Fundamentos de los Sistemas Operativos (FSO)

Departamento de Informática de Sistemas y Computadoras (DISCA)
Universitat Politècnica de València

Part 1: Introduction

Unit 2

System Call Concept

fSO

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DISCA

Questions

- Which of the following instructions should be privileged?
  - Set value of timer   Privileged
  - Read the clock.   User
  - Clear memory.   Privileged
  - Issue a trap instruction. User
  - Turn off interrupts. Privileged
  - Modify entries in device-status table. Privileged
  - Switch from user to kernel mode. User
  - Access I/O device. Privileged

Questions

- Some CPUs provide for more than two modes of operation. What are two possible uses of these multiple modes?
  - The use of modes of operation in related to the security and protection. More than two modes allow to manage some behaviour with a more narrow grain policy. For instance to deal with specific devices.
  - Use Case 1: kernel/user
  - Use Case 2: hypervisor/kernel/user
- Timers could be used to compute the current time. Provide a short description of how this could be accomplished.
  - The OS can use a hardware timer to program a periodic interrupt (e.g 100ms) that informs to the OS that the time is running. A variable (current_time) can be updated when the interrupt occurs. A precise clock read implies to compute the time with the current_time + current vale of the timer.
- How many hardware timers are needed to provide timers to processes.
  - The hardware provides not too much timers (2-5). One is use for the clock (see previous question). Another can be use for processes. The OS uses only one timer and uses a data structure (heap or linked list) with the timer programmed by the processes. Only the top is set in the hardware timer (earliest timer is programmed). When it finish, the OS informs to the process that its timer expired and programs the next timer in the list.

Exercise 1:

- In the following actions list, indicate from each one if it is shell code or OS code the one that performs it. Mark one, none or both options with a cross:

OS	Shell	
	X	Reading the command line and parse it
X		Programming a device controller
X*		Providing a system calls interface
X		Selecting a process to schedule in the CPU
X**		Performing a system call
	X	Providing a confortable user interface

* The interface is provided by the OS through the libC library that is executed by each process (application) that requests a system call.
** The system call is requested by the processes (applications) and executed by the OS.

Exercise 2

- Which state (new, ready, execution, suspended, finished) corresponds to every one os the following processes:

Process		State
P1	The CPU is executing intructions belowing to P1	Running
P2	P2 has ask for a disk access, but the disk is busy serving to process P3	Suspended
P3	P3 is doing disk access	Suspended
P4	P4 belongs to a user that has finished all his/her jobs in a terminal and is logging out the session	Finished
P5	P5 has a process identifier assigned and only its control table are already created	New
P6	P6 has all its SO control structures and its memory image stored in main memory	Ready or execution

Exercise 3

- In the following command line:

```
$ cat f1 f2 f3 | grep start | wc -l >tracd
```

Indicate:

- – How many processes will be created during its execution in a UNIX system
- – What files has associated every command

3 processes:

cat : <u>file input </u>f1, f2 and f3; <u>file output </u>=> grep

grep: <u>file input </u>cat output; <u>file output </u>=> wc

Exercise 4

- Given the following code:

```c
#include <stdio.h>
#include <sys/types.h>

int main(void)
{
   pid_t pid;
   int i;

   for (i=0; i<2; i++)
      pid=fork();
    sleep(10);
   return 0;
}
```

How many processes will be created along its execution?

3

I = 0 fork; i = 1 fork

i = 1 fork;