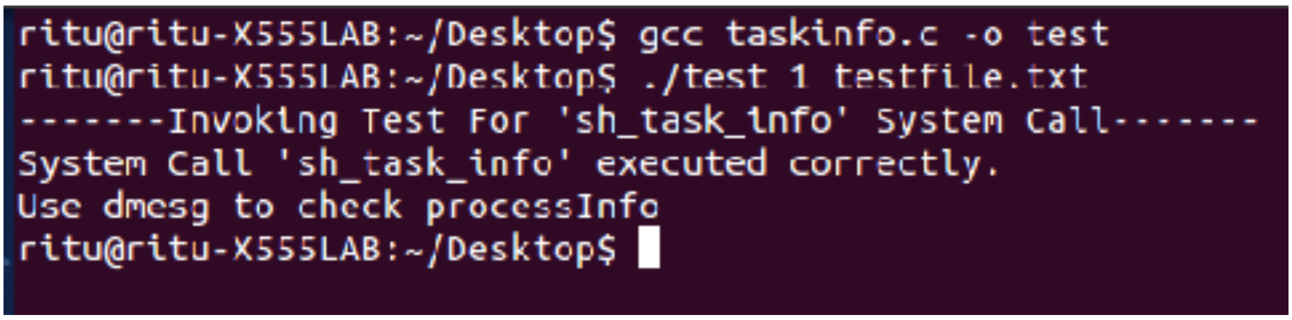
# Linux_System_call
# Sneha Sinha - 2016098 & Ritu Kumari - 2016078

System Call for Linux written in C, compiling linux source code

Part 1: Compiling the source code
The linux 3.13.0 source code is downloaded and it's extracted.
- cd to that directory
- do 'cp /boot/config-$(uname -r) .config' to copy the kernel's .config file then followed by 'make menuconfig'
- use 'sudo make -j 4 && sudo make modules_install -j 4 && sudo make install -j 4'.The kernel and modules are compiled.
- do 'update-initramfs -c -k 3.13.0' to update the kernel, and 'update-grub' to add it to the grub.

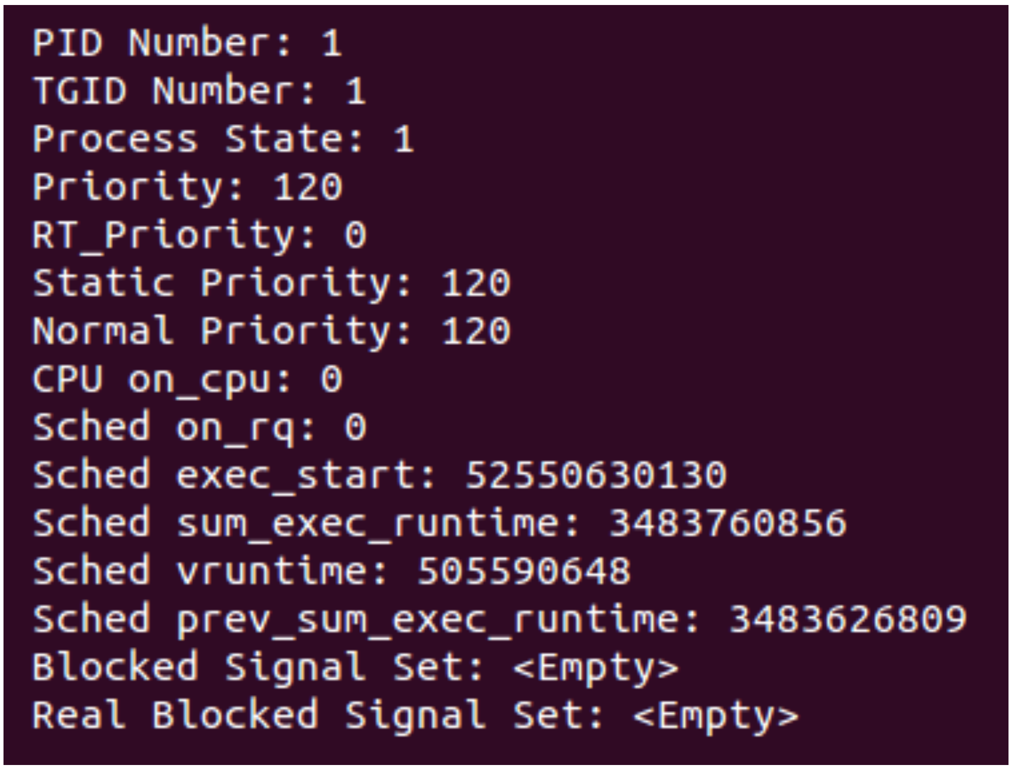
Part 2: Writing a System Call
- In the linux-3.13.0 dir, make a new dir and add the system call files there.
  - Make a new file sh_task_info.c
    - write_to_file function takes file* and char * as arguments and uses write() to write data in it.
    - In sys_sh_task_info(),a task_struct pointer task and struct file pointer file are created. For each task, it’s attributes are printed on the console and simultaneously written to the file.
  - A header file for it, sh_task_info.h
  - A Makefile for compiling the above.
- This new directory has to be added to the Linux-3.13.0's Makefile by changing the line 'core -y += kernel/ mm/ fs/ ipc/ security/ crypto/ block/' to 'core -y += kernel/ mm/ fs/ ipc/ security/ crypto/ block/ newdir/'.
- The pid for the new system call has to be added,in the arch/x86/syscalls/syscall_64.tbl. We added it at 318.
- Then, the function was included in the syscalls.h in include/
- It was recompiled again with the three parallel commands by 'sudo make -j 4 && sudo make modules_install -j 4 && sudo make install -j 4'.
- The system is restarted.

