Système utilisateur
.:: Système UNIX ::.

.:: Editeur VI ::.

L'éditeur VI

Bien qu'il existe sous Unix ou Linux de très nombreux éditeurs le seul que vous trouverez partout se nomme VI.

VI est un éditeur mode texte (pas de menu, pas de souris).

Grâce à cela il est utilisable localement comme à distance, quel que soit la liaison réseau et le terminal utilisé.

Il intègre en outre toutes les fonctionnalités d'un éditeur classique et même souvent plus !

L'éditeur VI

VI possède donc DEUX MODES

- le mode COMMANDES pour entrer les combinaisons de touche permettant les fonctions classiques d'un éditeur (copier/coller, supprimer, rechercher, remplacer, sauvegarder...)

- le mode EDITION pour saisir le texte du document.

L'éditeur VI – gestion des modes

Pour passer en mode commande : utiliser la touche ESC Il n'y a aucun risque à l'utiliser plusieurs fois de suite en cas de doute

/!\ en mode commande un caractère quelconque peut avoir des effets destructeurs

L'éditeur VI – gestion des modes

Une fois en mode commande, plusieurs touches permettent de revenir en mode EDITION :

- commande « i » : insertion à la position courante
- commande « a » : insertion à la position suivante
- commande « A » : insertion en fin de ligne
- commande « R » : mode remplacement

L'éditeur VI – gestion des commandes

Quelques autres commandes à connaître par coeur ...

- :q Quitter (si aucune modification)
- :q! Quitter sans sauver
- :w Enregistrer le document
- :w *nom* Enregistrer sous *nom*
- :r *nom* Insère le fichier *nom* à la position courante

L'éditeur VI – gestion des commandes

- r *car* Remplace le caractère courant
- x Efface le caractère courant
- u Undo
- [*num*]dd Efface la/*num* ligne*s* courante*s* (couper)
- [*num*]yy Copie la/*num* ligne*s* courante*s*
- P Coller les lignes mémorisées
- J Supprimer le retour chariot (remonte la ligne suivante)

L'éditeur VI – gestion des commandes

- :se ai Active le mode auto-indentation
- :se noai Annule le mode auto-indentation

- :se nu Active le mode numérotation des lignes
- :se nonu Annule le mode numérotation des lignes

.:: Les fichiers ::.

Les droits des fichiers

ls -l donne les informations relatives à un fichier :

```
paul@linux:~> ls – l toto
-rw-r--r--  1 paul users 0 2005-08-02 11:11 toto
```

Le fichier appartient à « Paul » et au groupe « Users »

Les droits des fichiers

Unix est un environnement multi-utilisateurs, il doit donc identifier quelles données appartient auquel des utilisateurs du système.
C'est ainsi que chaque fichier à un propriétaire identifié.

Le but d'un système multi-utilisateur étant entre autre de permettre aux utilisateurs de travailler ensemble, le système de groupe à été ajouté. Ainsi tout fichier appartient à un utilisateur mais aussi à un groupe.

