



GNU/Linux

Les processus



Sommaire

I. Les processus

- Définition

- États d'un processus

- Gestion des processus et commandes de base

- Les redirection I/O

- Les tubes

II. Les filtres

- Définition

- Exemples



Les processus

Définition

Un processus est la représentation en mémoire d'un programme qui est exécuté par un processeur . Il regroupe toutes les informations nécessaires à son contrôle (code machine, état des registres du processeur, mémoire allouée, descripteurs de fichiers ouverts, etc.). Un même programme peut faire l'objet de plusieurs processus.

- Un processus (processus fils) est toujours créé par un autre processus (processus père) sauf le 1^{er} processus appelé systemd

- **Principales caractéristiques d'un processus:**

- Un identifiant PID

- Un identifiant du processus parent (PPID)

- Son propriétaire

- Sa priorité

- ...

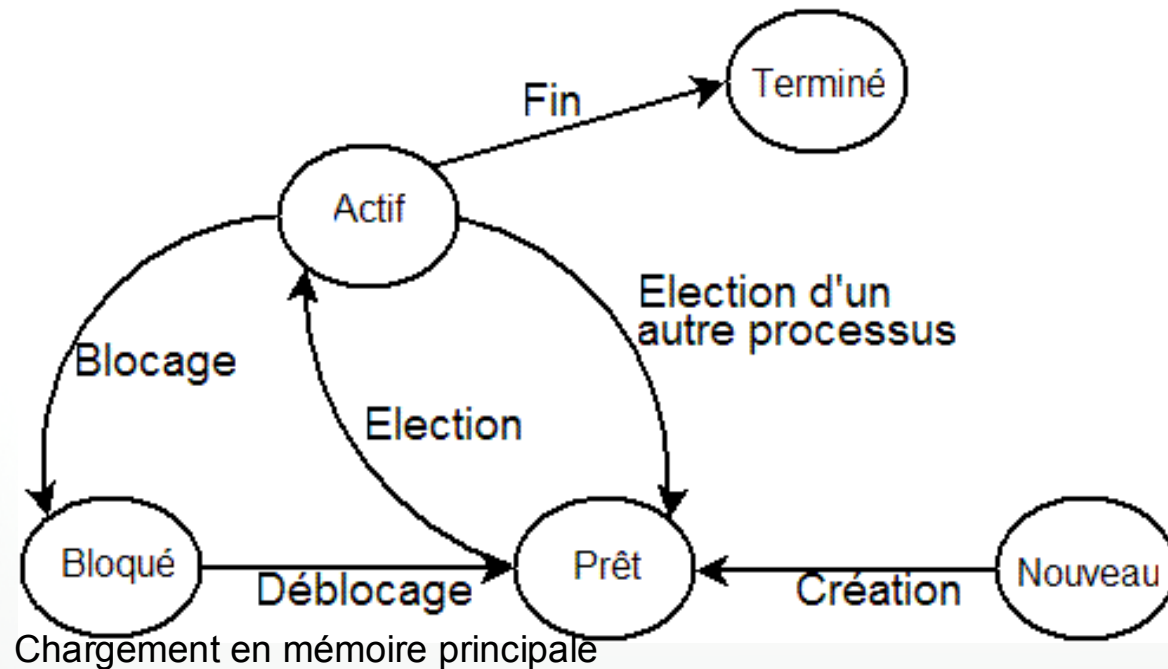
quelques commandes Pour afficher les processus en cours d'exécution : **ps, top**

Pour afficher la hiérarchie des processus : **pstree**

Les processus

États d'un processus

Depuis sa création jusqu'à sa destruction (Fin), chaque processus va transiter par un certain nombre d'états.



Les processus

Quelques informations retournées par la commande `ps` :

	PID	TTY	STAT	TIME	CMD
	3899	pts/1	R	0:00.08	bash
	4180	pts/1	S	0:00.04	ps

Annotations :

- Numéro de processus (pointe vers PID)
- Terminal associé (pointe vers TTY)
- État du processus (pointe vers STAT)
- Temps CPU utilisé (pointe vers TIME)
- Commande exécutée (pointe vers CMD)

**Terminal
associé**

**État du
processus:**

R	Actif
T	Bloqué
S	Endormi
IW	Swappé
Z	Zombie

La commande `ps` sans option affiche seulement les processus attachés au terminal exécuté.

