

# TP 1

## Gestion des processus

Antoine de Roquemaurel et Mathieu Soum  
Semestre 3

### 1 Exercice 1

```
1 drn0073a@r-info-jade-l03:~$
2 Appuyez sur ENTRÉE ou tapez une commande pour continuer
3
4 mardi 20 septembre 2011, 10:06:52 (UTC+0200)
5 Le login est satenske
6 L'IUD est 1000
7 -----
8
9
10 Processus fils 1. son PID est 2900
11 Processus fils 2. son PID est 2901
12 fin anormale fils PID=2901, numéro du signal=11
13 fin normale du PID fils=2900, statut=3
```

Listing 1 – Trace du programme proc1: **proc1.trace**

```
1  /*****
2  *   ASR => Gpr3
3  *****/
4  *   Login : drn0073a
5  *
6  *****/
7  *   Groupe: B
8  *
9  *****/
10 *   Nom-prénom : de ROQUEMAUREL Antoine
11 *   Nom-prénom : SOUM Mathieu
12 *
13 *****/
14 *   TP n 1
15 *****/
16 *   Nom du fichier : proc1.c
17 *****/
18
19 #include <stdlib.h>
20 #include <stdio.h>
21 #include <sys/types.h>
22 #include <unistd.h>
23
24 #define OK 0
25 #define PAS_OK -1
26
```

```
27 void traitement_fils1();
28 void traitement_fils2();
29
30 int main(int argc, char *argv[]) {
31     pid_t idFils1, idFils2, pidPere, idProcRetour;
32     int rapport, numSig, statut;
33
34     pidPere = getpid();
35
36     system("date");
37     system("echo Le login est $USER\n");
38     printf("L'IUD est %d\n", getuid());
39     printf("-----\n\n\n");
40
41     idFils1 = fork(); // On créer le premier fils
42     if (pidPere == getpid()) // Si c'est le père, on crée un deuxième fils!
43         idFils2 = fork();
44
45     switch ( idFils1 ) {
46         case PAS_OK :
47             perror("Erreur de lancement du processus fils1");
48             exit(1);
49         case OK :
50             traitement_fils1();
51             break;
52     }
53     switch ( idFils2 ) {
54         case PAS_OK:
55             perror("Erreur de lancement du processus fils2");
56             exit(1);
57         case OK :
58             traitement_fils2();
59             break;
60     }
61
62     idProcRetour = wait(&rapport);
63     // tant que le processus n'est pas arrêté
64     while (idProcRetour != -1) {
65         numSig = rapport & 0x7F;
66
67         switch (numSig) {
68             case 0 :
69                 statut = (rapport >> 8) & 0xFF;
70                 printf("fin normale du PID fils=%d, statut=%d\n", idProcRetour,
71                     statut);
72                 break;
73             default:
74                 printf("fin anormale fils PID=%d, numéro du signal=%d\n",
75                     idProcRetour, numSig);
76                 break;
77         }
78         idProcRetour = wait(&rapport);
79     }
80     return 0;
81 }
82 // processus fils 1
```

```

83 void traitement_fils1() {
84     printf("Processus fils 1. son PID est %d\n", getpid());
85     exit(3);
86 }
87
88 // processus fils 2
89 void traitement_fils2() {
90     int* ptr = NULL;
91     printf("Processus fils 2. son PID est %d\n", getpid());
92
93     *ptr = 42;
94     exit(OK);
95 }

```

Listing 2 – Code source de **proc1.c**

## 2 Exercice 2

```

1 drn0073a@r-info-jade-103:~$
2 Appuyez sur ENTRÉE ou tapez une commande pour continuer
3 mardi 20 septembre 2011, 10:17:44 (UTC+0200)
4 Le login est satenske
5 L'IUD est 1000
6 -----
7
8
9 Ici le processus fils 2 de PID 3458 !
10 Ici le processus fils 1 de PID 3457 !
11 La chaine passée en paramètre est : Cacamou
12 Le paramètre en décimal est 42(10) et en hexadécimal 2a(16) !
13 fin normale du PID fils=3457, statut=0
14 fin normale du PID fils=3458, statut=0

