Système d'Exploitation

(Les commandes de base de LINUX)

2019-2020

Plan

- Présentation générale
- Commandes de gestion des fichiers et des répertoires
- Les droits d'accès
- Archivage et restauration des fichiers
- Flux de redirection
- Gestion des processus

Chapitre d'introduction

Systèmes d'exploitations (Operating System OS)

Pour que les composants d'un ordinateur puissent être exploités il est nécessaire d'installer un OS sur cet ordinateur.

Un OS est un logiciel qui fait office d'interface entre l'utilisateur et le langage machine de l'ordinateur.

Définition : Un Système d'Exploitation (S.E.) est une machine abstraite conçue pour faciliter l'exploitation du matériel (pilotes de périphériques) ou pour arbitrer l'accès au matériel par les utilisateurs.

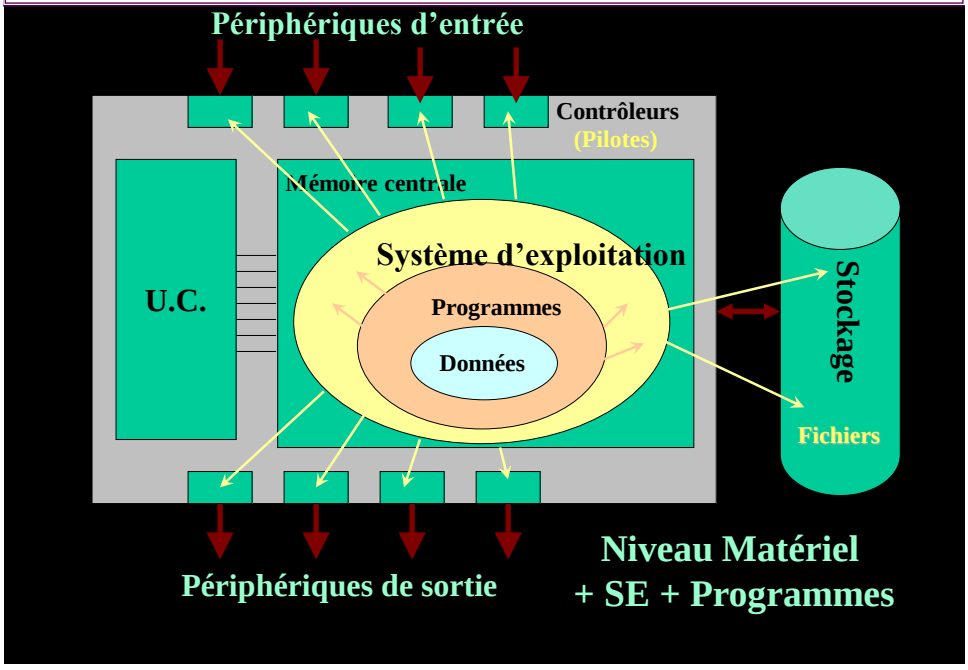
Généralement composé d'un **noyau** et d'un **ensemble d'outils** système, le S.E. permet de développer des applications portables, qui ne sont pas spécifiques à un ordinateur ou un système donné.

Systèmes d'exploitations (Operating System OS)

Un OS définit et gère :

- les périphériques, lecteurs, les structures de commandes, et les programmes,
- la répartition des ressources (gestion des ressources),
- la gestion des processus, les composants internes et les périphériques externes,
- un système de fichiers et l'interface utilisateur.
- un système de sécurisation, des routines de gestion des tâches

Utilisation d'Unix : Chapitre d'introduction



Systèmes d'exploitations (Operating System OS)

Il existe aujourd'hui une centaines de OS. La classification fréquente des systèmes d'exploitation est :

- OS mono-tâche : exécution d'un seul programme à la fois,
- OS multi-tâches : exécution de plusieurs processus simultanément,
- OS mono-session : un seul utilisateur sur la machine à la fois,
- OS multi-sessions : Plusieurs utilisateurs simultanément sur la même machine.

Utilisation d'Unix : Chapitre d'introduction

Systèmes d'exploitations (Operating System OS)

Exemples de OS:

MS-DOS : OS mono-tâche, mono-session,

Windons : OS multi-tâches, mono-session, multi-utilisateurs,

UNIX : OS multi-tâches, multi-sessions,

LINUX : Version d'UNIX pour PC,

Chapitre II

Présentation Générale d'UNIX

UNIX : Définition

UNIX est un OS multi-tâches multi-utilisateurs et multi-sessions.

- Contrôle le hardware (le matériel)
- Partage la mémoire entre les processus
- Contrôle la création et l'accès aux fichiers

UNIX : Définition

Les outils intégrés

- Système de fichiers hiérarchique et réparti
- Le Shell (interpréteur de commandes)
- Manipulations très puissantes des fichiers
- Communication entre processus (tubes)
- Nombreux utilitaires

Utilisation d'Unix : Ch. II. Présentation générale d'Unix

UNIX : Historique

- Naissance 1970
- Lieu : Laboratoires Bell de AT&T
- 1ère version commerciale: 1975 (V6 en C)

UNIX International

- Consortium fondé en 1988 par AT&T, Sun et Unisys
- But : Nouvelle version d'Unix (BSD 4.3 et System V R3)

UNIX et PC

- Linux , SunSolaris pour PC, ...
- Interfaces gratuites de la FSF (Free Software Fondation)

Utilisation d'Unix : Ch. II. Présentation générale d'Unix

UNIX : Connexion et Déconnexion

- LogIn

  L'accès au système se fait par la donnée :

  d'un nom d'utilisateur
  - unique, au plus 8 caractères
  - Créé par l'administrateur système (ouverture de Compte)

  d'un mot de passe :
  - ✓ strictement personnel
  - ✓ garantissant la confidentialité de vos informations
  - ✓ empêche les intrusions en votre nom

- LogOut

  Fin de l'utilisation du Shell : Ctrl-d ou (logout ou exit selon le Shell)

UNIX : Connexion et Déconnexion

nom1@nom2 :~$

nom1 : le premier élément est votre pseudonyme.

nom2 : ça c'est le nom de l'ordinateur sur lequel vous êtes en train de travailler.

La ligne d'invite de commandes se lit donc 'nom1' chez 'nom2'. En d'autres termes, je suis identifié en tant que nom1 sur la machine nom2.

: : c'est un séparateur.

~ :c'est le dossier dans lequel vous vous trouvez actuellement., le symbole ~ signifie que vous êtes dans votre dossier personnel, ce qu'on appelle le "Home" sous Linux. C'est l'équivalent du dossier "Mes documents" de Windows.

UNIX : Connexion et Déconnexion

nom1@nom2 :~$

$: ce dernier symbole est très important, il indique votre niveau d'autorisation sur la machine. Il peut prendre 2 formes différentes :

$: signifie que vous êtes en train d'utiliser un compte utilisateur "normal",

: signifie que vous êtes en mode super-utilisateur,

Activer le compte root (administrateur)

nom1@nom2 :~$ sudo passwd root

UNIX : Le Shell

- Interface entre l'utilisateur et Unix.
- Interprète les commandes, génère les noms de fichiers, crée des tubes, redirige les entrées/sorties, etc.
- Langage de programmation structuré.

Quand un utilisateur se connecte au système Unix, un Shell est démarré pour lui.

Les commandes

Une commandes est constituée d'une suite de mots, chaque mot est séparé du suivant par un blanc (espaces ou tabulations). Le premier de ces mots (c'est parfois le seul) est le nom de la commande.