Les processus

Gestion des processus et commandes de base

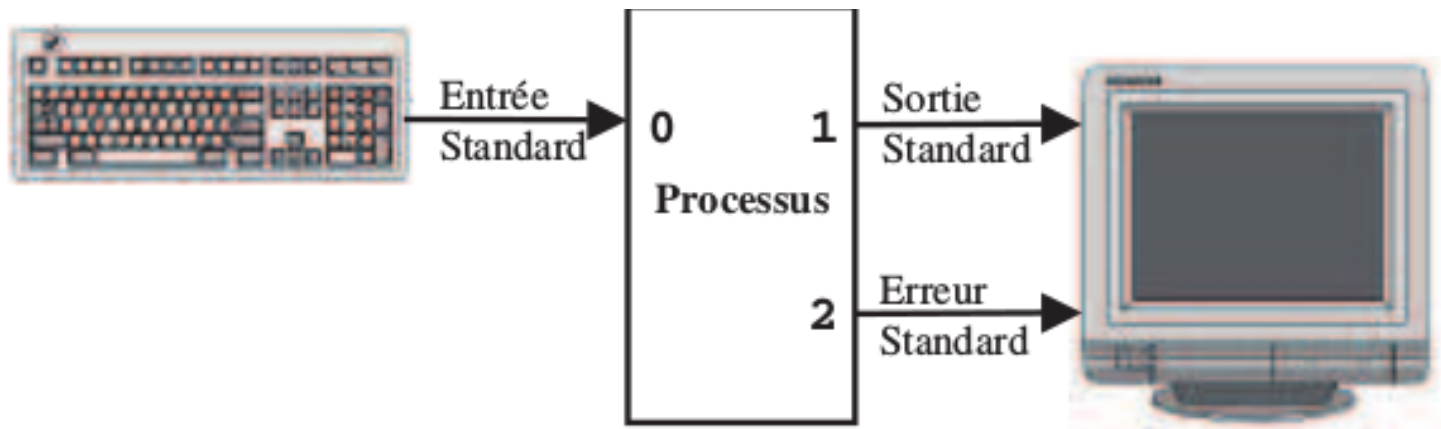
- **Pour afficher tous les processus en cours d'exécution on execute : `ps aux`**
Options :
 - a Liste tous les processus actifs
 - u Format d'affichage long
 - x Inclut les processus sans terminal
- **Pour Terminer (tuer) un processus: `kill -9 <PID>`**
- **Exécuter un processus en arrière-plan: `commande &` (le terminal n'est pas bloqué)**

Les processus

Les redirection I/O

Le principe de redirection

On appelle processus, ou tâche, l'exécution d'un programme exécutable. Au lancement de chaque processus, l'interpréteur de commandes ouvre d'office une entrée standard (par défaut le clavier), une sortie standard (par défaut l'écran) et la sortie d'erreur standard (par défaut l'écran)



Ces entrées-sorties standard peuvent être redirigées vers un fichier, un tube, un périphérique. La redirection de la sortie standard consiste à renvoyer le texte qui apparaît à l'écran vers un fichier. Aucune information n'apparaîtra à l'écran, hormis celles qui transitent par la sortie d'erreur standard.

Les processus

Les redirection I/O

Symbole	Signification
<	Redirige l'entrée standard
<<	Concatène l'entrée standard
>	Redirige la sortie standard
>>	Concatène la sortie standard
2>	Redirige la sortie d'erreur

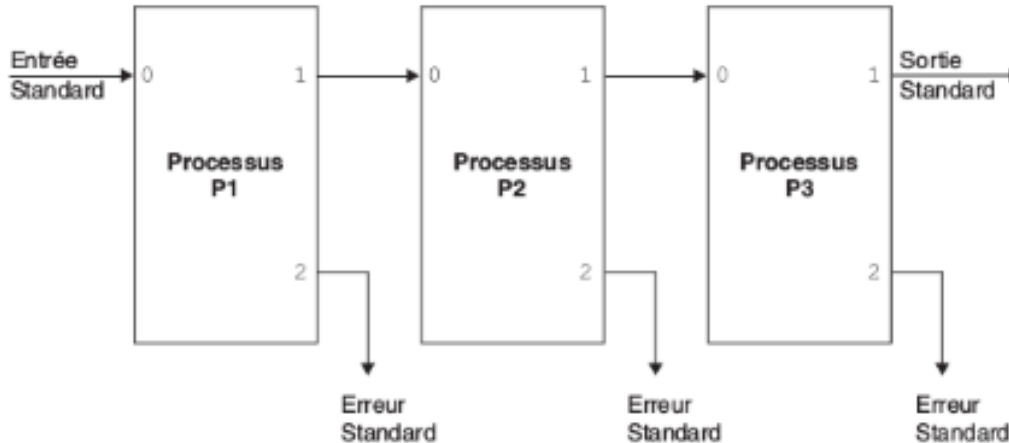
Exemples:

<code>ls /home > liste</code>	Le résultat de la commande <code>ls</code> est enregistré dans le fichier <code>liste</code>
<code>date >> liste</code>	Ajoute à la fin du fichier <code>liste</code> la sortie de la commande <code>date</code>
<code>wc -l < liste</code>	Envoie comme entrée à la commande <code>wc -l</code> le fichier <code>liste</code>

Les processus

Les tubes: | (AltGr6)

Un tube (*pipe* en anglais) est un flot de données qui permet de relier la sortie standard d'une commande à l'entrée standard d'une autre commande sans passer par un fichier temporaire



Exemple: Compter le nombre de fichiers dans le répertoire courant.

Sans le pipe:

```
ls > tmp ; wc -l < tmp
```

avec un pipe:

```
ls | wc -l
```

Les filtres

Un filtre est une commande qui lit les données sur l'entrée standard, les traite et les écrit sur la sortie standard. Le concept tube ou pipe avec sa simplicité devient un outil très puissant dans GNU/Linux et propose un choix très vaste de filtres. Les filtres les plus utilisés sont les suivant :

`cat` Affiche le contenu des fichiers passés en argument (par défaut, stdin)

`more` Affiche page par page les fichier passés en argument (par défaut, stdin)

`tee` Recopie l'entrée standard sur la sortie standard et dans le fichier passé en argument.

`cut` Sélectionne uniquement certaines colonnes du fichier passé en paramètre

`grep` Recherche dans le fichier passé en argument les lignes vérifiant une expression régulière donnée.

`find` Permet de retrouver des fichiers à partir de certains nombres de critères

Et encore plein d'autres...

`which, locate, tail, head, wc, tr ...`

Processus maîtrisés, ressources sécurisées..

Linux Process Control

- Stop
- End
- Kill