Part 3: Testing it with sample code
- The file taskinfo.c is the sample test code, compares the process IDs by using syscall. If the input process ID and the updated pid 318 match, and there is no error in the given pid and the filename sh_task_info() is executed correctly.
- It should be run with 'gcc taskinfo.c -o test'
- The executable is run with './test 1 testfile.txt'



```
ritu@ritu-X555LAB:~/Desktop$ gcc taskinfo.c -o test
ritu@ritu-X555LAB:~/Desktop$ ./test 1 testfile.txt
-------Invoking Test For 'sh_task_info' System Call-------
System Call 'sh_task_info' executed correctly.
Use dmesg to check processInfo
ritu@ritu-X555LAB:~/Desktop$ 
```

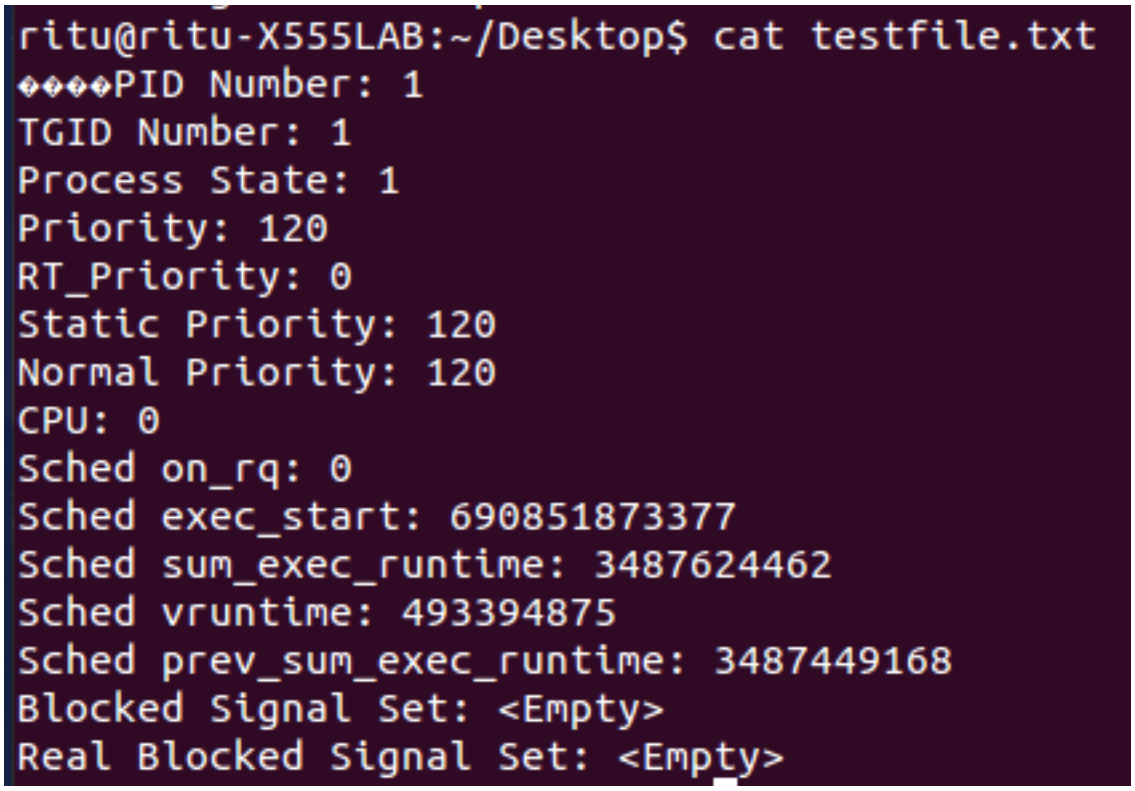
- If it executed successfully, we can check the log via 'dmesg'. The attributes corresponding to pid==1 are printed on the console, the corresponding process for it is init.



