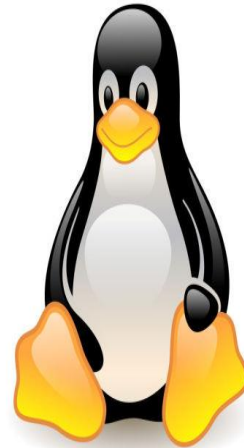


+ Module Linux

Enseignante
S. BEN YAALA



Redirections, Tubes et filtres

+ Plan

- Redirections
- Tubes de communications
- Filtres

+ Commande Linux

□ Une commande :

Entrée : arguments en entrée standard.

Sortie : une réponse en sortie standard

+

une réponse en sortie erreur standard.



+ Commande Linux

❑ Les redirections sont le détournement des 3 descripteurs de fichiers standards :

❑ l'entrée standard (notée 0) : le clavier ;

❑ la sortie standard (notée 1) : la console courante ;

❑ la sortie des erreurs (notée 2) : la console courante

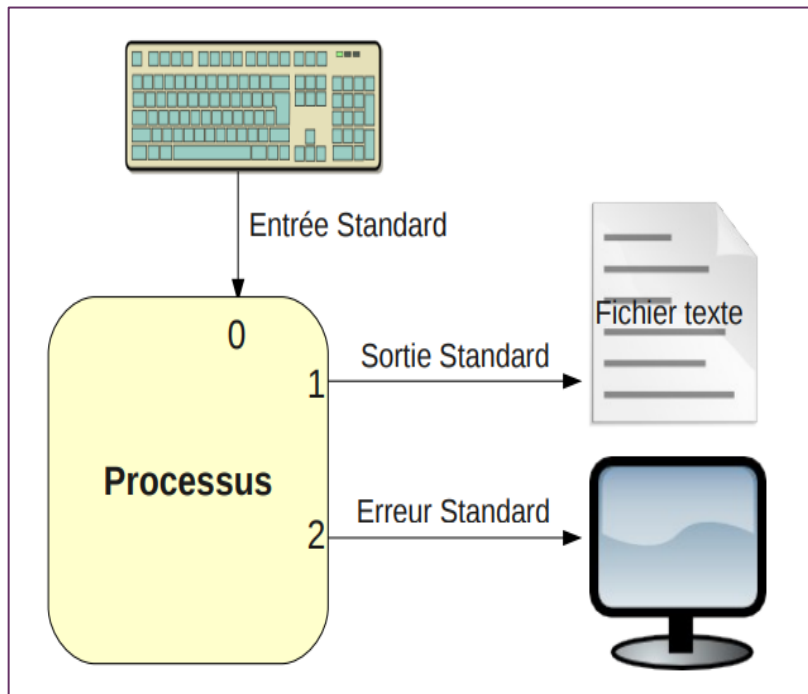
+ Redirections des entrées-sorties

■ Principe

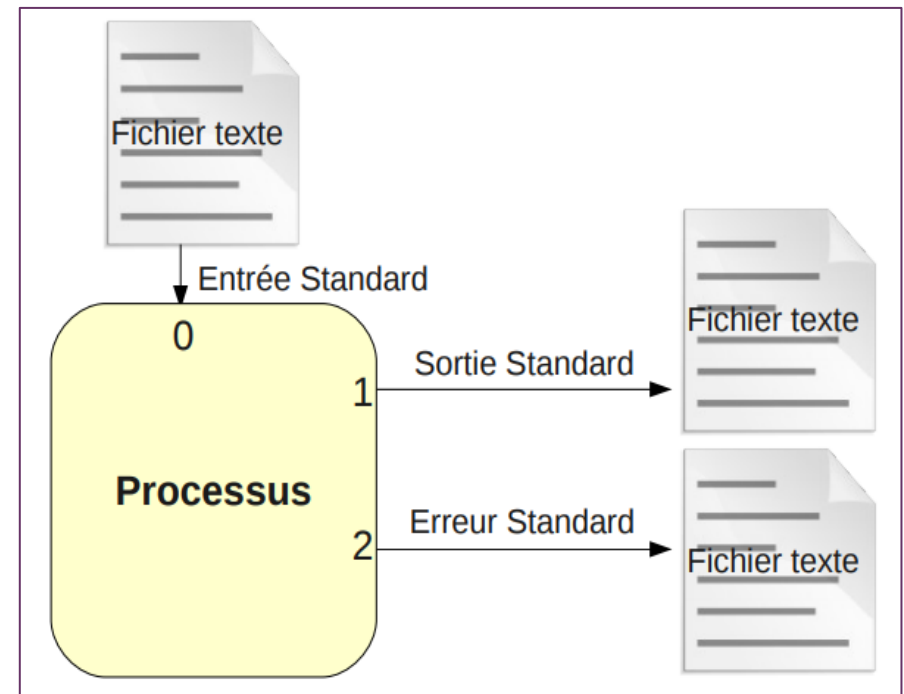
- Il s'agit de renvoyer le résultat d'une commande vers une sortie différente de la sortie normale .
- Valide pour une entrée ou une sortie