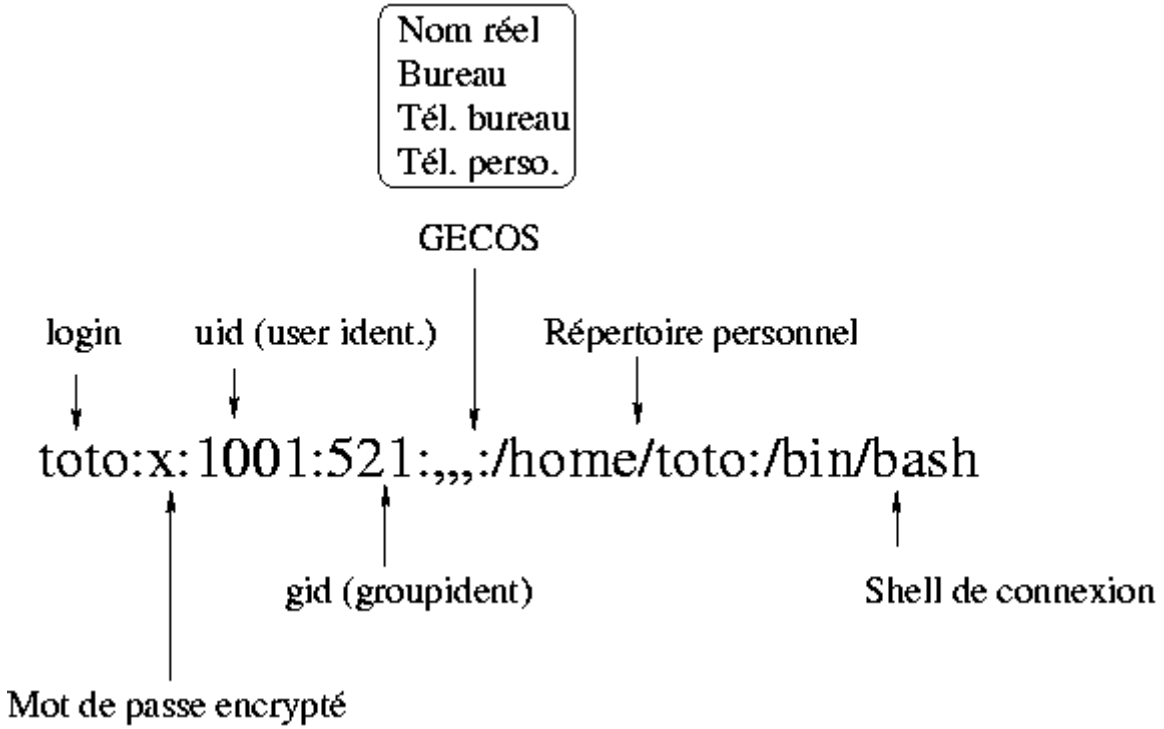
Déclaration des utilisateurs

Le fichier de configuration **/etc/passwd** détient la liste des utilisateurs du système.

```
paul@linux:~> cat /etc/passwd
[...]
sshd:x:71:65:SSH daemon:/var/lib/sshd:/bin/false
ntp:x:74:65534:NTP daemon:/var/lib/ntp:/bin/false
nobody:x:65534:65533:nobody:/var/lib/nobody:/bin/bash
paul:x:1000:100:paul:/home/paul:/bin/bash
```

Déclaration des utilisateurs

Le fichier de configuration **/etc/passwd** détient la liste des utilisateurs du système.



Déclaration des groupes

Le fichier de configuration **/etc/group** détient la liste des groupes du système :

```
paul@linux:~> cat /etc/group
[...]
nobody:x:65533:
nogroup:x:65534:nobody
users:x:100:
```

Rq : le groupe *users* n'indique pas *paul* : dans ce fichier ne sont indiqués que les utilisateurs pour lesquels le groupe est secondaire. *users* est le groupe principal de *paul* il est donc indiqué dans /etc/passwd.

Déclaration des groupes

Lorsque vous créez un fichier, il appartient à votre « user » et à votre « groupe principal ».

Si vous êtes inscrit dans des groupes secondaires vous aurez la possibilité de lire ou écrire des documents qui ont été créés par des personnes appartenant à ces autres groupes.

Sur Unix, on n'utilise pas les groupes uniquement pour les documents, mais aussi pour les programmes ou les périphériques. Il est ainsi facile de vous autoriser à utiliser une carte son par exemple en vous inscrivant ou non dans un groupe secondaire « audio »...

Déclaration utilisateurs

Le mot de passe doit être protégé car même s'il est crypté il peut être percé s'il n'est pas suffisamment complexe.
Le système de shadow password permet cela:

Le mot de passe est enregistré dans un fichier que seul root peut lire.

Le fichier *passwd* ne contient plus le mot de passe, mais les autres informations (nécessaires au fonctionnement du système) sont lisibles par tous.

Les mots de passes peuvent aussi être enregistrés dans une base de données (annuaire LDAP) ... le fonctionnement est alors similaire à celui des cartes banquaires.

Changer le propriétaire d'un fichier

Pour changer l'utilisateur et le groupe propriétaire du fichier, on utilise la commande **chown**

chown ***utilisateur:groupe*** ***fichier***

Seul ***root*** peut changer le propriétaire d'un fichier

L'utilisateur peut changer le groupe s'il appartient au groupe destinataire.

Les droits d'accès aux fichiers

```
paul@linux:~> ls – l toto
-rw-r--r--  1 paul users 0 2005-08-02 11:11 toto
```

Les droits d'accès sont indiqués par trois groupes de caractères : « **rwx** »

- Premier groupe : droits de l'utilisateur propriétaire
- Second groupe : droits du groupe propriétaire
- Dernier groupe : droits des autres

- Le droit est donné si le caractère est présent, il est sinon retiré.