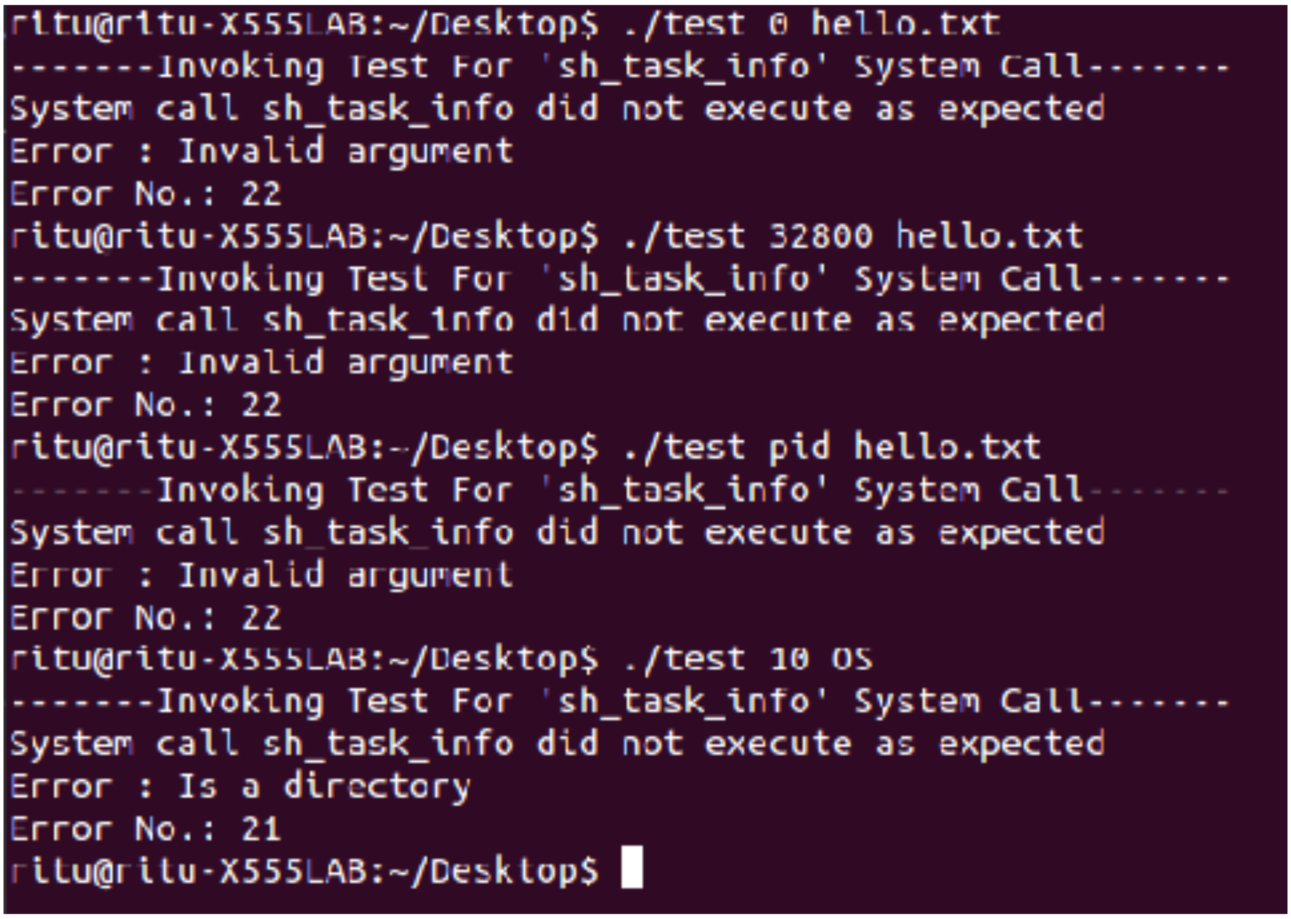
```
PID Number: 1
TGID Number: 1
Process State: 1
Priority: 120
RT_Priority: 0
Static Priority: 120
Normal Priority: 120
CPU on_cpu: 0
Sched on_rq: 0
Sched exec_start: 52550630130
Sched sum_exec_runtime: 3483760856
Sched vruntime: 505590648
Sched prev_sum_exec_runtime: 3483626809
Blocked Signal Set: <Empty>
Real Blocked Signal Set: <Empty>
```