Exemple :

La commande

date

affichera

Wed Oct 9 12:12:19 GMT+0:00 2003

Chapitre III

Système de fichiers sous UNIX

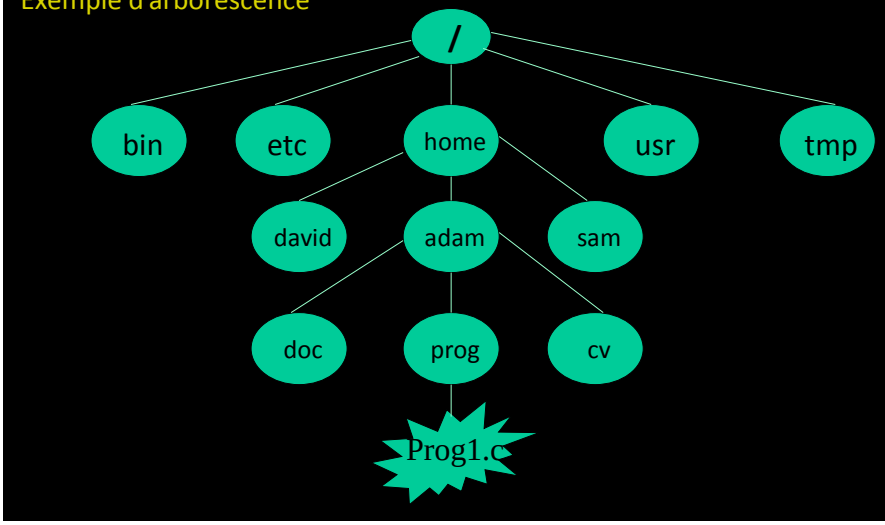
Système de fichiers

Comme dans tous OS, les données sous Unix sont organisées en fichiers eux-mêmes organisés en répertoires (et sous répertoires) sur les éléments de sauvegarde.

L'architecture du système de fichiers est sous la forme d'un arbre dont le répertoire noté / est la racine. Tout les autres répertoires en sont les nœuds et les fichiers les feuilles. Un système de fichiers sous Unix est un arbre n-aires.

Système de fichiers

Exemple d'arborescence



Système de fichiers

Sous Unix (et contrairement au DOS et WINDOWS), tous les caractères alphanumériques plus la ponctuation sont des caractères valides.

De plus il n'existe pas de notion extension de fichier. Cela fait du caractère '.' un caractère comme un autre qui peut apparaître plus d'une fois dans un nom de fichier.

Le système Unix n'accepte pas les caractères accentués et distingue les minuscules des majuscules.

Système de fichiers

Exemples

- MonProg.c monprog.c sont différents.
- a.b… est un nom de fichier valide sous Unix.
- tp6-31/03/2003.c est valide
- _z_ l'est aussi

Sous DOS les fichier sont au format 8.3, le système Unix autorise, tout comme windows, des noms de fichiers long (256 caractère environ).

les caractères invalides sous DOS (;+=[]', */<>\?|.) sont acceptés par Unix

Chemin d'un fichier

Un fichier est localisable sans ambiguïté par son chemin. Ce dernier représente la succession des répertoires à parcourir pour accéder au fichier (navigation dans l'arbre).

Les répertoires sont séparés par un slash, noté /, dans l'écriture du chemin.

Chemin d'un fichier

Répertoire parent

Un répertoire parent est celui immédiatement supérieur à celui courant. Il est noté par deux point '..'.

Répertoire courant

On appelle répertoire courant celui dans lequel on se trouve à un instant donné durant la navigation dans le système de fichier. Il est noté point '.'.

Utilisation d'Unix : Ch. III. Système de fichiers sous Unix

Chemin d'un fichier

Chemin absolu

Le chemin absolu désigne la succession des répertoires à parcourir depuis la racine pour accéder au fichier spécifié.

Exemple: /home/etude/tpc/tp3.

Remarque :
Commence par « / »

Chemin relatif

Un chemin relatif désigne la succession des répertoires à parcourir depuis le répertoire courant pour accéder au fichier spécifié.

Exemple:

../monprog.c pour accéder au fichier monporog.c lorsqu'on se trouve dans le répertoire tpc. La présence du répertoire parent dans ce chemin relatif permet de remonter dans l'arbre

Type de fichier

Sous Unix, les fichiers peuvent être de 5 types différents:

Fichiers exécutables

Correspondent à des programmes (écrits en langage C généralement).

Répertoires

Un répertoire est un ensemble de fichier et de sous-répertoire.

Un répertoire peut ne pas contenir aucun fichier. Mais, un répertoire contient toujours un répertoire parent '..' et un répertoire courant '.'.

Utilisation d'Unix : Ch. III. Système de fichiers sous Unix

Type de fichier

Les liens

Les liens sont des fichiers assez spéciaux puisqu'ils permettent d'associer plusieurs noms à un un seul et même fichier. Pour créer un lien, utiliser la commande ln. Il existe deux types de liens:

Lien symbique : un lien symbolique est un simple pointeur vers un autre fichier bien réel.

la commande ln –s permet de le créer.

Syntaxe: ln –s nom-du-fichier-à-pointer nom-du-lieu-symbolique

Type de fichier

Lien physique : un lien physique sur un fichier est un fichier qui occupe exactement le même espace disque que le fichier vers lequel il pointe.

Syntaxe: ln nom-du-fichier-à-pointer nom-du-lieu-symbolique

Fichier

Tout fichier qui est ni exécutable, ni un répertoire et ni un lien.

Fichier caché

Les fichiers et répertoires cachés se distinguent des autres par la seule présence d'un point (.) en première position dans leur nom. La commande de listage des fichiers ne les affichera pas par défaut.

La Navigation (cd)

La commande Unix de navigation dans le système de fichier est : cd pour Change Directory.

Exemple	Description
cd.	Se déplacer vers le répertoire courant, ie, ne pas changer de répertoire.
cd ..	Aller dans le répertoire parent.
cd /	Sauter vers le répertoire racine.
cd / home	Se déplacer vers le répertoire home fils de la racine.
cd /home/tc	Parcourir l'arbre jusqu'au répertoire TC en passant par la racine et home.
cd ../Mail	Remonter l'arbre d'un cran, puis aller dans le répertoire Mail.
cd ../..	Remonter de deux crans.

Chapitre IV

Les Commandes Elémentaires

Commandes sur le système de fichier

Commande	Description
rm	Supprime un fichier ou un répertoire
mv	Déplace ou renomme un fichier ou un répertoire
mkdir	Crée un répertoire
rmdir	Supprime un répertoire(vide)
cp	Copie un fichier ou un répertoire
pwd	Affiche le chemin du répertoire courant
cd	Se déplace dans le système de fichier
ls	Affiche la liste des fichiers et répertoires
ln	Crée un lien vers un fichier

Aux commandes (rm,mv,cp) on peut spécifier plusieurs arguments et utiliser les méta-caractères

Lister les fichiers (ls)

une commande essentielle pour tout OS est le listage des fichiers et sous-répertoires du répertoire courant.

Commande	description
ls	Listage par ordre alphabétique en colonnes
ls -l	Listage en ligne avec toute les informations
ls -a	Listage en plus des fichiers cachés
ls -m	Sépare les fichiers par une virgule
ls -t	Tri par date
ls -lu	Tri par date du dernier accès et l'affiche
ls -F	Affiche les types des fichiers i.e en rajoutant le symbole correspondant: / répertoire, * (exécutable), @ (lien)
ls -S	Tri par ordre de taille décroissante
ls -X	Tri par type extension
ls -r	Tri inverse

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Méta-caractères

les méta-caractères * (astérisque) et (?) sont très utiles lors de la manipulation de fichiers en groupe. (*) remplace dans l'expression dans laquelle apparaît, zéro, un ou plusieurs autre(s) caractère(s). Quand à ? il remplace un seul caractère.

Exemple :

- La commande rm * -r supprime tous les fichiers et sous-répertoires du répertoire courant.
- Alors que mv tp*.c tpc déplace dans le répertoire tpc tous les fichiers ayant n'importe quoi entre tp et .c.
- Mais mv tp??.c tpc, déplace les fichiers contenant exactement deux caractères entre tp et .c.