Les droits d'accès aux fichiers

```
paul@linux:~> ls – l toto
-rw-r--r--  1 paul users 0 2005-08-02 11:11 toto
```

Ici :

- L'utilisateur *paul* peut lire et écrire le fichier.
- Tous les utilisateurs appartenant au groupe *users* peuvent lire le fichier.
- Tous les autres utilisateurs (ni *paul* ni ceux du groupe *users*) peuvent lire le fichier.

Modifier les droits d'accès

La commande **chmod** permet de modifier les droits du fichier :

chmod ***option fichier***

Les options permettent d'indiquer les droits à mettre ou les changements à effectuer

Seul le propriétaire (ou root) peuvent modifier les droits d'un fichier.

Modifier les droits d'accès

Modifications de droits par changements :

zone-des-droits{+/-}droits

+ ajoute le droit / - le retire

exemples:

g+w,u+rw,o-r (ajout écrit au groupe, ajout lect/ecrit au user, retrait lect aux autres)

gu+w,u+r,o-r (ajour écrit au groupe et user, ajout lect au user, retrait lect aux autres)

chmod ug+w,u+r,o-r toto

Rq : Il est possible d'utiliser « a » pour identifier toutes les zones-des-droits.

Rq : root garde toujours tous les droits sur tous les fichiers.

Modifier les droits d'accès

Modifications de droits :

zone-des-droits=droits

chmod ug=rw,o=r toto

écriture en octal (R=4, W=2, X=1) => (RW = 4+2 = 6)

1 digit est utilisé pour chaque zone-des-droits.

chmod 664 toto (correspond à l'exemple ci-dessus)

Rq : option -R pour appliquer les modifications récursivement (y compris sur les sous répertoires)

Modifier les droits d'accès

Pour les répertoires l'interprétation des droits est différente de celle des fichiers :

Lecture = droit de lister les fichiers
Écriture = droit de créer / supprimer des fichiers
Exécution = doit de traverser ou accéder au répertoire

.:: Les Jokers ::.

Les caractères génériques

Utilisation de « jokers » pour identifier une liste de fichiers plutôt qu'un seul.

-> copier tous les fichiers .c

-> supprimer tous les fichiers .jpg

Utilisation de caractères spéciaux tels que « * » pour indiquer « n'importe quelle suite de caractères »

-> cp *.c toto/

-> rm *.jpg

Les caractères génériques

Autre Joker : un seul caractère quelconque : « ? ».

Supprimer les fichiers image pic00.jpg à pic99.jpg par exemple : **rm pic??.jpg**.

Les caractères génériques

Possibilité d'indiquer un caractère parmi un ensemble :
Sur le même exemple :

rm pic[0123456789][0123456789].jpg

L'ensemble peut être inversé:

rm pic[!013].jpg n'importe qu'elle caractère sauf 013

Des alias existent:

rm pic[0-9][0-9].jpg

/!\ Les ensembles sont convenablement reconnus lorsque votre variable d'environnement LANG à pour valeur « C »

Les caractères génériques

Comment supprimer le fichier *.c ??

-> rm *.c

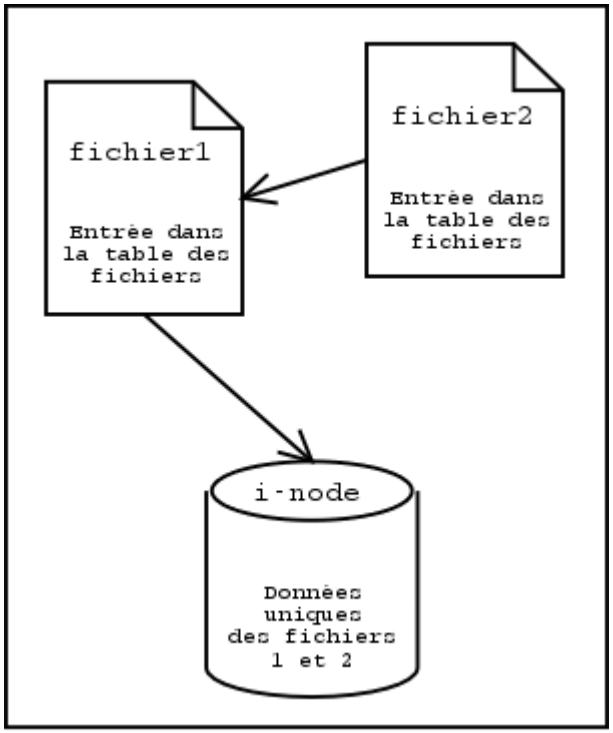
=> « \ » protège « * » de l'interprétation

-> il est aussi possible d'utiliser ' ' ou " "

-> rm '*.c' ou rm "*.c"

.:: Les liens ::.

