichier, on obtient les informations suivantes :

- son type (d, -, ...);
- les droits accordés sur le fichier (trois triplets correspondant aux droits rwx des trois types d'utilisateurs : *user, group, other*);
- le nombre de liens vers l'i-nœud;
- son propriétaire et son groupe propriétaire;
- sa taille;
- sa date de dernière modification.

Exercice 5 – afficher les caractéristiques d'un i-nœud

1. À l'aide de « ls -l », repérer les symboles décrivant les droits, le propriétaire et la taille de chaque fichier de votre répertoire IS1/TP3.
🔗 Quel est la taille du plus gros fichier *ordinaire* de ce répertoire ?
2. Copier le fichier ~gibernar/program dans votre répertoire IS1/TP3. Comparer le propriétaire, le groupe propriétaire et les droits des deux fichiers.
🔗 Que constatez-vous ?
3. 🔗 Déterminer combien de liens pointent vers l'i-nœud du répertoire IS1/TP3/Canards/CoffreDePicsou. Comment cela s'explique-t-il ?

Exercice 6 – effets des droits des fichiers ordinaires

Déplacez-vous dans le répertoire IS1/TP3/Toutou.

1. Quel(s) droit(s) avez-vous sur le fichier `tata`? Essayer d'afficher son contenu avec « `cat` ». Essayer ensuite de le modifier grâce à un éditeur de texte (« `emacs` » par exemple), puis à l'aide de la ligne de commande : `echo "nouveau contenu" > tata`. ➤ Qu'en concluez-vous ?
2. Quel(s) droit(s) avez-vous sur le fichier `titi` ? Faire les mêmes expériences. ➤ Qu'en concluez-vous ?
3. Quel(s) droit(s) avez-vous sur le fichier `toto`? Comparer le contenu de ce fichier avec celui de `tata`. Utilisez « `tata` » comme une commande (ou « `./tata` » pour être parfaitement sûr que `bash` n'interprètera pas `tata` autrement que comme « *le fichier de nom tata dans le répertoire courant* »), puis essayer avec « `toto` ». ➤ Qu'en concluez-vous ?
4. Quel(s) droit(s) avez-vous sur le fichier `tutu` ? Essayer de l'utiliser comme une commande. ➤ Qu'en concluez-vous ?