WRITEUP FOR PART2- Ayush Mahant

2019353

The compilation of the code-

We first download the tar file of 5.9.1 kernel and then extract it to cd /usr/src

STEPS-
1) cd /usr/src/linux-5.9.1
2) sudo make config
3) sudo make -j4(number of processor cores is 4)
4) sudo make modules_install install
5) shutdown -r now

Now we write the system call

1. Go to "cd /usr/src/linux-5.9.1" and mkdir sh_task_info
2. cd sh_task_info
3. gedit sh_task_info.c

Now I'll explain the c file-

The SYSCALL_DEFINE2 function takes 2 parameters, the pid number and the filename. We declare a struct file pointer which uses filpopen to create or write in it the contents of the task_struct, i.e, the process. We check for the pid in the taskstruct and print it's corresponding attributes(nutime or runtime,pid, state,process, parent process,priority, normal priority.) If the pid is not found error is send.The kernel_write is used to write to the file and filp_close is used to close the file.

4) cd .. to linux-5.9.1
5) cd arch/x86/entry/syscall_64.tbl
And add
548 64 sh_task_info sys_sh_task_info   at the end.
6)we go to linux-5.9.1 and cd include/linux and add
asmlinkage long sys_sh_task_info(int pid, char* filename) at the end
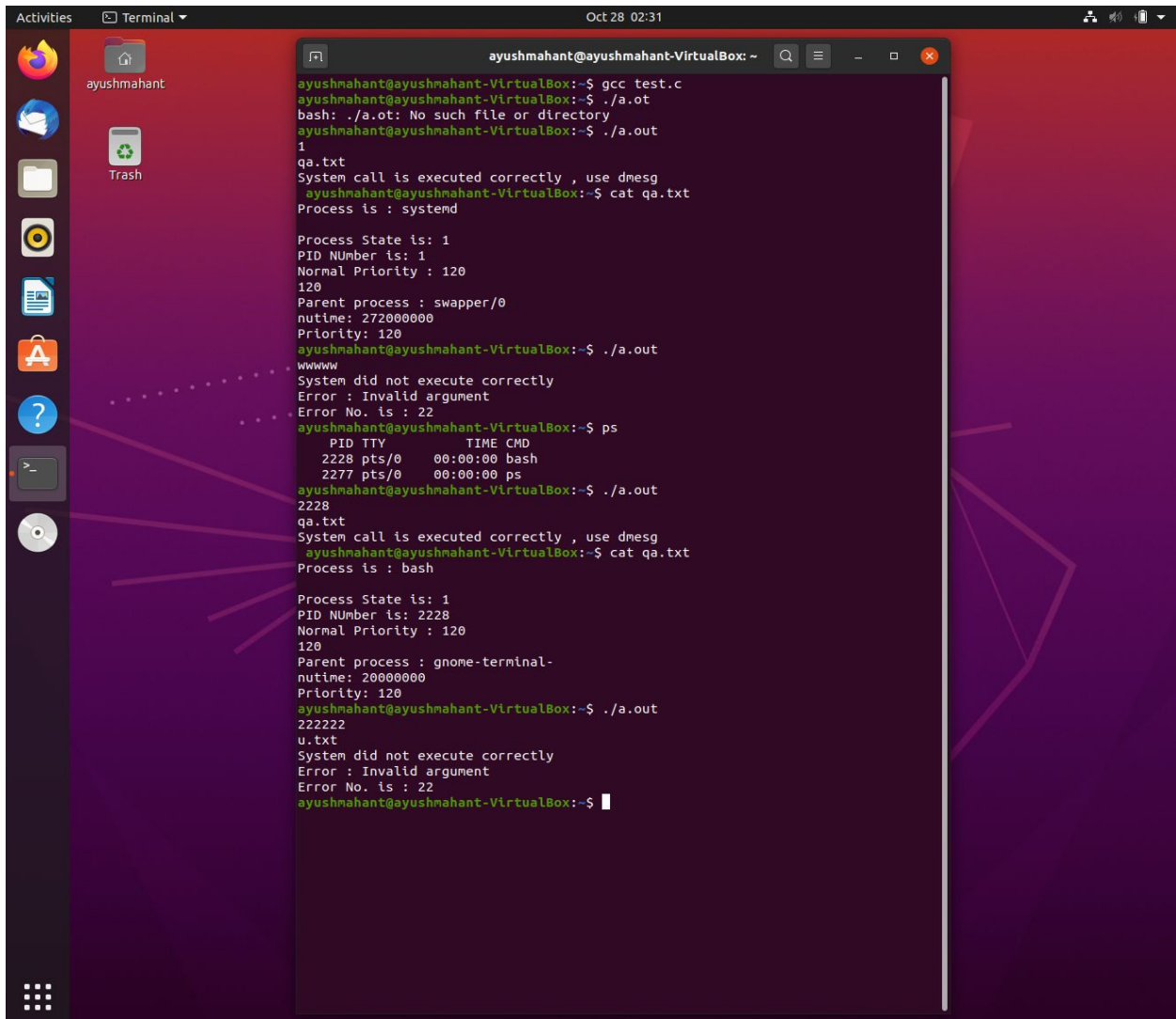7) We go to the Makefile and add sh_task_info/ to the second core-y line in the Makefile
8) we now do "sudo make -j4" and " sudo make modules_install install" and restart.

We now create the test.c file

We include the required packages. Via scanf we input the file name and the pid. We use getcwd to get the current working directory where our executable file will be created.

We then use syscall with 548 as one of it's argument and then do the error handling part.

TESTING THE CODE AND ERROR HANDLING
1) We give directly the pid and filename after the compilation.
2) WE arre returning -EISVAL for error on wrong pid if it is a string or a pid <0 or a pid not in taskstruck.