Caractère spécial	Signification	Expression	Chaînes contenant l'expression
.	Un Caractère quelconque	a.	ab,ac,ad,a2,a!,…
^	Début de ligne	^abc	abcAbc,abcabc,…
$	Fin de ligne	abc$	ABCabc,abCabc,…
[ ]	Un des caractères entre les crochets	[Aa] [a-z]	Abc,abc,Abc,aBc,.. abbbb,bcccc,zzzzzz,…
?	L'élément précédent est optionnel (peut être présent 0 ou 1 fois)	abc?	ab,abc,…
*	L'élément précédent peut être présent 0, 1 ou plusieurs fois	abc*	abc,abc,abcc,abccc,…
+	L'élément précédent doit être présent 1 ou plusieurs fois	abc+	abc,abcc,abccc,…
|	Ou	Abc|abc	Abc,abc
()	Groupement d'expressions	(abc)	abc,abcc,abcb,abcd,…

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers

Affichage

Il existe plusieurs commandes spécialisées qui permettent d'afficher sur la sortie standard le contenu d'un fichier.

+ Afficher le contenu d'un fichier (cat)

La commande cat permet d'afficher sur la sortie standard le contenu d'un fichier.

Syntaxe: cat fichier

Exemple: cat lettre.tex

Manipulation de fichiers

Affichage

+ Affichage inverse (tac)

La commande tac est homologue à cat mais affiche le contenu d'un fichier en partant de la dernière ligne vers la première.

+ Afficher l'entête (head)

La commande head affiche que les première ligne d'un fichier (10 par défaut). En voici les options.

- cN : affiche les N premiers octets
- nN : affiche les N premières lignes
- q : n'affiche pas le nom du fichier
- v : affiche le nom du fichier avant d'en afficher l'entête.

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers

Affichage

+ Afficher la fin (tail)

La commande tail permet d'afficher les dernières lignes d'un fichier. Elle est homologue à head et possède les mêmes attributs.

+ Afficher écran par écran (more et less)

Les commandes more et less permettent d'afficher page par page des fichiers volumineux sur la sortie standard.

Manipulation de fichiers

Affichage

+ Affichage avec tri (sort)

La commande sort permet de trier les lignes d'un fichier. En voici les options :

- b : ignore les espace en début de ligne
- d : ordre alphabétique (A-Z, a-z, 0-9, espace) (apr. défaut)
- f : ignore la casse
- n : ordre numérique
- r : inverse l'ordre de tri

Manipulation de fichiers

Statistique (wc)

La commande wc permet de compter le nombre de caractères, de mots, et de lignes d'un fichier. Ses options sont :

- l (Lignes): Compte le nombre de lignes
- w (Words): Compte le nombre de mots
- c (Char): Compte le nombre de caractères
- L (Length max ligne) : affiche la longueur de la ligne la plus longue.

Syntaxe: wc [option] fichier

Manipulation de fichiers

Type de fichier (file)

La commande file permet de connaître le type d'un fichier.

Syntaxe : file fichier

Cette commande retourne le type d'un fichier passé en paramètre. Pour opérer, elle fait appel à un fichier qui contient les signatures binaires d'un grand nombre de fichier.

Manipulation de fichiers

Création d'un fichier (touch)

La commande touch permet de créer un nouveau fichier vide.

Syntaxe : touch fichier

Appliquer à un fichier déjà existant, elle modifie son heure de dernier accès et met cette dernière à l'heure courante par défaut.

Manipulation de fichiers

supprimer les doublons (uniq)

Parfois, certains fichiers contiennent des lignes en double et on aimerait pouvoir les détecter ou les supprimer. La commande uniq est toute indiquée pour cela.

Nous devons travailler sur un fichier trié. En effet, la commande uniq ne repère que les lignes successives qui sont identiques.

Syntaxe:

ubuntu@ubuntu:~$ uniq fichier

Manipulation de fichiers

supprimer les doublons (uniq)

Exemple soit le fichier doublons.txt

doublons.txt

Amot1
Amot1
Amot1
Bmot2
Bmot2
Cmot3
Dmot4

ubuntu@ubuntu:~$ uniq doublons.txt

Amot1
Bmot2
Cmot3
Dmot4

- Ecrire dans un autre fichier plutôt qu'affiché dans la console :

La liste sans doublons sera écrite dans sans_doublons.txt.

ubuntu@ubuntu:~$ uniq doublons.txt sans_doublons.txt

Manipulation de fichiers

supprimer les doublons (uniq)

-c : compter le nombre d'occurrences

```
ubuntu@ubuntu:~$ uniq   -c    doublons.txt

  3    Amot1
  2    Bmot2
  1    Cmot3
  1    Dmot4
```

-d : afficher uniquement les lignes présentes en double

```
ubuntu@ubuntu:~$ uniq -d doublons.txt
Amot1
Bmot2
```


Manipulation de fichiers

couper une partie du fichier (cut)

Couper selon le nombre de caractères :

soit le fichier suivant :

conserver uniquement les caractères 2 à 5 de chaque ligne du fichier

```
ubuntu@ubuntu:~$ cut  -c    2-5   fichier.txt
lber
ranç
Ean
```

fichier.txt
Albert
François
Jean

Pour conserver du 1er au 3ème caractère :

```
ubuntu@ubuntu:~$ cut   -c   -3    fichier.txt
Alb
Fra
Jea
```

Manipulation de fichiers

couper une partie du fichier (cut)

De même, pour conserver du 3ème au dernier caractère :

```
ubuntu@ubuntu:~$ cut -c 3- fichier.txt
bert
ançois
an
```

Couper selon un délimiteur

Syntaxe : cut -d "délimiteur" -f "num_champs" fichier

Vous allez avoir besoin d'utiliser deux paramètres :

-d : indique quel est le délimiteur dans le fichier.

-f : indique le numéro du ou des champs à couper.

Manipulation de fichiers

couper une partie du fichier (cut)

Exemple

```
ubuntu@ubuntu:~$ cut -d , -f 1 notes.txt
Med
ahmed
Omar
Imane
```

notes.txt

```
Prénom note commentaire
med, 18 / 20 , Excellent
ahmed, 3 / 20 , Nul
omar, 14 / 20 , Bien
imane, 9 / 20 , Moyenne
```

```
ubuntu@ubuntu:~$ cut -d , -f 1,3 notes.txt
med, Excellent
ahmed, Nul
omar, Bien
imane, Moyenne
```

Manipulation de fichiers

couper une partie du fichier (cut)

Exemple

De même, il est possible de conserver toute une série de champs avec le tiret comme tout à l'heure :

ubuntu@ubuntu:~$ cut -d , -f 2-4 notes.txt

a pour effet de conserver les champs n°2, 3 et 4. D'autre part,

ubuntu@ubuntu:~$ cut -d , -f 3- notes.txt

conserve les champs du n°3 jusqu'à la fin

on peut utiliser aussi la commande grep avec cut

ubuntu@ubuntu:~$ grep Med notes.txt | uniq –f 1 | cut -d , -f 3

excellent // ici uniq retourne le premier élément rencontré

ubuntu@ubuntu:~$ grep Med notes.txt | cut -d , -f 2,3

18 / 20 excellent

Manipulation de fichiers

Découpé un fichier (split)

Cette commande permet de découper un fichier en plusieurs morceaux.

Syntaxe: split -n fichier fichiers_resultat

Ses options sont :

- b n(Bytes) : découpage par bloc de n octets
- l n(Lignes) : découpage par blocs de n lignes

Exemple:

ubuntu@ubuntu:~$ split -10 exemple petitfichiers

Cette exemple va créer les fichiers petitfichiersaa, petitfichiersab, …

contenant le contenu du fichier exemple par tranche de 10 lignes.

Manipulation de fichiers

Découpé un fichier (split)

Exemple:

ubuntu@ubuntu:~$ split –b 135000 vacances.mpeg vacances.mpeg
découpe vacances.mpeg en fichiers de 1.35 Mo

Les fichiers issus de la découpe auront un nom ayant pour suffixe des lettres du type aa, ab, ac… crées dans l'ordre lexicographique naturel (descendant de 'a' vers 'z').

