Univérsité Cheikh Anta Diop

Faculté des Sciences et Techniques
Département Mathématique et Informatique
Année Universitaire 2021-2022
MS2E
Travaux Pratiques de Système d'Exploitation

Atelier 02 : *Programmation des Signaux Systèmes UNIX/Linux*

Dr Mandicou BA

Exercice 1 : Pour le programme de la figure ci-dessous, après l'avoir analysé, dites qu'il réalise et proposer une méthodologie pour le tester via un shell. Proposez une amélioration pour prendre en compte les signaux SIGGUSR1, SIGUSR2. Proposer un traitement prenant en compte le signal SIGKILL. Que remarquez-vous ?

```c
/**
 * @author Dr Mandicou BA
 * @version 16/01/2017
 */
#include <stdlib.h>
#include <signal.h>
#include <errno.h>
#include <unistd.h>
#include <stdio.h>

#define DELAI 1

void traitement(int numero_signal)
{
    printf("\n Reception du Signal %d => ",numero_signal);
    switch(numero_signal) {

        case SIGTSTP :
            printf("Je m'endors....\n");
            kill(getpid(), SIGSTOP);
            printf ("Je me réveille !\n");
            signal (SIGTSTP, traitement );
            break ;
        case SIGINT :

        case SIGTERM:
            printf("Fin du programme.\n");
            exit(EXIT_SUCCESS);
            break;
        }
}

int main(void) {


    printf("Mon PID est %d => ",getpid());

    signal(SIGTSTP, traitement);
    signal(SIGINT , traitement );
    signal(SIGTERM, traitement);

    printf("Attente de reception d'un signal \n");

    while (1) {
        sleep(DELAI);
        printf(".");
        fflush (stdout);

    }
    printf("fin\n");
    exit (EXIT_SUCCESS);

}
```

Exercice 2 : L'appel système pause

Écrire un programme système qui montre l'utilisation de l'appel système pause. Il crée un fils qui place un gestionnaire associe au signal *SIGUSR1* puis qui se met en pause. Le père quant à lui, envoie un signal *SIGUSR1* après un sleep de 1s puis attend la fin d'exécution de son fils.

Exercice 3 : L'appel système alarme

Écrire programme système qui montre le fonctionnement de l'appel système alarme. Une alarme de 5s est mise en place et le programme se met en attente d'un signal quelconque. Dès que le signal *SIGARLM* est reçu, le programme peut se terminer avec le siganl SIGINT.

Exercice 4 :

1. Proposer un programme qui n'accepte de tuer le processus correspondant par un CTRL-C qu'après vérification par un mot de passe.
2. Écrire un programme qui tue le processus dont le PID est passé en argument seulement si l'utilisateur confirme. Le programme doit s'arrêter si le PID n'est pas fournit
3. Modifier le programme précédant de sorte qu'il prenne deux paramètres que sont le PID d'un processus et un numéro de signal N à lui envoyer. Après vérification, le numéro N est envoyé au processus d'identifiant PID. Le programme doit s'arrêter si les bons arguments ne lui sont pas fournis. Utiliser le programme proposé pour tester l'exercice 1.

Exercice 5 : Envoi de signaux entre un Client et Serveur

L'objectif de cet exercice est d'écrire un programme système qui illustre l'envoi de signaux entre un Client et un Serveur.

1. **Coté Serveur** :

   Le programme **signauxServeur** à écrire doit illustrer le fonctionnement de l'appel système signal qui permet de spécifier un gestionnaire a des signaux. Le programme attend de recevoir un signal **SIGUSR1** puis un signal **SIGUSR2**. Pour cela, il faut utiliser le programme **signauxClient** (à écrire à la question 2).

   (a) Déclarer une variable permettant de marquer les signaux reçus par le Serveur

   (b) Écrire est le gestionnaire (*void handler(int signum)*, signum est le numéro du signal reçu) qui est associe aux signaux *SIGUSR1* et *SIGUSR2*

   (c) Positionner le gestionnaire pour SIGUSR1 (dans la méthode main)

   (d) Positionner le gestionnaire pour SIGUSR2 (dans la méthode main)

   (e) Mettre en attente jusqu'à recevoir au moins un signal *SIGUSR1* et *SIGUSR2*

2. **Coté Client** :

   Le programme **signauxClient** à écrire doit illustrer comment envoyer des signaux a l'aide de l'appel système *kill*. Le PID du programme a qui envoyer les signaux est spécifie en argument (ici, le but sera d'exécuter le programme signauxServeur et de donner son PID au programme signauxClient). Un signal SIGUSR1 sera d'abord envoyé, suivi de SIGUSR2.

   (a) Gérer d'abord la récupération des arguments

   (b) Écrire le code permettant l'envoi du premier signal

   (c) Écrire le code permettant l'envoi du second signal

   (d) Se tuer avec le signal SIGKILL

Après avoir écrit le Client et Serveur, tester le bon fonctionnement du programme.