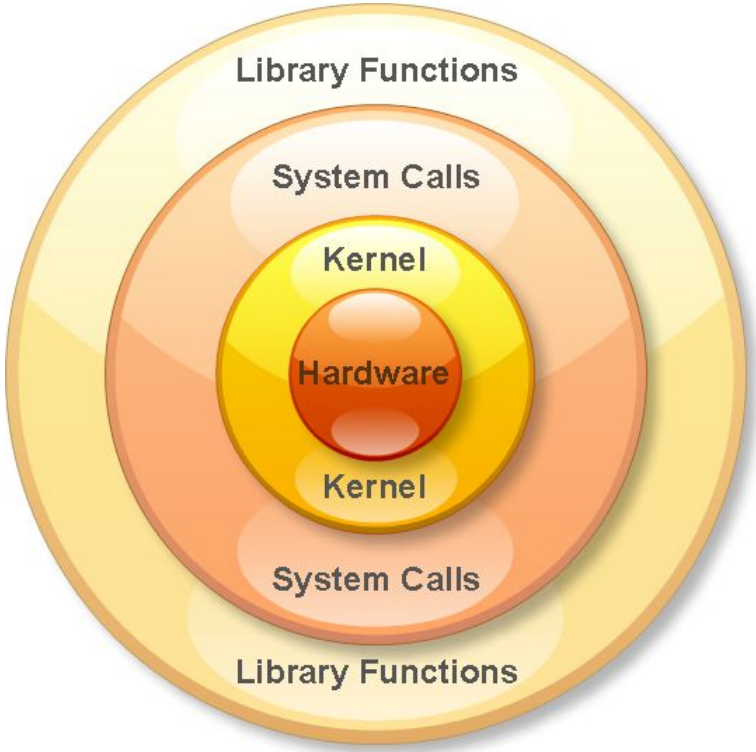
System Calls

(Φροντιστήριο για την 3η σειρά)

Dimitris Deyannis
deyannis@csd.uoc.gr

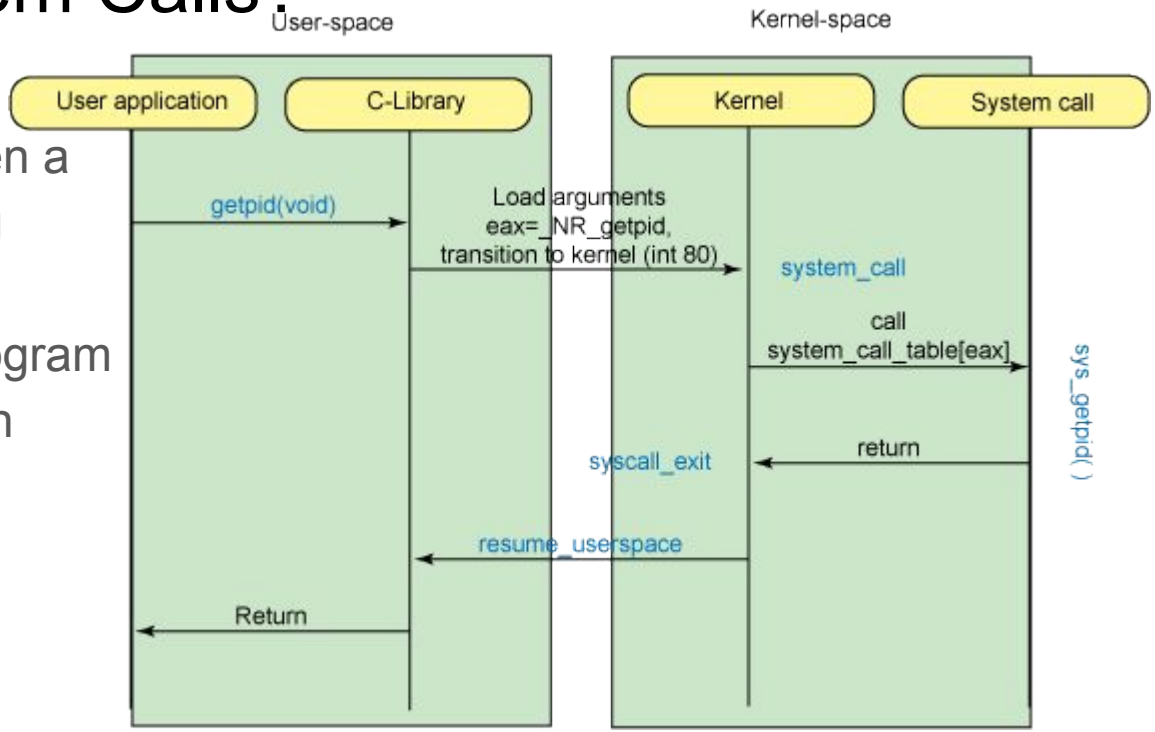
What is a System Call?