Note : pour découper des fichiers texte, faire la découpe en nombre de lignes. Tandis que pour découper les fichiers binaires, utiliser la découpe en nombre d'octets.

Manipulation de fichiers

Linges communes (comm)

comm La commande qui permet d'extraire les lignes communes et les lignes uniques de 2 fichiers préalablement triés.

ubuntu@ubuntu:~$ comm [123] fichier1 fichier2

1: supprimer la première colonne (lignes uniques au premier fichier).

2: supprimer la deuxième colonne (lignes uniques au deuxième fichier).

3: supprimer la troisième colonne (lignes communes aux deux fichiers).

Manipulation de fichiers

Linges communes (comm)

Exemple

Fichier1.txt

```
Med
Marcel Rajput
Bonjour
Albrt
```

Fichier2.txt

```
Med
Albrt
Omar
Meryem benomar
```

ubuntu@ubuntu:~$ comm fichier1 fichier2

```
            Med
Marcel Rajput
Bonjour
            Albert
      Omar
      Meryem benomar
```


Manipulation de fichiers

Linges communes (comm)

```
            Med
Marcel Rajput
Bonjour
            Albert
      Omar
      Meryem benomar
```

fichier1.txt

```
Med
Marcel Rajput
Bonjour
Albrt
```

fichier2.txt

```
Med
Albrt
Omar
Meryem benomar
```

La sortie ci-dessus contient trois colonnes où la première colonne est séparée par zéro tabulation et contient les noms uniquement présents dans le fichier1.txt ,la deuxième colonne contient les noms uniquement présents dans le fichier2.txt et séparés par une tabulation et la troisième colonne contient les noms communs aux deux fichiers et est séparée par deux tabulations à partir du début de la ligne. C'est le modèle par défaut de la sortie produite par la commande comm lorsqu'aucune option n'est utilisée .

Manipulation de fichiers

Linges communes (comm)

Exemple

```
ubuntu@ubuntu:~$ comm   -1  fichier1  fichier2
```

```
       Med
       Albert
Omar
Meryem benomar
```

supprimer la première colonne en utilisant -1

```
ubuntu@ubuntu:~$ comm   -2  fichier1  fichier2
```

```
        Med
Marcel  Rajput
Bonjour
        Albert
```

supprimer la deuxième colonne en utilisant -2

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers

Linges communes (comm)

Exemple

```
ubuntu@ubuntu:~$ comm   -3  fichier1  fichier2
```

```
Marcel Rajput
Bonjour
       Omar
       Meryem benomar
```

supprimer la première colonne en utilisant -1

```
ubuntu@ubuntu:~$ comm   -12  fichier1  fichier2
```

```
Med
Albert
```

la suppression des colonnes multiples

Manipulation de fichiers

Différence(diff)

La commande diff permet de comparer le contenu de deux fichiers ligne à ligne .

Syntaxe : diff [option] fichier1 fichier2

Les options :

- b : ignore les différences dus à des espaces blancs
- B : ignore les différences dus à des lignes blanches
- i : ignore les différences minuscules/Majuscules
- r : comparaison récursive des fichiers d'un répertoire, sous répertoire…

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers

Différence(diff)

Il est important de se rappeler que diff utilise certains symboles et instructions spéciales qui sont nécessaires pour rendre deux fichiers identiques. Il vous indique les instructions à suivre pour modifier le premier fichier afin de le faire correspondre au second.

Syntaxe de résultat : numéro1 [a,c ou d] numero2

a : ajouter
c : changer
d : supprimer

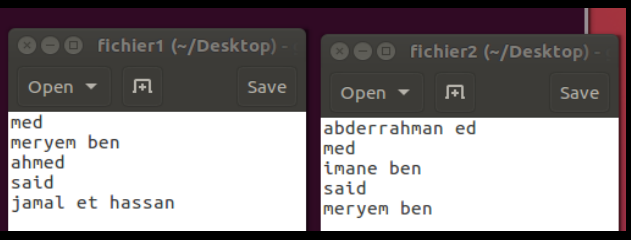
Numéro1 correspondant à la ligne du premier fichier,
Numéro2 correspondant à la ligne du deuxième fichier.

Manipulation de fichiers

Différence(diff)

ubuntu@ubuntu:~$ diff fichier1 fichier2

```
0a1
> abderrahman ed
2,3c3
< meryem ben
< ahmed
---
> imane ben
5c5
< jamal et hassan
---
> meryem ben
```



0a1 signifie qu'après les lignes 0 (début du fichier), vous devez ajouter abderrahma ed pour correspondre à la deuxième fichier, ligne numéro 1
2, 3c3 signifie de la ligne 2 à la ligne 3 du premier fichier doit être modifiée pour correspondre à la ligne 3 du deuxième fichier.
> Signifie les lignes du premier fichier
< Signifie les lignes du deuxième fichier

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers

Recherche de fichier (find)

La commande find est ultra puissante, elle permet de faire une recherche sur le système de fichier et d'afficher la liste des fichiers satisfaisant à une combinaison de critères très variés.

Syntaxe : find répertoire critères [-option]

Exemple : find . –name ''*.c '' -print

Manipulation de fichiers

Les critères de recherche sont les suivants :

-name recherche sur le nom du fichier, on peut spécifier le nom en utilisant les expressions régulières

Exemple

find /home –name [A-Z]* → recherche sur les noms qui commencent par majuscule

-perm recherche sur les droits d'accès du fichier,

-links recherche sur le nombre de liens du fichier,

-user recherche sur le propriétaire du fichier,

-group recherche sur le groupe auquel appartient le fichier,

-type recherche sur le type (d=rép., c=car., f=fichier normal),

-size recherche sur la taille du fichier en nombre de blocs (1

- bloc=512octets),

-atime recherche par date de dernier accès en lecture du fichier,

-mtime recherche par date de dernière modification du fichier,

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers

-ctime recherche par date de création du fichier.

On peut combiner les critères avec des opérateurs logiques :

critère1 critère2 ou critère1 -a critère2 correspond au et logique,

!critère non logique,

\ (critère1 -o critère2\) ou logique,

La commande find est récursive, c'est à dire où que vous tapiez, il va aller scruter dans les répertoires, et les sous répertoires qu'il contient, et ainsi de suite.

Manipulation de fichiers(La commande find)

Recherche par nom de fichier

Pour rechercher un fichier dont le nom contient la chaîne de caractères toto à partir du répertoire /usr, vous devez tapez :

```
find  /usr  -name toto
```

En cas de réussite, si le(s) fichier(s) existe(nt), vous aurez comme sortie :

toto

En cas d'échec, vous n'avez rien.

Pour rechercher tous les fichiers se terminant par .c dans le répertoire /usr, vous taperez :

```
find  /usr  -name " *.c "
```

Vous obtenez toute la liste des fichiers se terminant par .c sous les répertoires contenus dans /usr (et dans /usr lui même).

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers(La commande find)

Recherche suivant la date de dernière modification

Pour connaître les derniers fichiers modifiés dans les 3 derniers jours dans toute l'arborescence (/), vous devez taper :

```
find  / -mtime 3
```

Recherche suivant la taille

Pour connaître dans toute l'arborescence, les fichiers dont la taille dépasse 1Mo (2000 blocs de 512 octets), vous devez taper :

```
find  /  -size 2000
```

ou

```
find  /  -size 1M
```

Pour spécifier une taille tapez (M (majuscule) pour Méga octet, G (majuscule) pour Giga octet, k (minuscule) pour kilo octets, pour les octets on a un bloc contient 512 octets).

Manipulation de fichiers(La commande find)

Recherche suivant la date de dernière modification

Recherche combinée

Vous pouvez chercher dans toute l'arborescence, les fichiers ordinaires appartenant à olivier, dont la permission est fixée à 755, on obtient :

```
find / -type f -user olivier -perm 755
```

Recherche en utilisant les opérateurs logiques

Si vous voulez connaître les fichiers n'appartenant pas à l'utilisateur olivier, vous taperez :

```
find  .  ! -user  olivier
```

! -user olivier, est la négation de -user olivier, c'est à dire c'est tous les utilisateurs sauf olivier.

