Signaux

Mise en place d'une alarme

Cet article présente comment mettre en place une alarme à l'aide de l'appel système `alarm`.

1 Le programme

Le but du programme est de mettre en place une alarme de 5 secondes. Il attend ensuite soit la fin de l'alarme, soit la réception d'un `SIGINT`.

Nous utilisons une variable globale `stop` initialisée à 0 et un gestionnaire de signal. Ce dernier est appelé à la réception du signal de l'alarme ou du SIGINT. Il place la variable globale à 1. À noter que l'appel `alarm(0)` permet de désactiver l'alarme.

```
int stop = 0;
void handler(int signum) {
  stop = 1;
  if(signum == SIGALRM)
    printf("Signal␣SIGALRM␣reçu.\n");
  else {
    alarm(0);
    printf("Signal␣d'arrêt␣reçu.\n");
  }
}
```

Dans le `main`, nous mettons en place le gestionnaire sur les deux signaux à l'aide de l'appel système `sigaction`.

```
struct sigaction action;

action.sa_handler = handler;
sigemptyset(&action.sa_mask);
action.sa_flags = 0;
if(sigaction(SIGALRM, &action, NULL) == -1) {
  perror("Erreur␣lors␣du␣positionnement␣");
  exit(EXIT_FAILURE);
}
if(sigaction(SIGINT, &action, NULL) == -1) {
  perror("Erreur␣lors␣du␣positionnement␣");
  exit(EXIT_FAILURE);
}
```

Le programme planifie ensuite l'alarme à l'aide de `alarm`.

```
alarm(5);
```

La boucle principale se met en attente de la réception de l'un ou l'autre des signaux.

```
printf("J'attends la fin de l'alarme ou un SIGINT...\n");
while(stop == 0) {
  pause();
}
```

2 Compilation et exécution

Le `makefile` fourni permet de compiler le programme précédent. Tapez la commande suivante :

```
make
```

Pour tester le programme, exécutez la commande suivante :

```
./alarme
```

Soit vous attendez 5 secondes pour que le programme s'arrête, soit vous pressez sur CRTL+C pour envoyer le signal SIGINT et stopper le programme.