- The system call is the fundamental interface between an application and the Linux kernel



Why we need System Calls?

- System calls provide an essential interface between a process and the operating system
- A system call is how a program requests a service from an operating system's kernel



What can System Calls do?

- File management
  - create, open, delete..
- Process control
  - exec, kill, wait...
- Device management
  - request, release...
- Information maintenance
  - get time, set time...
- Communication
  - sockets, send, receive...

How do we use System Calls?

- sys/syscall.h is a small library that implements
  long syscall(long number, …);
- This function invokes the system call that corresponds to the “number” while “...” corresponds to the rest of the arguments

Implementing a new System Call

1. Define a system call number
2. Define a function pointer
3. Define a function
4. Implement the system call

Using Qemu

- Load the image and start the guest OS

  $ qemu-system-i386 -hda hy345-linux.img

- Load the image and start the guest OS with the new kernel

  $ qemu-system-i386 -hda hy345-linux.img -append "root=/dev/hda" -kernel linux-2.6.38.1/arch/x86/boot/bzImage

Define a System Call number

- Every system call has an invocation number
- Edit: linux-2.6.38.1/arch/x86/include/asm/unistd_32.h
  - Define the new system call number at the bottom of the list
    - e.g. #define __NR_dummy_sys 341
  - Update the number of system calls
    - #define NR_syscalls 342

Define a function pointer

- The Kernel needs to have a function pointer pointing to the new system call
- Edit: linux-2.6.38.1/arch/x86/kernel/syscall_table_32.S
- Define the function pointer at the bottom of the list
  - e.g. .long sys_dummy_sys /* 341 */

Define a function

- We have to define the function signature in syscalls.h file
- Edit: linux-2.6.38.1/include/asm-generic/syscalls.h
- At the bottom of the file add:

```
#ifndef dummy_sys
        asmlinkage long sys_dummy_sys(int arg0);
#endif
```

Implement the System Call part 1

- Touch and edit: linux-2.6.38.1/kernel/dummy_sys.c as such:

```
#include <linux/kernel.h>
#include <linux/syscalls.h>
#include <asm/uaccess.h>

asmlinkage long sys_dummy_sys(int arg0)
{
	printk("Called system call dummy_sys with argument: %d\n", arg0);
	return ((long)arg0 * 2);
}
```

Implement the System Call part 2

- Edit: linux-source-2.6.38.1/kernel/Makefile
- Add: obj-y += dummy_sys.o

- Now you are ready to compile the Kernel with your new system call!

Compile the Linux Kernel

$ cp ~hy345/qemu-linux/linux-2.6.38.1.tar.bz2 /spare/[username]/
$ tar -jxvf linux-2.6.38.1.tar.bz2
$ cd linux-2.6.38.1

Edit kernel source code to implement the new system calls

$ cp ~hy345/qemu-linux/.config

Edit .config, find CONFIG_LOCALVERSION="-hy345", and append to the kernel's version name your username and a revision number

$ make ARCH=i386 bzImage

Periodic processes

- A periodic process "i" has a period p_i and a computation time c_i
- Every p_i milliseconds the process needs to run for c_i milliseconds.
- The process may run at the beginning of the period i, at the end of the period (i.e. time p_i-c_i) or anywhere in between

Periodic processes

- Once the counting of the period starts, we would like the process to receive c_i milliseconds of CPU time (neither more nor less) in each period
- If a process does not receive c_i milliseconds of CPU time in a period it is said to have missed a deadline

Implementation

- For this assignment you have to implement the following system calls
  - set_period_parameters(int pid, unsigned int p_time, unsigned int c_time)
  - get_period_parameters(int pid, struct p_params *p_arguments)
  - get_missed_deadlines(int pid, struct d_params *d_argument)

Implementation

- Add 3 new fields in task_struct
  - unsigned int period_time; // The period duration
  - unsigned int computation_time; // The CPU time
  - unsigned int missed_deadlines; // The missed deadlines
- Implement the p_params and d_params structs
  - struct p_params
    - unsigned int period_time;
    - unsigned int computation_time;
  - struct d_params
    - unsigned int missed_deadlines;