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers(La commande find)

Recherche suivant la date de dernière modification

Recherche des fichiers qui ont pour nom a.out et des fichiers se terminant par .c. On tape :

```
find  .  \  ( -name a.out -o -name " *.c " \  )
```

On recherche donc les fichiers dont le nom est a.out ou les fichiers se terminant par *.c, une condition ou l'autre.

Recherche des fichiers qui obéissent à la fois à la condition a pour nom core et à la condition a une taille supérieure à 1Mo.

```
find . \  (-name core -a size +1M \  )
```

Manipulation de fichiers(La commande find)

Les commandes en option

on dispose de l'option -exec. find couplé avec exec permet d'exécuter une commande sur les fichiers trouvés d'après les critères de recherche fixés. Cette option attend comme argument une commande, celle-ci doit être suivi de {}\ ;

Exemple recherche des fichiers ayant pour nom core, suivi de l'effacement de ces fichiers.

```
find . -name core -exec rm {}\   ;
```

Tous les fichiers ayant pour nom core seront détruits, pour avoir une demande de confirmation avant l'exécution de rm, vous pouvez taper :

```
find . -name core -ok rm {}\   ;
```

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers (La commande grep)

La commande grep permet de rechercher une chaîne de caractères dans un fichier. Les options sont les suivantes :

- -v affiche les lignes ne contenant pas la chaîne
- -c compte le nombre de lignes contenant la chaîne
- -n chaque ligne contenant la chaîne est numérotée
- -x ligne correspondant exactement à la chaîne
- -w lignes où le mot apparaît tel quel
- -l affiche le nom des fichiers qui contiennent la chaîne
- -i ne pas tenir compte de la casse (majuscules / minuscules)
- -r rechercher dans tous les fichiers et sous-dossier
- -E pour les expressions régulières plus compliquées,

Syntaxe

```
grep    critères_de_recherche     nom_fichier_ou_chemin_fichier
```

Manipulation de fichiers (La commande grep)

Le modèle de critères(noms) peut contenir les caractères spéciaux suivants :

[…] Plage de caractères permis à cet emplacement
[^…] Plage de caractères non permet à cet emplacement
. Un caractère quelconque (y compris un espace)
* Signe de répétition. Agit sur le caractère placé devant le *
$ Matérialise une fin de ligne
| ou
^ début de ligne
\{…\} Caractère de répétition, entre les accolades
\{nombre\} nombre exacte
\{nombre,\} nombre minimum
\{nombre1,nombre2\} de nombre1 à nombre2a

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers(La commande grep)

- Exemples avec le fichier carnet-adresse :

```
Olivier:29:0298333242:Brest devenez
alivier:29:0298333242:Brest devenez
blivier:29:0298333242:Brest devenez
marcel:13:0466342233:Gardagnes devenez
Myriam:30:0434214452:Nimes  nez
yvonne:92:013344433:Palaiseau nez
```

On peut utiliser les expressions régulières avec grep pour spécifier les critères ainsi que le nom de fichier ou répertoire si on tape la commande :

grep ^[A-Z] carnet-adresse

On va obtenir toutes les lignes commençant par une majuscule. Dans notre exemple :

Olivier:29:0298333242:Brest devenez
Myriam:30:0434214452:Nimes nez

Manipulation de fichiers(La commande grep)

vous recherchez «ahmaed» et qu'une ligne contient «AHMAD», grep ne la renverra pas. Pour que grep renvoie toutes les lignes qui contiennent «ahmaed» c-à-d ahmaed ou AHMAD, peu importent les majuscules et les minuscules, utilisez l'option -i :

grep -i ahmad nom_fichier

grep ^[a-d] nom_fichier

On va obtenir toutes les lignes commençant par les caractères compris entre a et d

grep Brest carnet-adresse

Permet d'obtenir les lignes contenant la chaîne de caractère Brest.

grep nez carnet-adresse

Permet d'obtenir les lignes contenant la chaîne de caractère nez.

Utilisation d'Unix : Ch. IV. Les commandes élémentaires

Manipulation de fichiers(La commande grep)

grep -w nez carnet-adresse

Permet d'obtenir les lignes contenant exactement (-w) la chaîne de caractère nez.

grep -v nez carnet-adresse

Permet d'obtenir les lignes ne (-v) contenant exactement (-w) la chaîne de caractère nez.

grep -wn nez carnet-adresse

Permet d'obtenir les lignes contenant exactement (-w) la chaîne de caractère nez, plus le numéro de chaque ligne contenant la chaîne est numérotée (n) soit :

5 : Myriam:30:0434214452:Nimes nez
6 : yvonne:92:013344433:Palaiseau nez

Manipulation de fichiers(La commande grep)

grep -wc nez carnet-adresse

↔ grep -w nez carnet-adresse | wc -l

Permet de compter les lignes contenant exactement (-w) la chaîne de caractère nez soit : 2

grep -x 'Myriam:30:0434214452:Nimes' carnet-adresse

ligne correspondant exactement à la chaîne soit :

Myriam:30:0434214452:Nimes

Lister tous les fichiers qui commencent par une majuscule

Ces commandes sont équivalents :

ls [[:upper :]]*

ls | grep ^[A-Z]

find . –type f -name [A-Z]*

Chapitre V

Droits d'accès aux fichiers

Notion de droits (permission)

les droits d'accès aux fichiers (appelés aussi modes ou permissions) sont un point essentiel du système Unix. Ils permettent de définir des droits différents sur un même fichier selon la catégorie d'utilisateurs.

Ainsi les manipulations de fichiers sont restreintes selon les droits alloués à chaque fichier. A chaque catégorie d'utilisateur correspond des droits spécifiques sur un fichier.

Affichage des droits

Pour afficher les droits alloués à un fichier, il faut utiliser la commande ls –l qui permet de lister les fichiers d'un répertoire avec toutes les informations connexes possibles dont les droits du fichiers.

Exemple d'affichage de droits

```
hduser@master:~/Desktop$ ls -l
total 4
-rw-rw-r-- 1 hduser hduser    0 Mar 13 04:36 file
drwxrwxr-x 2 hduser hduser 4096 Mar 13 04:56 rep1
```

total 4	Taille totale en Kilooctets (ko)								
-	rw-	rw-	r--	1	hduser	hduser	0	Mar 13 04 :36	file
d	rwx	rwx	r-x	2	hduser	hduser	4096	Mar 13 04 :56	rep1
Type de fichier	**Droits propriétaire (u)**	**Droits des membres de groupe (g)**	**Droits des autres**	**Nombre de liens (hard link)**	**Propriétaire (u)**	**Groupe (g)**	**Taille en Octets**	**Date et heure de dernière modification**	**Nom du fichier**

Droits d'accès

Les droits d'accès d'un fichier sont au nombre de trois :

Lecture : autorise l'accès en lecture du fichier. Cet accès est désigné par la lettre r (Read).

Ecriture : autorise la modification et la suppression du fichier. Il est désigné par la lettre w (Write).

Exécution : autorise l'exécution d'un fichier et l'ouverture d'un répertoire. Il est désigné par la lettre x (eXecute).

Catégories d'utilisateur

A un fichier on affecte les droits correspondants à trois catégories d'utilisateurs :

Propriétaire : c'est la personne qui crée le fichier. Il est désigné par la lettre u (owner).

Groupe : c'est un ensemble d'utilisateurs ayant certaines permissions pour un fichier. Il est désigné par la lettre g (groupe)

Autres : tous les utilisateurs qui ne sont ni propriétaires ni faisant parti du même groupe que le propriétaire. On le désigne par o (other).