Les liens symboliques



Pour adapter l'arborescence.
(création d'un raccourcis)

Peut s'appliquer à un fichier ou un dossier.

Création par *ln -s source destination*

S'utilise comme un fichier standard avec vi, rm, cp ...

Visible par *ls -l*

```
lrwxrwxrwx 1 root root   4 avr 24  2002 rbash -> bash
lrwxrwxrwx 1 root root  12 avr 24  2002 /usr/bin/X11 -> ../X11R6/bin
```

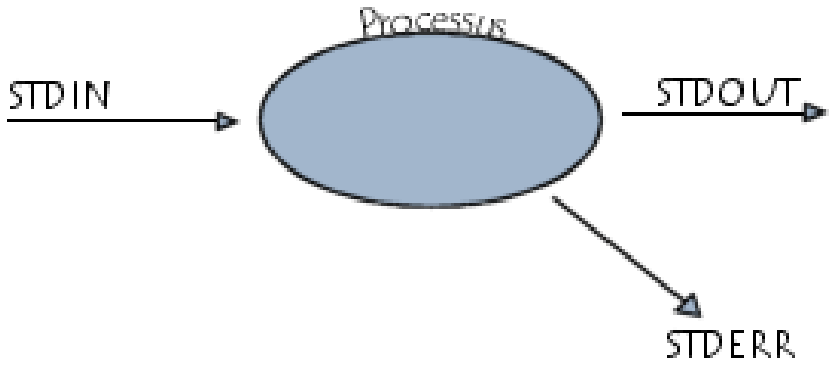
.:: E/S Standards ::.

Les Entrées/Sorties standards

Chaque processus/programme possède :

- Une entrée standard (par défaut le clavier)
- Une sortie standard (par défaut la console/shell)
- Une sortie erreur (par défaut la console/shell)

appelées respectivement : stdin, stdout, stderr



Les Entrées/Sorties standards

Il est possible de rediriger ces flux vers des fichiers pour, par exemple, enregistrer la sortie d'un programme. Ce système peut aussi permettre de passer par un fichier le jeu de test d'un programme.

processus > *fichier*	redirige la sortie standard vers le fichier *fichier*
processus >> *fichier*	ajoute à la fin du fichier *fichier* la sortie standard
processus < *fichier*	passe les données de *fichier* comme entrée du processus.
processus 2> *fichier*	redirige la sortie erreur vers le fichier *fichier*
processus &> *fichier*	redirige les sorties erreur et standard vers le *fichier*
processus >&2	redirige la sortie standard vers la sortie d'erreur.

Les Entrées/Sorties standards

Exemples :

date > date.txt	=> écrit la date dans le fichier date.txt
sort < fic.txt > fic.trié.txt	=> trie le fichier fic.txt et enregistre le résultat dans fic.trié.txt
ls *.c 2>/dev/null	=> redirige les erreurs vers le périphérique null (les supprime)
echo “Erreur !” >&2	=> envoie erreur sur la sortie erreur

Les Entrées/Sorties standards

Chaînage des entrées / sorties : les TUBES ou PIPES

Pour connecter la sortie d'un processus sur l'entrée d'un autre.
Permet d'enchaîner des commandes sans passer par des fichiers intermédiaires.

La syntaxe est *commande1* | *commande2*.
Le tube est donc identifié par " | "

Rq : les commandes sont lancées en parallèle et non l'une après l'autre.
Rq : si le lecteur se termine avant l'écrivain => « Broken Pipe ».

Les Entrées/Sorties standards

Exemple d'utilisation des TUBES :

- ls | wc -l compte le nombre de ligne retournées par ls
- cat /etc/passwd | grep “/bin/bash” | cut -d “:” -f 1 | sort > resultat.txt

  extrait du fichier passwd les lignes des utilisateurs utilisant le shell bash ; extrait de ces lignes le login (champ 1) ; trie ces login et enregistre le résultat dans *resultat.txt*.

