Unit 4 Programming HW - Writeup

SER-334 Fall B 2018

John Z. Orr

Providing a short write-up for items found while researching Linux source files and documentation.

<linux/sched/signal.h>

There is a difference between the space used by the kernel siginfo and the basic siginfo. Siginfo is about half of the size, after the padding is removed. As long as the required chinks of the kernel siginfo are maintained, this allows the kernel to move quicker.

void myFunc(void)

This is really the meat and potatoes of the code. Items contained within this function provide for most of the information displayed to the user during run time.

Struct task_struct

This is part of the <linux/sched… header file. This is a commonly used structure typically used to pass details of a specific process to. Returned as an integer value, it can typically be traced to a specific message as to the integers meaning.

Size_t

Instead of using a generic int data type, a size_t is the best way to represent the size of a value or how many times a value has changed. This was used within my for loop to count the number of processes encountered throughout the program.

For_each_process ("command string")

In order to generate the count of processes currently running on the kernel, this is the provided format for a for loop. This is the fastest and most efficient way to populate the required information for each active process.

*Printk (const char *fmt, …)*

Basically the brother of the printf() function in typical C libraries, however, this prints directly to the kernel. The kernel does not recognize the C language libraries.

Int simple_init(void)

This is the simple function created based on the textbook's recommendation. I have tweaked the contents to facilitate the homework information. It is a void function so it does not have any parameters. It uses a printk() and calls the myFunc() created and described above.

void simple_exit(void)

This is a simple function used to indicate that I have removed the module, using the printk() function.

module_init(simple_init)

This initializes the module to run. The parameter is an init function.

module_exit(simple_exit)

This concludes the module from running and closes the connection. The parameter taken is a simple exit function.

Overall, I would say that most of this homework was accomplished through extensive research, A LOT of trial and error, and bouncing ideas off the TAs for assistance. Google helped, when needed, but it was for specific error codes encountered and how to get around them. The two biggest were installing needed software updates in order to read the kernel and modifying the secure boot process on my Ubuntu machine.