

I2180
LINUX
APPELS SYSTÈME
PIPE

PIPES : GENERALITES

- En ligne de commande : `ls -l | wc -l`
- Un *pipe* (tube) est un moyen de communication entre processus.
- Utilisé généralement de manière FIFO et unidirectionnel :
 - un seul processus lit
 - un seul processus écrit

PIPE

Création d'un pipe par programmation :

```
#include <unistd.h>

int pipe(int filedes[2])
```

Où: filedes : tableau vide qui sera initialisé par pipe

filedes[0] : permettra de lire dans le pipe

filedes[1] : permettra d'écrire dans le pipe

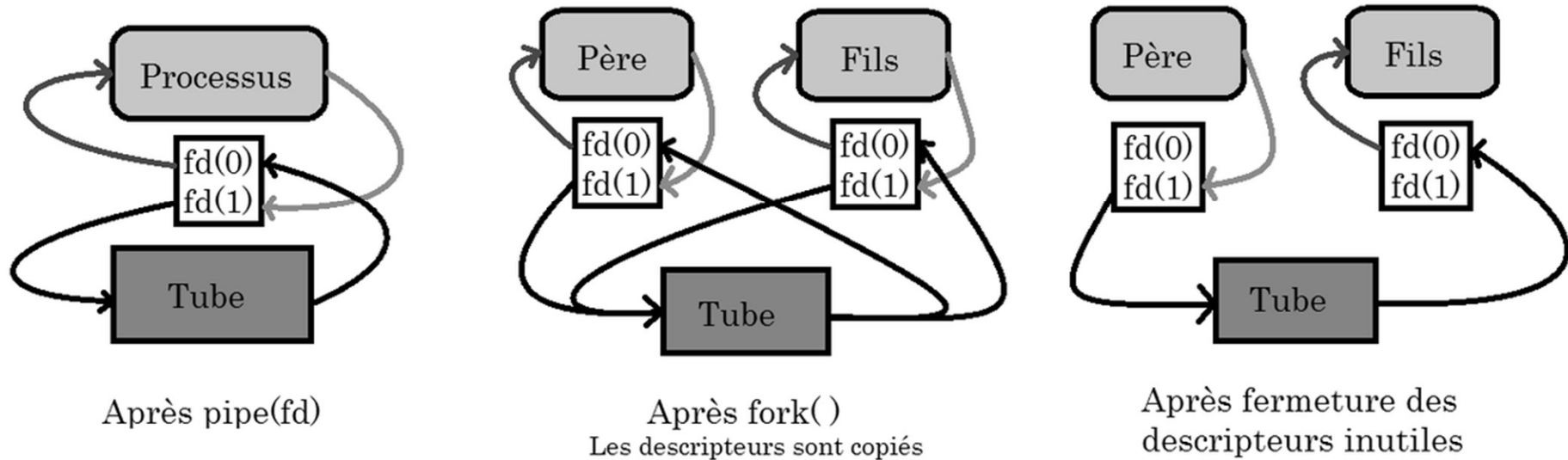
REMARQUE

- Tubes anonymes
 - Communication entre un processus père et un fils
→ Objet de ce cours
- Tubes nommés
 - Communication entre processus quelconques
 - `int mkfifo(char* name, mode_t mode)`
→ Pas l'objet de ce cours