.:: Les variables shell ::.

Les variables du shell

Les variables SHELL doivent commencer par une lettre et être composées de caractères alphanumériques. Le type de ces variables est par défaut « chaîne de caractères ».

Une variable est initialisée comme suit :

paul@linux> maVariable="maValeur"

Une variable est, par défaut, locale : elle n'est pas accessible aux applications lancées par ce shell.

Pour être visible des processus fils, la variable doit être **exportée :**

paul@linux> export maVariable= "maValeur"

Les variables du shell

La suppression d'une variable se fait pas la commande :
paul@linux> unset maVariable

La liste des variables peut être consulté par la commande **env**

Pour accéder au contenu d'une variable il faut faire précéder son nom du symbole $
paul@linux> echo $PS1

Les variables du shell

Il est possible d'utiliser des variables contenant des nombres :

```
paul@linux> a=1
paul@linux> b=$a+1
            /!\ Dans ce cas b vaut : "1+1" et non 2
paul@linux> b=$(( $a + 1 ))
            /!\ Dans ce cas b vaut 2
```

=> Pour effectuer des calculs : $((calculs))

.:: Commandes utiles ::.

Quelques commandes utiles

- **cut** *option fichier* extrait de chaque ligne une zone ou un champ

  exemple d'option :

  -d “:” -f 2 => sélectionne le second champ ; les champs étant délimités par le symbole “:”

  -b 5-8 => sélectionne les caractères 5 à 8

- **grep** *option fichier* afficher les lignes contenant un motif donné

  grep “/bin/bash” /etc/passwd

Quelques commandes utiles

- **printf** *format arguments* — Mettre en forme des données et les afficher
- **sed** *options fichier* — Éditeur non interactif (recherche, remplacement de motifs...)
- **sort** *options fichier* — Trie les lignes d'un fichier texte

Quelques commandes utiles

find parcours l'arborescence du disque à la recherche de fichiers répondant aux critères demandés et exécute une action pour chacun.

Exemple de critères :

-name "*.c" -empty

Le système cherche d'abord les fichiers correspondant à *.c et ne retient que ceux qui sont vides. (de gauche à droite ; And else).

Quelques commandes utiles

Exemple d'action :

-print	affiche le nom des fichiers retenus
-exec cat {} “;”	affiche le contenu des fichiers en exécutant la commande cat pour chacun {} est remplacé par le nom du fichier.

Exemple complet :

```
find / -name "*.c" -exec cat {} ";"
```

Quelques commandes utiles

Critères :

- anewer *file* : plus récent que le fichier
- atime *n* : dernier accès il y a n*24h
- cmin *n* : dernière modif il y a n minutes
- cnewer *file* : dernière modif plus récente que *file*
- empty : fichier vide
- name *motif* : le nom du fichier correspond au motif
- iname *motif* : idem mais case insensitive
- path *motif* : recherche dans le chemin du fichier le motif
- prune : ne pas descendre le répertoire en cours

Quelques commandes utiles

Critères :

- o : prédicat OU => permet de réaliser des actions conditionnées :

find . -path "/dev" -prune -o -print

si le chemin contient /dev ne pas descendre l'arborescence sinon afficher le nom du fichier

=> affiche tous les fichiers hors mis ceux du répertoire /dev

1. Quelle commande permet de se déplacer dans les répertoires ?

a - rmdir **b – cd** **c – cat**

2. Comment aller directement dans le répertoire home de l'utilisateur courant ?

a – cd home **b – cd /** **c – cd**

3. Comment quitter une page de manuel pour retourner dans le shell ?

a – taper exit **b - taper q** **c – fermer le terminal**

4. Quelle commande permet de connaître le répertoire courant ?

a – ls **b - cd** **c - pwd**

5. Dans quel répertoire sont rattachés les périphériques du système (carte son ....) ?

a – /etc **b – /proc** **c - /dev**

6. Linux est issu ?

a – d'un UNIX d'AT&T **b - de XENIX** **c – de rien**

7. Comment supprimer un répertoire non vide :

a - rm **b – delete** **c - rmdir**

8. Quelle commande permet de copier un fichier ?

a – mv **b – copy** **c - cp**

9. Lequel de ces programme est un shell ?

a – bash **b - kbsh** **c - myaishache**

10. Ou ce situe votre home ?

a – /home/ **b- /home/etud/info/** **c- dans la cité U en bas de la pente.**