Report for CSC3150 assignment 1

Name: 舒欣

Student ID: 118020362

How did I design my program?

Program1:

In this program I only have a main function, first I use fork to create a child process, if the return is negative, then creating a new process is failed, the program will just exit; if no, then for the child process I firstly print out its pid, then I call execve to execute the test program; for the parent process, I firstly print out its pid, then use waitpid to wait for the child process to return, and use WUNTRACED to catch not only normally return signal SIGCHLD, but also abnormally signal, like SIGSTOP, SIGKILL, etc. After that, according to the value of state, I print out different messages for different return signals. In detail, I divide state into three categories, EXITED, SIGNALED and STOPPED, and divide 13 subcategories in SIGNALED. Done!

Program2:

In this program, first I export the functions that I will use using EXPORT_SYMBOL(), and update and install the modified kernel. Then in program2.c I firstly define some data structures and import some kernel functions that I would use later, such as struct: wait_opts, task, function: do_execve, getname, do_wait and _do_fork. Then I write functions that I would use later in my_fork function. One is my_exec, which is to execute the test program and exit from it, and the other is my_wait, which is to get the return signal of the test program and print out different messages for different signals. **Note** that pid_t can not be used in kernel mode, since it is defined in <sys/types.h>, which is a C library, I use long to define pid. Another point to **notice** is that I use WEXITED | WUNTRACED as wo.flags, to capture both SIGSTOP and other signals, either normal or error.

Then I define my_fork, which is to fork a process (use _do_fork) to execute test program (use my_exec) and wait for and print out the return signal (use my_wait). Finally, in __init program2_init(), I use kthread_create to create a kernel thread , which executes my_fork and then wake up the process, in __exit program_exit() I print out a kernel log saying that my module has exit. Done!

Environment:

OS: Ubuntu 16.04 (32 bit)

Kernel: linux-4.10.14

Steps:

Program1: you go to program1 directory, then you type make clean, then you type make, then if you want to execute normal.c, you can type ./program1 ./normal. Other files are the same.

Program2: you should be root user to execute the program. You go to program2 directory, then you type gcc -o test test.c, then you type make clean, then you type make, then you

type insmod program2.ko, then you type dmesg, you could see module_init message, then you type rmmod program2, then you type dmesg, you could see module_exit message. If you want to see messages for different signals, you can modify test.c and repeat my direction from the very start.

Screenshots

Program1:

```
root@VM:/home/seed/Desktop/source/program1# ./program1
./segment_fault
I am the parent process, mypid is 7709
I am the child process, my pid is 7710
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSEGV program

Parent process receiving the SIGCHLD signal.
child process get SIGSEGV signal.
child process is terminated by segment_fault signal.
CHILD EXECUTION FAILED!!
root@VM:/home/seed/Desktop/source/program1#
```

```
root@VM:/home/seed/Desktop/source/program1# ./program1
./normal
I am the parent process, mypid is 7713
I am the child process, my pid is 7714
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the normal program

------------CHILD PROCESS END------------
Parent process receiving the SIGCHLD signal.
Normal termination with EXIT STATUS = 0
root@VM:/home/seed/Desktop/source/program1#
```

```
root@VM:/home/seed/Desktop/source/program1# ./program1
./stop
I am the parent process, mypid is 7717
I am the child process, my pid is 7718
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSTOP program

Parent process receiving the SIGCHLD signal.
child process get SIGSTOP signal.
child process stopped.
CHILD PROCESS STOPPED.
root@VM:/home/seed/Desktop/source/program1#
```