CONFIGURATION D'UN PIPE

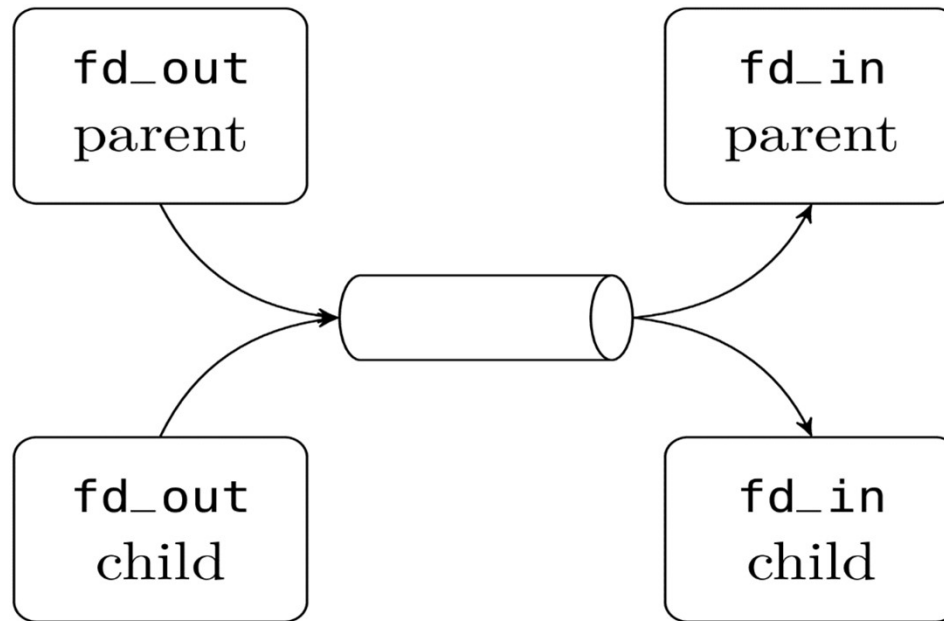
- Création du pipe AVANT la création du processus fils
- Les 2 processus ont ainsi accès aux descripteurs de fichiers du pipe
- Dans chaque processus, fermer le descripteur de fichier non utilisé

CONFIGURATION D'UN PIPE



[https://fr.wikipedia.org/wiki/Tube_\(informatique\)](https://fr.wikipedia.org/wiki/Tube_(informatique))

CONFIGURATION D'UN PIPE



- La fermeture d'un descripteur de fichier (*close*) permet d'indiquer au système d'exploitation qu'un processus n'utilise plus ce descripteur.
- Une lecture sur un descripteur de fichier totalement fermé (plus aucun processus n'a de référence vers ce descripteur) renvoie 0 (EOF)