+ Redirection des entrées-sorties

□ Principe :



Redirection vers un fichier



Redirection vers des fichiers

+ Flux de redirection

❖ `>` et `>>` : renvoient le résultat vers un fichier.

- `>` : rediriger dans un nouveau fichier

```
#ls > file1
```

Le contenu de fichier sera supprimé automatiquement si le fichier existe déjà.

- `>>` : rediriger à la fin d'un fichier

```
#ls >> file1
```

Si le fichier n'existe pas, il sera créé. Sinon, les données seront ajoutées à la fin.

+ Flux de redirection

❖ < : lire depuis un fichier

< : permet d'indiquer d'où vient l'entrée qu'on envoie à la commande.

Exemple : Le fichier est lu par la commande cat

#cat < fichier1

+ Flux de redirection

❖ Redirection des erreurs : (2>, 2>> et 2>&1)

Pour chaque commande exécutée, il existe deux possibilités :

❑ Cas 1 : Tout va bien :

Le résultat de la commande sera affiché sur la sortie standard

❑ Cas 2 : Une erreur se produit

Le résultat de la commande s'affiche dans la sortie d'erreurs.

❖ Syntaxe

2> : Rediriger les erreurs dans un fichier à part

2>> : pour ajouter les erreurs à la fin du fichier.

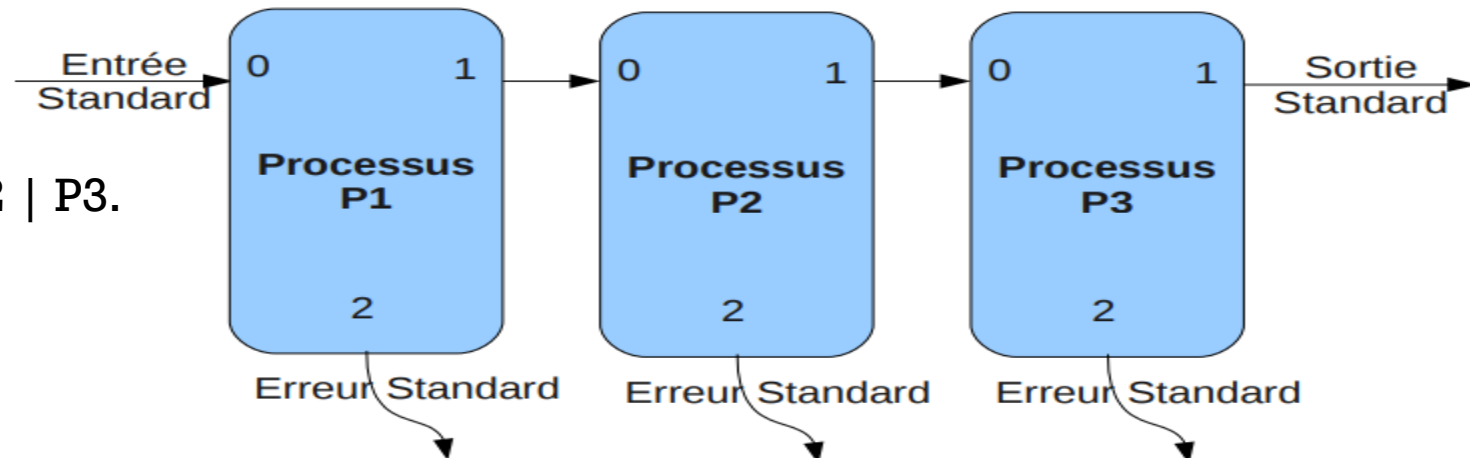
+ Tubes de communication

10

- Le symbole |, qui s'appelle "tube" ou "pipe", permet de relier deux commandes entre elles.
- Il s'agit d'utiliser le premier paramètre à gauche comme entrée de deuxième paramètre

commande1 | commande2 | commande3

- Redirections = assure la communication entre (fichier, processus).
- Tube = assure la communication entre deux processus.