- The process, it's process ID, process state, RT_priority, static priority and normal priority are printed.
- Alternatively, we can check the log with 'cat testfile.txt'.

- Process is an instance of a running program.
- Priority is the priority for the linux kernel.
- Realtime priority has the highest priority, static priority is the fixed priority for some jobs. Normal priority is the priority assigned to non major and non critical tasks.
- Here,”CPU” refers to on_cpu flag in the task_struct during the context switch. Here, it’s value is 0 because the process can be shifted to another CPU.
- tgid is thread pid and it’s equal to pid for a normal process.
- sched entity structure has the folowing attributes:
- sched on_vruntime is the virtual run time based on the number of runnable processes.
- sched sum_exec_runtime is the cpu time of a thread, used in the completely fair scheduler.
- sched exec_start is the runtime relative to the system boot time.
- sched on_rq is 0 because it’s on the ready queue.
- sched prev_sum_exec_runtime is the runtime of a process which has been taken off the CPU.
- Real Blocked Signal and Blocked Signal Set are empty for this signal, because we have not blocked any signals.



```
ritu@ritu-X555LAB:~/Desktop$ cat testfile.txt
����PID Number: 1
TGID Number: 1
Process State: 1
Priority: 120
RT_Priority: 0
Static Priority: 120
Normal Priority: 120
CPU: 0
Sched on_rq: 0
Sched exec_start: 690851873377
Sched sum_exec_runtime: 3487624462
Sched vruntime: 493394875
Sched prev_sum_exec_runtime: 3487449168
Blocked Signal Set: <Empty>
Real Blocked Signal Set: <Empty>
```



```
ritu@ritu-X555LAB:~/Desktop$ ./test 0 hello.txt
-------Invoking Test For 'sh_task_info' System Call-------
System call sh_task_info did not execute as expected
Error : Invalid argument
Error No.: 22
ritu@ritu-X555LAB:~/Desktop$ ./test 32800 hello.txt
-------Invoking Test For 'sh_task_info' System Call-------
System call sh_task_info did not execute as expected
Error : Invalid argument
Error No.: 22
ritu@ritu-X555LAB:~/Desktop$ ./test pid hello.txt
-------Invoking Test For 'sh_task_info' System Call-------
System call sh_task_info did not execute as expected
Error : Invalid argument
Error No.: 22
ritu@ritu-X555LAB:~/Desktop$ ./test 10 OS
-------Invoking Test For 'sh_task_info' System Call-------
System call sh_task_info did not execute as expected
Error : Is a directory
Error No.: 21
ritu@ritu-X555LAB:~/Desktop$ █
```

Part 4: Errors Handled

- 1. If the user doesn't enters an invalid pid such as a char or float
- 2. If the user enters pid <=0 or greater than 32768, the function returns the errno 22 EINVAL invalid argument.
- 3. If the sys_open() for creating a file if it doesn't exist, with write access returns an int less than 0, the function returns the errno 21 EISDIR is a directory.