```

Listing 3 – Trace du programme proc2: **proc2.trace**

```

1  /*****
2   *   ASR => Gpr3                                     *
3   *****/
4   *   Login : drn0073a                                 *
5   *                                               *
6   *****/
7   *   Groupe: B                                       *
8   *                                               *
9   *****/
10  *   Nom-prénom : de ROQUEMAUREL Antoine             *
11  *   Nom-prénom : SOUM Mathieu                       *
12  *                                               *
13  *                                               *
14  *****/
15  *   TP n: 1                                           *
16  *****/
17  *   Nom du fichier : proc2.c                         *
18  *****/
19
20 #include <stdlib.h>
21 #include <stdio.h>
22 #include <sys/types.h>

```

```
23 #include <unistd.h>
24
25 #define OK      0
26 #define PAS_OK -1
27
28 int main(int argc, char *argv[]) {
29     pid_t idFils1, idFils2, pidPere, idProcRetour;
30     int rapport, numSig, statut, err1, err2;
31     char* argP2fils2[] = { "main", "42", NULL };
32
33     pidPere = getpid();
34
35     system("date");
36     system("echo Le login est $USER et l UID $UID\n");
37     system("echo -----\n");
38     printf("\n\n");
39
40     idFils1 = fork(); // On créer le premier fils
41     if (pidPere == getpid()) { // Si c'est le père, on créé un deuxième fils!
42         idFils2 = fork();
43     }
44
45     switch ( idFils1 ) {
46         case PAS_OK :
47             perror("Erreur de lancement du processus fils1");
48             exit(1);
49             break;
50         case OK :
51             err1 = execl("p2fils1", "main", "une super chaine de caracteres", NULL
52             );
53             if (err1 == -1) {
54                 perror("Erreur au niveau du execl");exit(2);
55             }
56             break;
57     }
58     switch ( idFils2 ) {
59         case PAS_OK:
60             perror("Erreur de lancement du processus fils2");
61             exit(1);
62             break;
63         case OK :
64             err2 = execvp("p2fils2", argP2fils2);
65             if (err2 == -1) {
66                 perror("Erreur au niveau du execvp");exit(2);
67             }
68             break;
69     }
70
71     idProcRetour = wait(&rapport);
72     // tant que le processus n'est pas arrêté
73     while (idProcRetour != -1) {
74         numSig = rapport & 0x7F;
75
76         switch (numSig) {
77             case 0 :
78                 statut = (rapport >> 8) & 0xFF;
79                 printf("fin normale du PID fils=%d, statut=%d\n", idProcRetour,
80                     statut);
```

```

79         break;
80     default:
81         printf("fin anormale fils PID=%d, numéro du signal=%d\n",
82             idProcRetour, numSig);
83         break;
84     }
85     idProcRetour = wait(&rapport);
86 }
87 return 0;
88 }

```

Listing 4 – Code source de **proc2.c**

```

1  /*****
2  *   ASR => Gpr3
3  *****/
4  *   Login : drn0073a
5  *
6  *****/
7  *   Groupe: B
8  *
9  *****/
10 *   Nom-prénom : de ROQUEMAUREL Antoine
11 *   Nom-prénom : SOUM Mathieu
12 *
13 *
14 *****/
15 *   TP n: 1
16 *****/
17 *   Nom du fichier : p2fils1.c
18 *****/
19
20 #include <stdlib.h>
21 #include <stdio.h>
22 #include <sys/types.h>
23 #include <unistd.h>
24
25 int main(int argc, char** argv) {
26     printf("Ici le processus fils 1 de PID %d !\n", getpid());
27     printf("La chaine passée en paramètre est : %s\n", argv[1]);
28     exit(0);
29
30     return 0;
31 }

```

Listing 5 – Code source de **p2fils1.c**

```
1  /*****
2  *   ASR => Gpr3
3  *****/
4  *   Login : drn0073a
5  *
6  *****/
7  *   Groupe: B
8  *
9  *****/
10 *   Nom-prénom : de ROQUEMAUREL Antoine
11 *   Nom-prénom : SOUM Mathieu
12 *
13 *
14 *****/
15 *   TP n: 1
16 *****/
17 *   Nom du fichier : p2fils2.c
18 *****/
19
20 #include <stdlib.h>
21 #include <stdio.h>
22 #include <sys/types.h>
23 #include <unistd.h>
24
25 int main(int argc, char** argv) {
26     printf("Ici le processus fils 2 de PID %d !\n", getpid());
27     printf("Le paramètre en décimal est %d(10) et en hexadécimal %x(16) !\n",
28           atoi(argv[1]), atoi(argv[1]));
29     exit(0);
30     return 0;
31 }
```

Listing 6 – Code source de **p2fils1.c**