Type de fichier :

d : (Directory) : indique si l'élément est un dossier ;

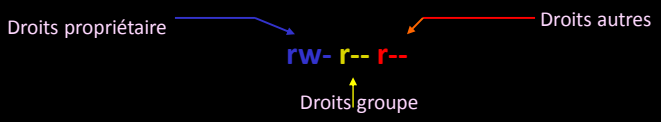
- : Un fichier ordinaire (simple)

l : (Link) : indique si l'élément est un lien (raccourci) ;

Identification des droits

A chaque catégorie d'utilisateur on associe un triplet de droits : lecture, écriture et exécution. Au total (3*3) sont affectés à chaque fichier.

Lorsqu'un droit est alloué, on voit la lettre correspondante (r, w ou x). Si le droit est refusé, on voit un tiret (-).



Interprétation des droits

	Fichier	Répertoire
r	Visualisation du contenu du fichier	Possibilité de lister le contenu du répertoire
w	Modification du contenu du fichier	Possibilité de créer et de détruire des fichiers
x	Exécution possible du fichier	Permet de traverser le répertoire pour atteindre un fichier

Utilisation d'Unix : Ch. V. Droits d'accès aux fichiers

Changement des droits

Il est offert au propriétaire d'un fichier (et seulement à lui seul) de modifier les droits du fichier. C'est-à-dire qu'il peut supprimer (-) des droits ou bien en rajouter (+) de nouveaux à chacune des catégories des trois utilisateurs.

Pour cela, on doit utiliser le commande chmod (change mode) selon la syntaxe suivante (il faut être root ou propriétaire du fichier) :

chmod droits fichiers

pour affecter à chaque catégorie les droits voulus, on peut utiliser une notation symbolique (Décimale) selon la syntaxe :

chmod catégorie+opération+liste_des_droits fichier

où les termes catégorie, opération et liste des droits doivent êtres respectivement remplacés par leur notation décrite dans les tableaux suivants.

Changement des droits

Catégorie	Description
u	propriétaire
g	groupe
o	autres

Opération	Description
+	ajouter
-	retirer
=	définir

Droit	Description
r	lecture
w	écriture
x	exécution

Utilisation d'Unix : Ch. V. Droits d'accès aux fichiers

Changement des droits

- On utilise le signe + pour ajouter un droit, - pour en enlever un, avec les lettres r, w, x (lecture, écriture, exécution):

  chmod -w nom_fichier : enlever les droits d'accès en écriture sur le fichier pour tout le monde

- Pour appliquer les changements seulement à certains niveaux d'appartenance, il faut utiliser les lettres u pour user (propriétaire du fichier), g pour group (utilisateurs appartenant au groupe du fichier), et o pour other (tous les autres) comme ceci :

  chmod ug +wx nom_fichier : ajouter, pour le propriétaire et les membres du groupe, les droits en écriture et en exécution sur le fichier

Changement des droits

Exemple :

chmod g=rwx temps.txt : alloue au groupe tous les droits.

chmod g-w temps.txt : retire aux groupe le droit d'écriture

chmod o+r temps.txt : rajoute aux autres le droit en lecture

chmod 600 rapport.txt // 600 ←→ rw- --- ---

Et toujours… -R pour affecter récursivement Le paramètre -R existe aussi pour chmod. Si vous affectez des droits sur un dossier avec -R, tous ses fichiers et sous-dossiers récupèreront le même droit.

Changement des droits

Remarque :

chmod g=rx temps.txt : alloue au groupe les droits de lecture et l'exécution et au même temps supprime le droit de l'ecriture.

En général définir un droit pour un propriétaire avec (=) va supprimer les autres droits de même propriétaire.

Avant de changer les droits :

```
hduser@master:~/Desktop$ ls -l
total 0
-rw-rw-r-- 1 hduser hduser 0 Mar 16 10:15 fichier
```

Après le changement des droits

```
hduser@master:~/Desktop$ chmod u=r,g=w,o=x fichier
hduser@master:~/Desktop$ ls -l
total 0
-r---w---x 1 hduser hduser 0 Mar 16 10:15 fichier
```

Ces droits sont supprimés

Exemple

Changement des droits

Il existe une autre façon d'indiquer les permissions (la notation octale): On utilise un nombre de 3 chiffres (le 1^{er} pour le propriétaire, le $2^{ème}$ pour les membres du groupes, et le $3^{ème}$ pour tous les autres). Les chiffes veulent dire :

Droit	Chiffre
r	4
w	2
x	1

Exemple : rwxr-xr-- ←---→ 754

r	w	x	r	-	x	r	-	-
4	2	1	4	0	1	4	0	0
4 + 2 + 1			4 + 0 + 1			4 + 0 + 0		
7			5			4		

Gestion des propriétaires d'un fichier (chown)

Seul l'utilisateur root peut changer le propriétaire d'un fichier.

Supposons que user possède dans son répertoire personnel un fichier appelé rapport.txt.

```
root@ubuntu:~ $  ls -l  rapport.txt
-rw-r--r-- 1 user  userg   0   2020-11-15   23:14   rapport.txt
```

On souhaite donner ce fichier à user1. C'est là qu'intervient la commande chown.

chown : changer le propriétaire d'un fichier. La commande chown, qui doit être utilisée en tant que root, attend deux paramètres au moins :

- le nom du nouveau propriétaire ;
- le nom du fichier à modifier.
- Cela donne donc :

```
root@ubuntu :~# chown user1 rapport.txt
```

Gestion des propriétaires d'un fichier (chown)

On peut voir ensuite que user1 est bien le nouveau propriétaire du fichier :

```
root@ubuntu:~# ls -l rapport.txt
-rw-r--r--  1  user1  usergr  0  2020-11-15  23:14  rapport.txt
```

il appartient toujours au groupe user !

Changer le groupe propriétaire d'un fichier (chgrp)

chgrp s'utilise exactement de la même manière que chown à la différence près qu'il affecte cette fois le groupe propriétaire d'un fichier.

```
root@ubuntu:~# chgrp  usergr1  rapport.txt
```

Cette commande affectera le fichier rapport.txt au groupe amis.

Un petit ls -l nous confirmera que rapport.txt appartient désormais à user1 et au groupe usergr1 :

Gestion des propriétaires d'un fichier (chown)

chown peut aussi changer le groupe propriétaire d'un fichier !

```
root@ubuntu:~# chown  user1 : usergr1  rapport.txt
```

Cela affectera le fichier à l'utilisateur user1 et au groupe usergr1.

Il suffit de séparer par un symbole deux points (« : ») le nom du nouvel utilisateur (à gauche) et le nom du nouveau groupe (à droite).

-R : affecter récursivement les sous-dossiers

l'option -R de chown permet de modifie tous les sous-dossiers et fichiers contenus dans un dossier pour y affecter un nouvel utilisateur (et un nouveau groupe).

Exemple: si je suis X et que je veux donner tout le contenu du dossier personnel de user à user1 (et au groupe user1gr):

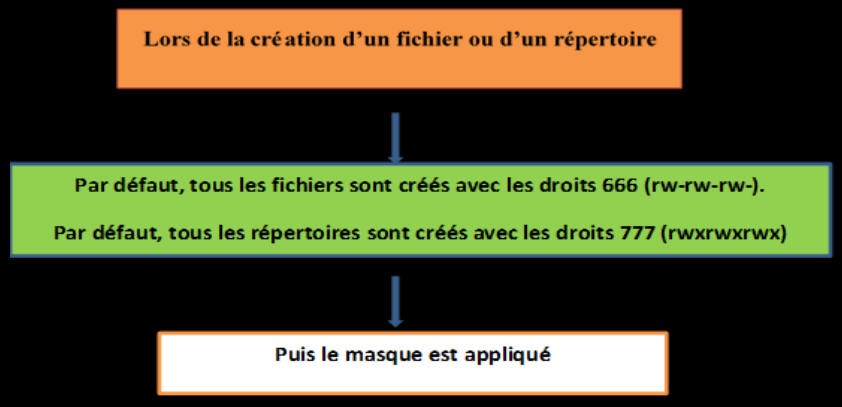
```
root@ubuntu:~# chown -R user1:user1gr /home/user/…
```

Le masque

Lors de la création d'un fichier ou d'un répertoire, des droits leur sont automatiquement assignés. Généralement c'est rw- r-- r--- (644) pour les fichiers et rwx r-x r-x (755) pour les répertoires. Ces valeurs sont contrôlées par un masque lui-même modifiable par la commande umask. La commande umask prend comme paramètre une valeur octale dont chaque droit individuel sera supprimé des droits d'accès maximum du fichier ou du répertoire. Le masque sert à masquer quelques droits. Le masque se compose de trois valeurs. umask fixe les droits par défaut pour les fichiers nouvellement créés.

