P1 - System Calls

COP4600

Slides by: Ernesto A. Mujica

What is a System Call?

- Usually a function that allows a programmer to access kernel functions through what is essentially just an API.
- Acts as an interface between a process (your program) and the OS.
- Necessary for process creation, memory management, file system writing/reading, IO, networking… etc

Project Structure

- So what are we actually doing? According to the project pdf:
- Create a kernel-wide process log level attribute.
- Create system calls that allow a process to get or set the process log level of the system.
- Create system call that allows a process to add a process log message at a defined log level.
- Create static library functions that allow the system calls to be invoked via a C API.
- Let’s break these down:

Project Structure (cont.)

- First and foremost, we need to establish how we go about adding new system calls to the kernel.
- The assignment page has several useful resources, but the first is the most important. Let's go through it here.
- Now we know how to add and create syscalls, we can explore the more specific aspects of the project.

“Process Log-Level Attribute”

- Some kind of storage structure that can be manipulated by your system calls.
- Exact implementation may vary, but we recommend a global variable in the same file as your function definitions.

System Calls

- For the system calls themselves, you will need three:
- A ‘get’ function to read the current log level.
- A ‘set’ function to set the current log level.
- A ‘message’ or ‘log’ function to actually log the correct message. You will log the messages using an identical or very similar function to the one used in P0.
- Explicit rules and instructions are in the project pdf.

System Calls (cont.)

- The message passed into the log syscall will be in the form of a cstring pointer, that is, a character array. This means it will point to a place in memory, however that introduces a problem…
- Passing userspace pointers into kernelspace will cause the kernel to panic… which means you WILL crash and possibly lose a good deal of progress.
- The solution? copy_from_user()
- Preexisting kernel functions already use this.
- Look how they use it and do the same!

Library

- Along with the syscalls themselves, you will be writing a small library that actually calls the syscalls as well as harness functions.
- Besides the regular library functions that perform the syscalls, we will also calling them ourselves, and using your harness functions to interpret their results. This is to confirm that the syscalls are actually implemented, and you didn't just write everything in your library.

Project PDF

(Go over project pdf, and demo test functionality).

Testing

- Build your kernel using the commands from Ex0. Building the kernel after you install your syscalls WILL take a while. (Can be as fast as 30 mins or as slow as several hours.)
- Take frequent snapshots between kernel builds, and take screenshots of your syscalls or backup their code in case something horribly breaks.
- Test your patch file on a fresh kernel, post Ex0.
- Testing files are provided to you on canvas.
- Look at the assignment page!

Report/Screencast

- Follow the rules for the report we went over in Ex1. Man formatting, 500 words. Mention all syscalls, all the files you changed, all library functions, how you checked for superuser, how you found the executable name and pid, how you tested your patch file and code, etc.
- Show all code in the screencast. This includes all the kernel files, the library code, and demo functionality by running the testing files.

Submission

- The rules for submission are explicitly laid out in the project pdf. Follow them exactly, and ask questions if you have any doubts. Proper formatting is **essential**, for this project and any future assignments as well.
- There will be a second step for submission for this project that **must** be completed for credit on this portion. Your kernel must have the syscalls installed for that portion, so take a snapshot after project completion. We will walk you through that and show you how to submit it at a later date.