+ Filtres

- Ce sont des programmes traitant des données qui proviennent de l'entrée standard.
- Très utiles en les combinant avec des filtres.

+ les commandes filtres courantes

■ La commande `sort`

Tri sur des lignes de texte :

■ Options

- `n` : ordre numérique
- `s` : lexicographique (par défaut)
- `d` : selon le dictionnaire
- `k` : Tri d'un champ particulier
- `r` : Tri en ordre inverse

Req: Les champs sont délimités par défaut par le caractère de tabulation mais il est possible de spécifier un autre caractère avec l'option "-t".

■ Exemples

Commande	Action
<code>\$ sort -n < /etc/passwd</code>	tri le fichier <code>/etc/passwd</code> par ordre numérique
<code>\$ sort -nt : -k 3 < /etc/passwd</code>	tri le fichier <code>passwd</code> par ordre numérique sur le 3ème champ avec <code>:</code> comme délimiteur de champs
<code>\$ sort -nrt : -k 3 < /etc/passwd</code>	même type de tri en présentant les résultats inversés

+ La commande grep

- La commande **grep** permet la recherche dans des fichiers d'une expression particulière.
- Les options basiques sont:
 - n** : numéro de ligne.
 - i** : ignorer majuscules et minuscules
 - v** : rechercher tout sauf la chaîne de caractères dans un fichier.

■ Exemples

- Rechercher la chaîne de caractères dans une arborescence `/etc/`

grep -r « chaîne » /etc/

- Afficher la ligne contenant `home` sans tenir compte des majuscules et minuscules :

grep -i « home » < /etc/passwd

- Afficher les lignes ne contenant pas `home` :

grep -v « home » < /etc/passwd

+ La commande grep

■ Recherche avec des expressions

Caractère spécial	Signification
.	Caractère quelconque
^	Début de ligne
\$	Fin de ligne
[]	Un des caractères entre les crochets
?	L'élément précédent est optionnel (peut être présent 0 ou 1 fois)
*	L'élément précédent peut être présent 0, 1 ou plusieurs fois
+	L'élément précédent doit être présent 1 ou plusieurs fois
	Ou
()	Groupement d'expressions

+ La commande grep

15

- Rechercher une chaîne de caractères 'chaine' en début de ligne dans un fichier

```
#grep -E ^chaine fichier1
```

- Rechercher une chaîne de caractères 'chaine' en fin de ligne dans un fichier

```
#grep -E chaine $ fichier1
```

- Rechercher une chaîne de caractères 'chaine' ou 'Chaine' dans un fichier

```
#grep -E [Cc]haine fichier1
```

+ La commande wc

- Cette commande permet de compter le nombre de lignes, de mots et de caractères dans un fichier.
- Options :
 - l : nombre de lignes,
 - w : nombre de mots
 - c : nombre de caractères.

■ Exemples

Commande	Action
\$ wc -l < /etc/passwd	compte le nombre de lignes dans <i>/etc/passwd</i>
\$ wc -c < /etc/passwd	compte le nombre de caractère dans <i>/etc/passwd</i>

+ La commande cut

■ Options :

- c : extraire des colonnes
- f : extraire des champs
- d : spécifier délimiteur

■ Exemples

Commande	Action
\$ cut -f3,7 -d : /etc/passwd	filtre les champs 3 et 7 de chaque ligne de <i>passwd</i> en considérant le caractère : comme délimiteur
\$ date cut -c1-3	filtre les caractères 1 à 3

+ La commande head

- Identifie le début d'un fichier (ou de l'entrée standard).
- Options :
 - n : nombre de lignes
 - c : nombre de caractères.
- Exemples

Commande	Action
<code>\$ head -c 1000 /etc/passwd</code>	édite à l'écran les 1000 premiers caractères du fichier
<code>\$ head -n 10 /etc/passwd</code>	édite les 10 premières lignes du fichier

+ La commande tail

- Identifie la fin d'un fichier ou de l'entrée standard.