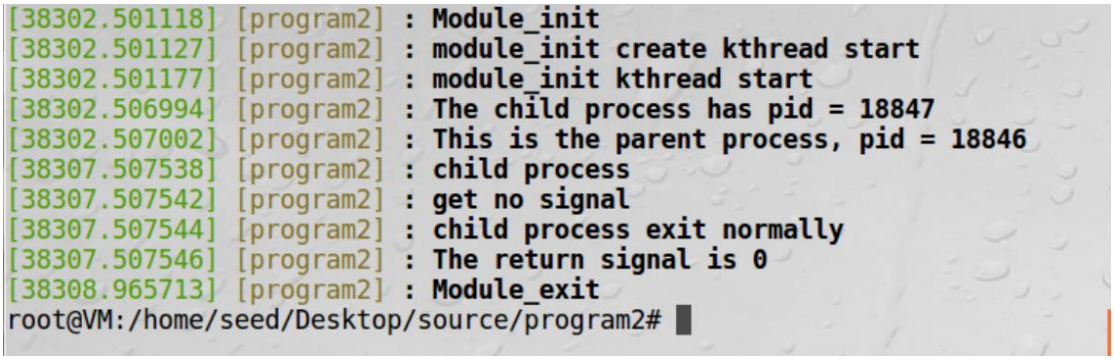
```
root@VM:/home/seed/Desktop/source/program1# ./program1
./pipe
I am the parent process, mypid is 7725
I am the child process, my pid is 7726
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGPIPE program

Parent process receiving the SIGCHLD signal.
child process get SIGPIPE signal.
child process is terminated by pipe signal.
CHILD EXECUTION FAILED!!
root@VM:/home/seed/Desktop/source/program1#
```

Program2:

```
[35997.959455] [program2] : Module_init
[35997.959463] [program2] : module_init create kthread
start
[35997.959510] [program2] : module_init kthread start
[35997.963713] [program2] : The child process has pid =
 7731
[35997.963720] [program2] : This is the parent process,
 pid = 7730
[35997.965671] [program2] : child process
[35997.965678] [program2] : get SIGBUS signal
[35997.965681] [program2] : child process has bus error
[35997.965685] [program2] : The return signal is 7
[36005.050369] [program2] : Module_exit
root@VM:/home/seed/Desktop/source/program2#
```

```
root@VM:/home/seed/Desktop/source/program2# insmod program2.ko
root@VM:/home/seed/Desktop/source/program2# rmmod program2
root@VM:/home/seed/Desktop/source/program2# dmesg | tail -10
[36495.328897] [program2] : Module_init
[36495.328905] [program2] : module_init create kthread start
[36495.329095] [program2] : module_init kthread start
[36495.329122] [program2] : The child process has pid = 9806
[36495.329126] [program2] : This is the parent process, pid = 9805
[36495.336724] [program2] : child process
[36495.336732] [program2] : get SIGSTOP signal
[36495.336736] [program2] : child process has stop error
[36495.336739] [program2] : The return signal is 19
[36499.449013] [program2] : Module_exit
root@VM:/home/seed/Desktop/source/program2#
```

```
root@VM:/home/seed/Desktop/source/program2# insmod program2.ko
root@VM:/home/seed/Desktop/source/program2# rmmod program2
root@VM:/home/seed/Desktop/source/program2# dmesg | tail -10
[36627.193230] [program2] : Module_init
[36627.193237] [program2] : module_init create kthread start
[36627.193323] [program2] : module_init kthread start
[36627.193350] [program2] : The child process has pid = 10497
[36627.193354] [program2] : This is the parent process, pid = 10496
[36627.207292] [program2] : child process
[36627.207300] [program2] : get SIGILL signal
[36627.207304] [program2] : child process has illegal_instr error
[36627.207307] [program2] : The return signal is 4
[36630.839925] [program2] : Module_exit
root@VM:/home/seed/Desktop/source/program2#
```



Note that when you test for normal case in program2, you should decomment the line raise(SIGBUS), also change return 100 to return 0.

What do I learn?

From program 1, I learned how to create a process and let parent to wait for it and capture its return signal. More than that, I can create a process to execute another program code, which is really cool.

From program2, I learned how to do all things in program1 in kernel mode, which is quite amazing. Also, I learned how to read kernel code and how to use kernel functions.