Utilisation d'Unix : Ch. V. Droits d'accès aux fichiers

Le masque



Pour afficher le masque :
ubuntu @ubuntu:$ umask

Le masque

Pour les fichiers :

Supposons que le masque est 022 et les droits par défaut sont rw-rw-rw (666), pour trouver les droits après l'application du masque on va faire la soustraction élément par élément comme suit :

Droits par défaut (666)	r	w	-	r	w	-	r	w	-
Masque (022)	-	-	-	-	w	-	-	w	-
Droits après l'application du masque (644) ?????	r	w	-	r	-	-	r	-	-

Simple calcule pour trouver les droits à en appliquant un masque donné.

```
   666    (droits par défaut)
-  022    (masque)
--------------------------------
   644      (Droits après l'application du masque)
```

Utilisation d'Unix : Ch. V. Droits d'accès aux fichiers

Le masque

Pour les répertoires :

Supposons que le masque est 022 et les droits par défaut sont rwxrwxrwx (777), pour trouver les droits après l'application du masque, on va faire la soustraction élément par élément comme suit :

Droits par défaut (777)	r	w	x	r	w	x	r	w	x
Masque (022)	-	-	-	-	w	-	-	w	-
Droits après l'application du masque (755) ?????	r	w	x	r	-	x	r	-	x

Simple calcule pour trouver les droits à en appliquant un masque donné.

```
   777    (droits par défaut)
-  022    (masque)
--------------------------------
   755      (Droits après l'application du masque)
```

La masque

Pour modifier le masque à partir de la ligne de commande on utilise la commande umask :

ubuntu @ubuntu:$ umask 022

Après avoir modifié le masque, umask fixe les droits par défaut (ici rw-r--r-- (644)) pour les fichiers nouvellement créés .

r	w	-	r	-	-	r	-	-

Après avoir modifié le masque, umask fixe les droits par défaut (ici rwxr-xr-x (755)) pour les répertoires nouvellement créés .

r	w	x	r	-	x	r	-	x

Chapitre VI

La compression

Compression

Le but de la compression est de réduire la taille des données en utilisant un programme de compression. Un fichier compressé n'est plus utilisable tel quel à mois de le décompresser. Les programmes de compression suivants remplacent le fichier spécifié en argument par un autre d'extension caractéristique du programme.

Utilisation d'Unix : Ch. VI. La compression

Compresseur gzip et décompresseur gunzip

Le programme gzip, compress un ou plusieurs fichiers en leur rajoutant l'extension par défaut .gz. ces option sont :

-Compresse un ou plusieurs fichiers (pas les répertoires) dont le nom est passé en paramètre.

-Le fichier source (initial non compressé) est supprimé et seul subsiste le fichier compressé.

-Le fichier compressé qui apparaît porte le même nom que le fichier initial avec l'extension .gz ajoutée à la fin.

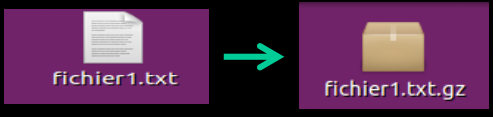
Compresseur gzip et décompresseur gunzip

-La compression

Syntaxe : gzip fichier1 <fichier2…>

Exemple : gzip tpc.tar

Cet exemple compresse le fichier tpc.tar en tpc.tar.gz



-La décompression

Syntaxe : gunzip fichier1.gz <fichier2.gz…>

Exemple : gunzip tpc.tar.gz

Cet exemple décompresse le fichier tpc.tar.gz en tpc.tar

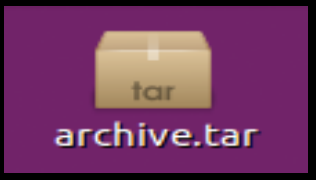
Remarque :

-Le programme gunzip possède les mêmes options que gzip.

-Pour compresser un répertoire il faut l'archiver d'abord.

Archiver

Archiver des fichiers consiste en les réunir en un seul autre fichier. Ce dernier n'est pas compressé c'est-à-dire que sa taille est égale à la somme ce celle des fichiers qu'il regroupe.



Archiver

Le programme d'archivage tar

Pour archiver des fichiers, on utilise le programme tar. Dont les principales options sont :

- c (Create) : pour créer une archive
- x (eXtrate) : pour extraire les fichiers d'une archive
- t (lisT) : pour affiche la liste d'une archive
- f (Force) : pour forcer le remplacement de fichiers
- v (Verbose) : pour le mode verbeux (convertir les caractères spéciaux des fichiers binaires en caractères affichables)
- z (gZip) : traite les fichier avec gzip (compression après archivage, décompression avant extraction et décompression temporaire pour afficher la liste des fichiers).

Utilisation d'Unix : Ch. VI. La compression

Archiver

Le programme d'archivage tar

Créer une archive :

Syntaxe : tar cvf nom_archive.tar fichier ou répertoire (motif)

Exemple : tar cvf tpc.tar *.c

Dans cet exemple, on crée une archive qui porte le nom tpc.tar qui contient tous les fichier d'extension .c du répertoire courant. Le motif est une expression régulière du Shell qui peut contenir des méta-caractères.

Les fichiers archives doivent porter l'extension .tar

Archiver

Le programme d'archivage tar

Extraire les fichiers d'une archive :

On remplace l'option c (Create) par x (eXtract) pour extraire les fichier d'une archive.

Syntaxe : tar xvf fichier.tar

Exemple : tar xvf tpc.tar

On peut n'extraire de l'archive que les fichiers satisfaisant un motif :

Exemple : tar xvf tpc.tar poly*

Dans cet exemple seuls les fichiers dont le nom commence par poly sont extraits

Archivage et compression automatique

Pour compresser automatiquement le fichier archive pendant sa création, on utilise l'option z (gzip).

Syntaxe : tar zcvf fichier.tar.gz motif

Exemple : tar zcvf tpc.tar.gz *.c

Les fichiers compressés avec gzip ont .gz pour extension.

Contenu d'une archive

Pour visualiser la liste des noms des fichiers contenus dans une archive, on utilise l'option t.

Syntaxe : tar t fichier.tar

Exemple : tar t tpc.tar

Si le fichier est compressé par gzip, on ajoute l'option z.

Syntaxe : tar zt fichier.tar

Exemple : tar zt tpc.tar.gz

Chapitre VIII

Redirection des E/S
Tubes, Processus

Redirection des E/S

La plupart des commandes que nous avons vues jusqu'ici produisent des sorties sur le terminal. Il est important de pouvoir remplacer, aussi bien en entrée qu'en sortie, le terminal par un fichier. Par exemple :

$ ls : imprime la liste des fichiers du répertoire courant

$ ls >fichier.liste : produit la même liste qui est placée dans le fichier fichier.list. Rien n'est affiché sur le terminal.

Utilisation d'Unix : Redirection des E/S

Redirection des E/S

Le symbole > signifie « mettre la sortie dans le fichier qui suit plutôt que sur le terminal ». Le fichier est créé s'il n'existe pas. S'il existe, son ancien contenu disparaît.

