



جامعة سيدي محمد بن عبد الله  
كلية العلوم ظهر المهرارز

جامعة سيدي محمد بن عبد الله  
كلية العلوم ظهر المهرارز

Université Sidi Mohamed Ben Abdellah  
Faculté des Sciences Dhar Mahraz



## Compte Rendu De Travaux Pratique

### System d'Exploitation

### Travaille Pratique 2



SMI – S4

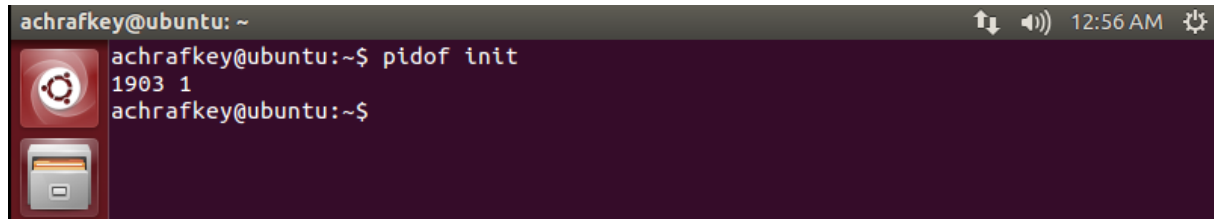
Réaliser par :

Prénom :	CNE :	Nom :
Tazi	1513755449	Achraf
Nadrani	1311778906	Oussama

### **Travail 1** : Manipulation des processus :

## 1. PID du processus INIT ?

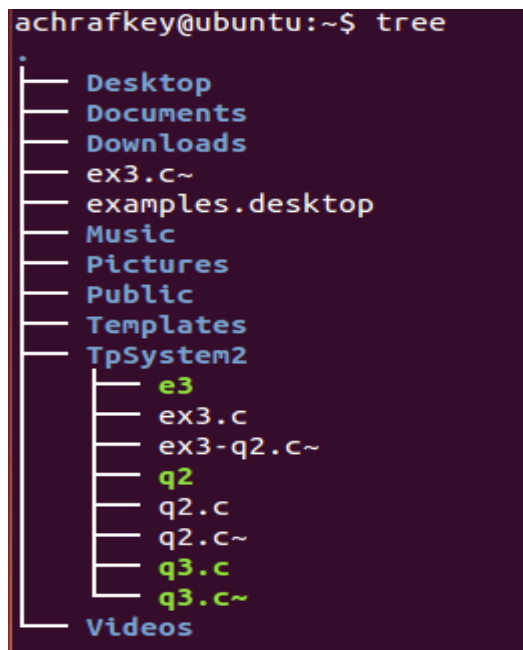
Utilisant la commande « pidof init », on trouve que le pid du processus INIT est 1. C'est le père de tous les processus.



```
achrafkey@ubuntu: ~  
achrafkey@ubuntu:~$ pidof init  
1903 1  
achrafkey@ubuntu:~$
```

## 2. Comment afficher tous les processus sous la forme d'une arborescence ?

La commande unix « tree » affiche sur la sortie standard tous les processus existants sous forme d'un arbre.



```
achrafkey@ubuntu:~$ tree  
.  
├── Desktop  
├── Documents  
├── Downloads  
├── ex3.c~  
├── examples.desktop  
├── Music  
├── Pictures  
├── Public  
├── Templates  
├── TpSystem2  
├── e3  
├── ex3.c  
├── ex3-q2.c~  
├── q2  
├── q2.c  
├── q2.c~  
├── q3.c  
├── q3.c~  
└── Videos
```

## 3. Pratiquez les principales commandes (ps, pstree, pgrep, kill, top) qui permettent de manipuler ou visualiser des processus.

a) PS :

```

achrafkey@ubuntu:~$ ps
  PID TTY          TIME CMD
 2622 pts/1        00:00:00 bash
 3148 pts/1        00:00:00 ps
achrafkey@ubuntu:~$

```

## b) PSTREE :

```

achrafkey@ubuntu:~$ pstree
init--ModemManager--2*[{ModemManager}]
  |--NetworkManager--dhclient
  |                  |--dnsmasq
  |                  |--3*[{NetworkManager}]
  |--accounts-daemon--2*[{accounts-daemon}]
  |--acpid
  |--anacron--sh--run-parts--apt--sleep
  |--avahi-daemon--avahi-daemon
  |--bluetoothd
  |--colord--2*[{colord}]
  |--cron
  |--cups-browsed
  |--cupsd
  |--dbus-daemon
  |--6*[getty]
  |--gnome-keyring-d--6*[{gnome-keyring-d}]
  |--kerneloops
  |--lightdm--Xorg
  |          |--lightdm--init--at-spi-bus-laun--dbus-daemon
  |          |                  |--3*[{at-spi-bus-laun}]
  |          |                  |--at-spi2-registr--{at-spi2-registr}
  |          |                  |--bamfdaemon--3*[{bamfdaemon}]
  |          |                  |--dbus-daemon
  |          |                  |--dconf-service--2*[{dconf-service}]
  |          |                  |--evolution-calen--4*[{evolution-calen}]
  |          |                  |--evolution-sourc--2*[{evolution-sourc}]