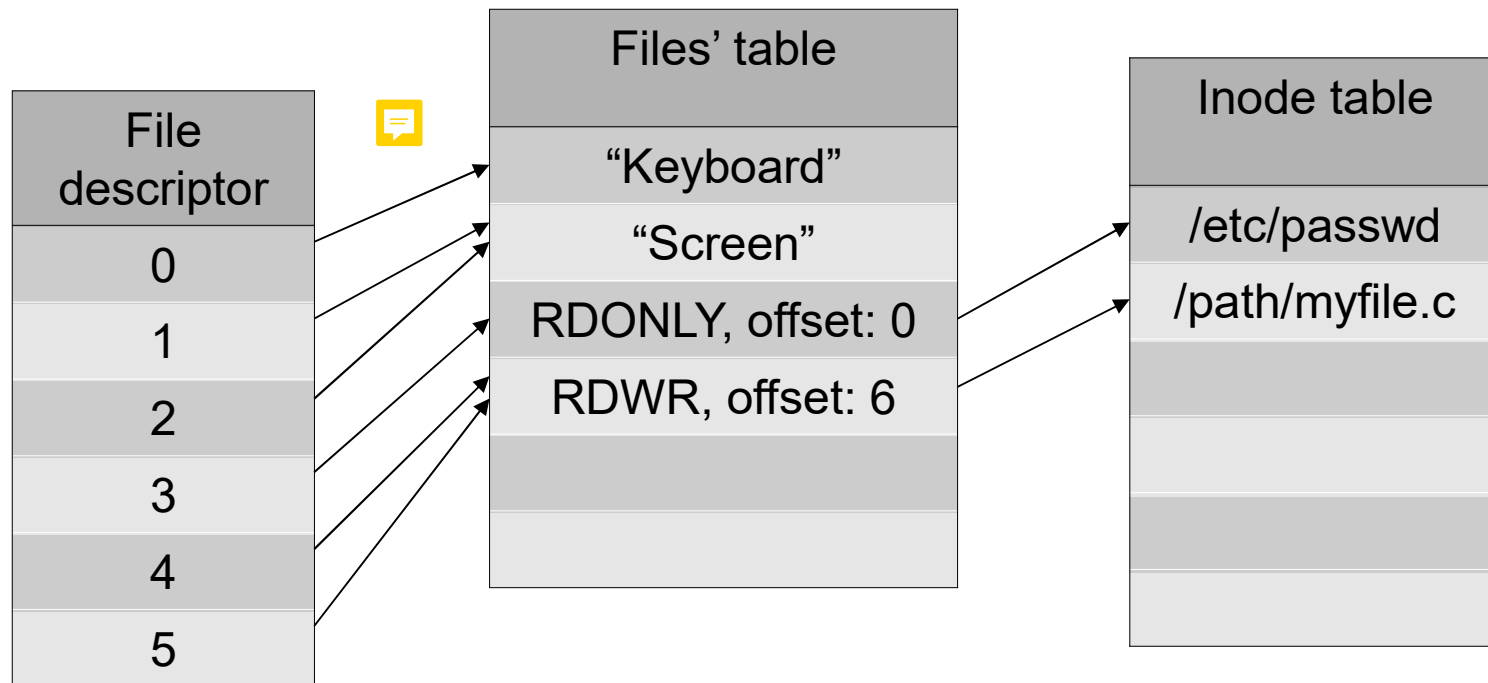
EXEMPLE

- Envoi d'un entier depuis le processus père vers son fils

GESTION DES FILE DESCRIPTORS

- Gestion des file descriptors (fd) utilise trois tables :
 - Table des fd : nb entiers positifs, une table par processus
 - Table des fichiers : permissions et offset de chaque fd utilisé, table globale
 - Inode table : “path” des fichiers, table globale

GESTION DES FILE DESCRIPTORS



GESTION DES FILE DESCRIPTORS

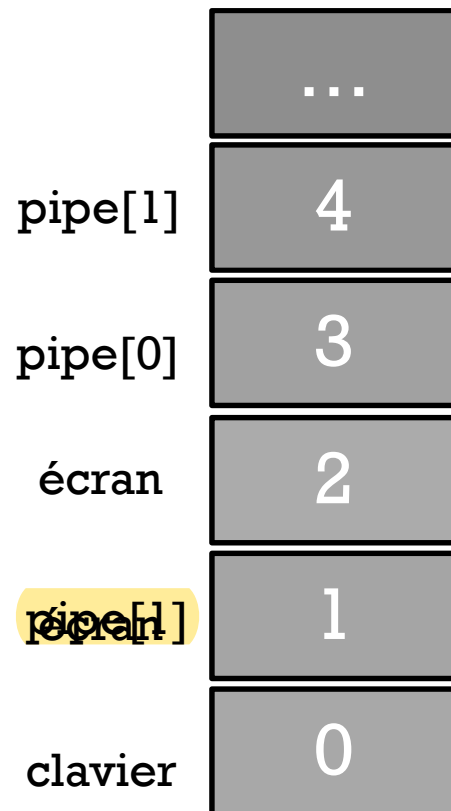
- Appel système `open` associe un fd vers une ressource (table des fichiers)
- Plusieurs fd peuvent pointer la même ressource dans la table des fichiers (mécanisme de duplication – appel système `dup/dup2`)
- Appel système `close` libère le fd. Si c'est la dernière référence vers une ressource, celle-ci est libérée



REDIRECTION

- `ls -l > sortie.txt`
- Exemple de redirections :
 - Redirection de l'entrée standard dans un fichier
 - Redirection de la sortie standard dans un pipe
 - ...

REDIRECTION




```
close (1) ;  
dup (pipe [1] ) ;
```

OU

```
dup2 (pipe [1] , 1)
```

REDIRECTION

- Ce qu'on appelle redirection est en fait de la duplication de fd
- `dup` : le plus petit fd disponible pointe la même ressource que le fd en paramètre – il le duplique 
- `dup2` : un fd spécifié est fermé si nécessaire, puis pointe vers la même ressource qu'un autre fd spécifié

DUP / DUP2

```
int dup(int fd) ;
```

- **dup()** makes the lowest available fd be the copy of *fd*

```
int dup2(int oldfd, int newfd) ;
```

- **dup2()** makes *newfd* be the copy of *oldfd*, closing *newfd* first if necessary