Voici un autre exemple où plusieurs fichiers sont combinés pour n'en former qu'un seul en redirigeant la sortie de cat vers un fichier :

```
$ cat  f1  f2  f3  >temp
```

Redirection des E/S

Le symbole >> se comporte de la même manière que > mais il signifie « ajouter à la fin de ». C'est-à-dire :

```
$ cat  f1  f2  f3  >>temp
```

copie le contenus de f1, f2 et f3 à la suite de ce qui se trouvait précédemment dans temp sans l'effacer.

Utilisation d'Unix : Redirection des E/S

Redirection des E/S

D'une façon similaire, le symbole < indique à un programme de prendre son entrée depuis un fichier et non pas depuis le terminal. Par exemple pour obtenir la liste par ordre alphabétique des utilisateurs d'un système :

```
$ who  >temp
$ sort <temp
```

Puisque who imprime une ligne par utilisateur connecté, et que wc –l compte les lignes de son entrée, vous pouvez compter les utilisateur connectés par :

```
$ who  >temp
$ wc -l <temp
```

Redirection des E/S

Les caractères < et > sont interprétés par le shell et non pas par les commandes.

la commande : $ sort <temp

trie le contenu du fichier temp, comme le fait

$ sort temp

mais il y a une différence : la chaîne <temp est interprétée par le shell, sort ne voit pas le nom du fichier temp comme un argument mais comme entrée standard.

Redirection des E/S

La sortie erreur a été établie pour que les messages sortent toujours sur le terminal :

```
$ diff  fic1 fic2  <diff.out
diff: fic2: No such file or directory « message d'erreur »
$
```

Les tubes

Un tube (pipe en anglais) est un moyen de connecter la sortie d'un programme à l'entrée d'un autre sans passer par un fichier temporaire; un pipeline est la connexion d'un ou plusieurs programmes à travers des tubes.

Reprenons les exemples précédents en utilisant des tubes au lieu de fichiers temporaires. La barre verticale | indique au shell la présence d'un tube.

$ who | sort	liste alphabétique des utilisateurs
$ who | wc -l	compte les utilisateurs

Utilisation d'Unix : Les tubes

Les tubes

Tout programme capable de lires depuis le terminal est capable de lire depuis un tube; tout programme qui écrit sur le terminal peut écrire vers un tube. Le nombre de pipeline dans un programme n'est pas limité.

$ ls | pr –3 | lpr

réalise sur l'imprimante une liste sur trois colonnes les noms de fichiers du répertoire courant.

Les messages d'erreur doivent être traités différemment, pour éviter qu'ils ne se perdent dans un fichier ou dans un tube. Ainsi chaque commande possède une sortie standard qui, par défaut est dirigée vers le terminal.

Les processus

On désigne par le terme processus une instance de programme en cours d'exécution et son environnement. Chaque processus s'exécute avec les droits de l'utilisateur qui l'a lancé.

En interne, le système identifie les processus de façon unique grâce à un numéro. Ce numéro est appelé PID (Process ID, identifiant de processus). Avec ce PID, le système sait qui a lancé le processus

Les processus

Linux permet un contrôle aisé des processus grâce aux signaux. Avec ceux-ci vous pouvez, par exemple, suspendre ou tuer un processus. Envoyez simplement de signal correspondant au processus et c'est fait. Vous serez limité à l'envoie de signaux à vos propres processus, pas aux processus lancés par un autre utilisateur. L'exception à la règle est root.

L'arborescence des processus

De même que pour les fichier, tout les processus en cours d'exécution sur un système Linux sont organisés sous forme d'arborescence, et chaque processus possède un numéro PID, ainsi que le numéro de son processus parent (PPID, Parent Process ID).

Les signaux

Chaque processus sous Unix est susceptible de réagir à des signaux qui lui sont envoyés. Il existe 64 signaux différents que l'on identifie soit par leur numéro (en partant de 1). Soit par leur nom symbolique. Les 32 signaux (de 33 à 64) sont des signaux temps réel.

Pour chacun de ces signaux, le processus peut redéfinir son propre comportement par défaut, sauf pour deux : le signal numéro 9 (KILL), et le signal numéro 19 (STOP).

Les signaux

le signal 9 tue un processus de façon irrémédiable, sans lui laisser le temps de se terminer correctement. C'est ce signal qu'il faut envoyer à des processus dont vous voulez vous débarrasser. Une liste complète des signaux est disponible en utilisant la commande kill –l.

Information sur les processus : ps et pstree

Ces deux commandes affichent une liste des processus existants sur le système, selon les critères que voulez.

Commande ps

Lancer cette commande sans argument montrera les processus dont vous êtes l'initiateur et sont rattachés au terminal que vous utilisez.

Information sur les processus : ps et pstree

Commande ps

Les options sont nombreuses, nous citrons que les plus courants :

a : affiche aussi les processus lancés par d'autres utilisateurs

u : affiche pour chaque processus le nom de l'utilisateur qui l'a lancé et l'heure de son lancement.

Utilisation d'Unix : Les processus

Information sur les processus : ps et pstree

Commande pstree

Affiche les processus sous forme d'arborescence et permet de les visualiser par leurs liens de parenté. Ainsi, pour tuer une série de processus de la même famille, il suffira d'en découvrir l'ancêtre commun. Vous aurez avantage à utiliser l'option –p, qui affiche le PID de chaque processus, ainsi que l'option –u, laquelle donne le nom de l'utilisateur ayant lancé le processus. L'arborescence étant longue, il est plus facile d'invoquer pstree de cette façon :

```
$ pstree  -up  | less
```

pour avoir une vue d'ensemble.

Envoyer des signaux aux processus : kill, killall, top

Commandes kill, killall

Ces deux commandes permettent d'envoyer des signaux à des processus. La commande kill attend un numéro de processus en argument, tandis que killall attend un nom de commande.

Les deux commandes peuvent, de façon optionnelle, recevoir un numéro de signal en argument. Par exemple, si vous voulez tuer le processus de PID 785, vous entrez la commande :

```
$ kill 785
```

Si vous voulez lui envoyer le signal 9, vous entrerez alors :

```
$ kill -9 785
```

Utilisation d'Unix : Les processus

Envoyer des signaux aux processus : kill, killall, top

Commande kill, killall

Supposons que vous vouliez tuer un processus pour lequel vous connaissez le nom de la commande. Au lieu de repérer le numéro de processus à l'aide de ps, vous pouvez tuer le processus directement :

```
$ killall  -9 netscape
```

Envoyer des signaux aux processus : kill, killall, top

Commande top

top est un programme tout en un : il remplit à la fois les fonctions de ps et kill.

Le programme se contrôle entièrement au clavier. Une aide est disponible en tapant sur h.

Utilisation d'Unix : Les processus

Lancement et manipulation de processus en arrière-plan

Pour interrompre une commande, lancer en premier plan, qui peut prendre plusieurs minutes et reprendre la ligne de commande ($) on peut soit :

- Utiliser les touches Ctrl+c pour tuer la commande,
- Utiliser les touches Ctrl+z pour placer le processus en arrière plan. Le processus et alors suspendu dans l'attente d'être relancer. On peut le relancer mais en le maintenant en arrière plan en tapant :$ bg (pour background).

Un processus qui tourne en tâche de fond, ou arrière plan, est appelé un job.

Lancement et manipulation de processus en arrière-plan

On peut lancer directement des processus en arrière plan en ajoutant une & à la fin de la commande.

Syntaxe : $ commande [options] [arguments] &

On peut également remettre ce processus au premier plan et attendre qu'il se termine en tapant fg (pour ForeGround, soit premier plan). Taper alors la séquence Ctrl+z, bg pour le remettre en arrière plan.

Utilisation d'Unix : Les processus

Lancement et manipulation de processus en arrière-plan

On peut lancer plusieurs jobs de cette façon : chacune de ces commandes recevra un numéro de job. La commande jobs du shell indique la liste de tous les jobs associés au shell courant. Le job précédé d'un signe + désigne le dernier processus mis en tâche de fond.

Pour remettre un job en particulier au premier plan, on peut alors taper fg <n> où <n> désigne le numéro de job, par exemple $ fg 5.