İşletim Sistemleri Uygulama III

Unix'de Süreçler
Fork ve Exec

Fork Sistem Çağrısı

- int fork();
- fork() bir defa çağrılır …
- Ancak iki defa geri döner!!
  - Ana süreçte bir defa
  - Çocuk süreçte bir defa
- Ana ile çocuğu nasıl ayırt edebiliriz??
  - Çocukta döndürülen değer = 0
  - Anada döndürülen değer = çocuğun süreç tanımlayıcısı

Örnek Program

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>

int main (void)
{
        int f;

        f=fork();
        if (f==-1) {
                printf("Hata \n");
                exit(1);
        } else if (f==0) /* cocuk */
        {
                printf("    Cocuk: Proses no: %d \n", getpid());
                sleep(1);
                printf("    Cocuk: Annemin no: %d \n", getppid());
                exit(0);
        } else {
                printf("Anne: Proses no: %d \n", getpid());
                printf("Anne: Cocugumun no: %d \n", f);
                printf("Anne: Annemin no: %d \n", getppid());
                wait(NULL);
                printf("Anne: Sonlaniyorum...\n");
                exit(0);
        }
        return (0);
}
```

Örnek Program Çıktısı

- [root@koknar root]$ gcc fork.c
- [root@koknar root]$ ./a.out
- Cocuk: Proses no: 446
- Anne: Proses no: 445
- Anne: Cocugumun no: 446
- Anne: Annemin no: 402
- Cocuk: Annemin no: 445
- Anne: Sonlaniyorum...
- [root@koknar root]$

Exec Sistem Çağrısı

```
int execlp(char * filename, char
  * arg0, char * arg1,…,char
  *argn,(char*)0);
```

- Unix'in bir süreç içinden yeni bir program çalıştırmasının tek yoludur.
- PID değişmez
- Yeni bir süreç değildir!

Exec Sistem Çağrısı

- Exec komutunun çağrılması için 6 yolu vardır.
  - execl, execv, execle, execve, execlp, execvp
    - int execl (const char path, const char arg0, .../*, const char argn, (char *)0*/);
    - int execv (const char path, char const argv);
    - int execle (const char path, const char arg0, .../*, const char argn, (char *)0, const char envp[]*/);
    - int execve (const char path, char const argv, char const envp);
    - int execlp (const char file, const char arg0, .../*, const char argn, (char *)0*/);
    - int execvp (const char ***file***, char const ***argv***);

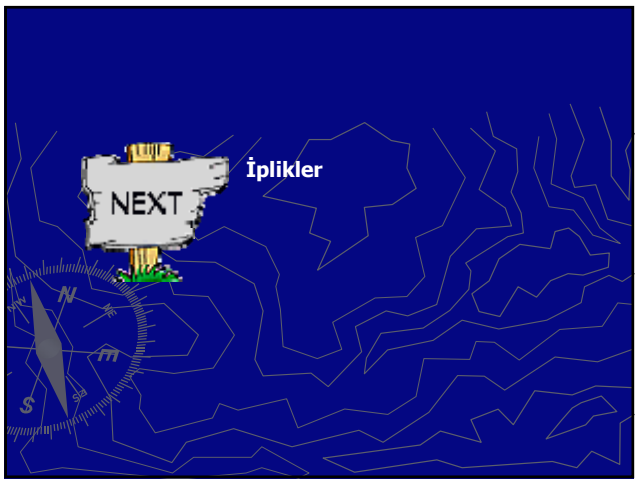
Örnek Program

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int f;

int main (void)
{
        printf("\nBirinci program calisiyor: PID=%d \n",getpid());
        f=fork();
        if (f==0)   /*cocuk*/
        {
                printf("\nBen cocuk ... PID= %d\n", getpid());
                execlp("./uyg2_2","./uyg2_2","a","b",(char *)NULL);
        } else  /* anne */
                exit(0);
        return(0);
}
```

Örnek Program (Devamı)

```
#include <unistd.h>
#include <stdio.h>

int main (int argc, char *argv[])
{
        printf("\nIkinci program calismaya basladi... PID= %d\n",getpid());
        printf("Programin adi:%s\n",argv[0]);
        printf("Programin 1. parametresi:%s\n",argv[1]);
        printf("Programin 2. parametresi:%s\n",argv[2]);
        return(0);
}
```

```
[root@koknar root]$ gcc ex.c
[root@koknar root]$ gcc uyg2_2.c -o uyg2_2
[root@koknar root]$ ls
a.out  ex.c  uyg2_2  uyg2_2.c
[root@koknar root]$ ./a.out
Birinci program calisiyor: PID=477
Ben cocuk... PID= 478
Ikinci program calismaya basladi... PID= 478
Programin adi:./uyg2_2
Programin 1. parametresi:a
Programin 2. parametresi:b
[root@koknar root]$
```


İplikler
NEXT