- Options :

- c : nbre. de caractères

- n : nbre. de lignes

- b : nbre. de blocs (exemple 512 octets)

Exemples

Commande	Action
\$ tail -c 15 /etc/passwd	édite les 15 derniers caractères de <i>/etc/passwd</i>
\$ tail -n 5 /etc/passwd	édite les 5 dernières lignes de <i>/etc/passwd</i>

+ La commande tr

- Fonction : substituer ou supprimer des arguments
- Arguments : - Deux chaînes de caractères
 - La 1ère chaîne représente les caractères recherchés
 - La 2^{ème} représente le remplacement
- Options :
 - c : inverse la recherche
 - s : Traitement d'une seule occurrence
 - d : suppression

Commande	Action
\$ cat /etc/passwd tr : "\t"	remplace les caractères : par une tabulation
\$ cat /etc/passwd tr -d [A-Z]	supprime tous les caractères majuscule de A à Z
\$ last tr [:lower:] [:upper:]	remplace toutes les minuscules par des majuscules

+ Sed : fonction de substitution : s

21

- **s** permet de changer la 1ère ou toutes les occurrences d'une chaîne par une autre.
- Syntaxe :
 - sed "**s**/toto/TOTO/**g**" fichier : va changer toutes les occurrences de la chaîne toto par TOTO

+ Sed : La fonction de suppression : d

22

- La fonction de suppression d supprime les lignes comprises dans un intervalle donné.

- Syntaxe : **sed "20,30d" fichier1**

Cette commande va supprimer les lignes 20 à 30 du fichier 'fichier1'.

- On peut utiliser les expressions régulières :
 - sed "/toto/d" fichier1 : supprime les lignes contenant la chaîne toto
 - sed "/toto/!d" fichier : supprime toutes les lignes ne contenant pas la chaîne toto

En fait les lignes du fichier d'entrée ne sont pas supprimées, elles le sont au niveau de la sortie standard.

+ Uniq

- Prend en paramètre un fichier trié.
- Elle identifie uniquement identiques qui sont successives.

uniq doubl.txt

- Options :
 - d : afficher les lignes en double
 - c : nombre d'occurences

+ Find

24

- La commande find permet de retrouver des fichiers à partir de certains critères.
- Syntaxe : **find repertoire-de-recherche critère-de-recherche**
- Options:
 - name : nom du fichier , -perm : droits d'accès
 - link : nombre de liens , -user : propriétaire
 - group : recherche suivant le groupe , -type : recherche suivant le type
 - size : recherche suivant la taille , -atime : date de dernier accès en lecture
 - mtime : date de dernière modification du fichier
 - ctime : recherche sur la date de création du fichier

+ Find

- Utilisation de la commande find
- La commande find permet de chercher des fichiers, et éventuellement d'exécuter

une action dessus :

- Rechercher un fichier nommé fichier1 dans le répertoire

```
/home/ludo    #find /home/ludo -name fichier1 -print
```

+ Find

- Rechercher tous les fichiers appartenant à l'utilisateur tekup et les déplacer dans son répertoire home

```
# find / -user tekup 2>/dev/null
```

```
# find / -user tekup -exec mv {} /home/tekup/ \; 2>/dev/null
```

- Rechercher tous les fichiers commençant par fichier ET appartenant à l'utilisateur tekup et les copier dans le répertoire /root

```
#find / -name "fichier*" -a -user tekup -exec cp {} /root/* \; 2>/dev/null
```

Option	Signification
-name	Recherche par nom de fichier.
-type	Recherche par type de fichier.
-user	Recherche par propriétaire.
-group	Recherche par appartenance à un groupe.
-size	Recherche par taille de fichier.
-atime	Recherche par date de dernier accès.
-mtime	Recherche par date de dernière modification.
-ctime	Recherche par date de création.
-perm	Recherche par autorisations d'accès.
-links	Recherche par nombre de liens au fichier.

■ Pour les options -size, -atime, -mtime, -ctime et -links, Il faut spécifier une valeur, précédée par le signe ``+" pour

■ ``supérieur à", ``-" pour ``inférieur à", ou rien pour ``égal à". Par exemple :

find . -mtime -3 -print

■ Cette commande affiche les fichiers dont les dernières modifications remontent à moins de 3 jours (donc tous les fichiers modifiés entre aujourd'hui et il y a trois jours seront affichés).

■ De même, +5 afficherait les fichiers dont les dernières modifications remontent à plus de 5 jours.

+ Locate

- la commande locate cherche les fichiers dans une base de données.

Syntaxe : locate fichier

- Il faut mettre à jour la base de locate :

Syntaxe : updatedb



Merci