```

## c)top :

```
achrafkey@ubuntu:~$ top

top - 01:02:33 up 13 min,  2 users,  load average: 1.34, 1.32, 1.10
Tasks: 163 total,   3 running, 160 sleeping,   0 stopped,   0 zombie
%Cpu(s):  7.2 us, 26.2 sy,  0.0 ni,  0.0 id, 65.5 wa,  0.0 hi,  1.0 si,  0.0 st
KiB Mem:  1027200 total,  768464 used,  258736 free,   52592 buffers
KiB Swap: 1046524 total,   7332 used, 1039192 free.  367848 cached Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
 3519 root        20   0    2324    1784   1504 D  15.9   0.2    0:00.48 updatedb.mlo+
 1087 root        20   0   147792  34296  18476 S   3.6   3.3    0:45.76 Xorg
   141 root        20   0         0         0        0 S   0.7   0.0    0:01.28 jbd2/sda1-8
    3 root        20   0         0         0        0 R   0.3   0.0    0:02.14 ksoftirqd/0
    7 root        20   0         0         0        0 R   0.3   0.0    0:06.33 rcu_sched
 2220 achrafk+    20   0   305548  88988  57948 S   0.3   8.7    0:22.56 compiz
 2610 achrafk+    20   0   127848  29408  23056 S   0.3   2.9    0:06.01 gnome-termin+
 3478 achrafk+    20   0    5436    2696    2300 R   0.3   0.3    0:00.05 top
    1 root        20   0    4468    3412   2524 S   0.0   0.3    0:05.25 init
    2 root        20   0         0         0        0 S   0.0   0.0    0:00.01 kthreadd
    4 root        20   0         0         0        0 S   0.0   0.0    0:00.00 kworker/0:0
    5 root         0 -20         0         0        0 S   0.0   0.0    0:00.00 kworker/0:0H
    6 root        20   0         0         0        0 S   0.0   0.0    0:00.44 kworker/u16:0
    8 root        20   0         0         0        0 S   0.0   0.0    0:00.00 rcu_bh
    9 root        rt    0         0         0        0 S   0.0   0.0    0:00.00 migration/0
   10 root        rt    0         0         0        0 S   0.0   0.0    0:03.79 watchdog/0
   11 root         0 -20         0         0        0 S   0.0   0.0    0:00.00 khelper
   12 root        20   0         0         0        0 S   0.0   0.0    0:00.01 kdevtmpfs
   13 root         0 -20         0         0        0 S   0.0   0.0    0:00.00 netns
   14 root        20   0         0         0        0 S   0.0   0.0    0:00.00 khungtaskd
```

**Travail 2** : Ecrire un programme en C qui permet d'afficher les identifiants d'un processus:

```
#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

void main()
```

```
{  
printf("Le pid de pere est %d \n", getpid());  
printf("Le pid de fils est %d \n", getpid());  
}
```

```
achrafkey@ubuntu:~$ cd TpSystem2/  
achrafkey@ubuntu:~/TpSystem2$ gcc exercice2-tp2.c -o ex2  
achrafkey@ubuntu:~/TpSystem2$ ./ex2  
Le pid de pere est 2908  
Le pid de fils est 2991  
achrafkey@ubuntu:~/TpSystem2$
```

**Travail 3** : Ecrire un programme en C qui permet de créer un processus fils en utilisant la fonction `fork`.

```

#include<stdlib.h>
#include<unistd.h>
void main(){
int pid;

    printf("\n[processus %d] avant le fork\n", getpid());
    printf("APPEL A FORK...\n\n");

    // Création d'un processus enfant
    pid = fork();

    switch (pid) {

        // En cas d'erreur lors du fork
        case -1 : perror("Creation enfant impossible");
        exit(EXIT_FAILURE);
        // Pour le processus enfant
        case 0 : printf("[Processus enfant %d] : Valeur renvoyée par fork =%d\n",getpid(),pid);
        break;
        // Pour le processus père
        default : printf("[Processus père %d] : Valeur renvoyée par fork =%d\n", getpid(),pid);
        }
        exit(EXIT_SUCCESS);
    }
}

```

Activer Windows  
Accédez aux paramètres po

```

achrafkey@ubuntu:~/TpSystem2$ gcc trav3.c -o ex3
achrafkey@ubuntu:~/TpSystem2$ ./ex3

```

```

[processus 3407] avant le fork
APPEL A FORK...

```

```

[Processus père 3407] : Valeur renvoyée par fork =3408

```

```

achrafkey@ubuntu:~/TpSystem2$ [Processus enfant 3408] : Valeur renvoyée par fork
=0

```

Activer Windows  
Accédez aux paramètres p

**Travail 4** : En utilisant la fonction « fork », écrire un programme en C dont le père permet la création d'un

fil. Intégrer dans ce programme la fonction « wait » pour que le père attende la terminaison de son fils.

```
travail4.c x
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
void main(){
int pid,j;

printf("\n[processus %d] avant le fork\n", getpid());
printf("APPEL A FORK...\n\n");

pid = fork();

// Processus enfant
If (pid==0)
{
sleep(1);
printf("[Processus enfant %d] \n", getpid());
// Processus pere
} else {
wait(NULL);
printf("[Processus père %d] \n",getpid());
}
exit(EXIT_SUCCESS);
}
```

```
achrafkey@ubuntu:~/TpSystem2$ gcc travail4.c -o ex4
achrafkey@ubuntu:~/TpSystem2$ ./ex4
```

```
[processus 3523] avant le fork
APPEL A FORK...
```

```
[Processus enfant 3524]
[Processus père 3523]
achrafkey@ubuntu:~/TpSystem2$
```

