Kernel Module For Printing Process Details

Steps: For taking input, we use module_param() function.

Since we take input of process id, we need to get the task_struct for the process. We find process task structure by iterating over all processes and comparing the pid of all process with the given process using the for_each_process() function provided in the kernel. If we find a process with same pid, we store that process task_struct and break for_each loop and set flag to 1. If a process with same pid is not present, we produce message that process not found. Since task_struct stores name and proces id , uid, etc. We determined them from task_struct only. For making Group id into an integer, a kernel function is used. To find the path we first find the process exe file using a function get_task_exe_file() .But this function isn't available in header files directly; therefore, its implementation and dependency are written into our kernel module to use. The function is used from kernel source code 1) link 2) link .exe file path is made human readable by using function dentry_path_raw().The path returned by dentry_path_raw() is the human readable path and therefore printed directly using printk. __kuid_val(task_list->cred->uid) function is used for making user id into integer. pid_vnr(task_pgrp(task_list)) is used for finding pgid of the process, first we find task_struct of parent process by using task_pgrp() and then this structure is passed into pid_vnr() to find pid of the process. Since pgid is same as pid of parent of group the group, therefore we find pid of the parent process of the group . Details is printed using printk() function provided by the kernel.

Compiling: Make sure the file is present in the Desktop folder only. Otherwise an error will be produced.

Commands:

1) make
2) sudo insmod task_struct_module.ko process=<pid>
3) sudo dmesg
4) sudo rmmod task_struct_module.ko