CSC3150 Projecet1

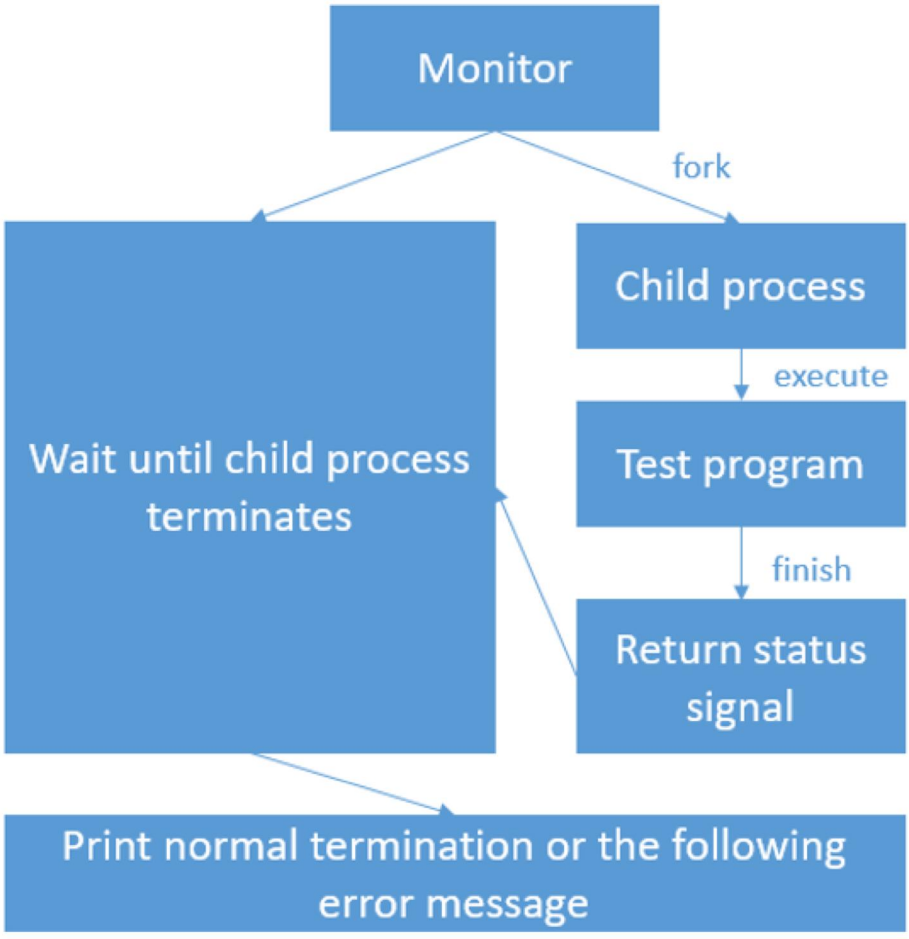
120090533 Zhou Zerui

Design

Task 1

1. In Task 1, we should write a program (program1.c) that implement the functions below:

    1) fork a child process to execute the test program
    2) parent process wait for the child process
    3) child process execute and send signal to parent process
    4) print out the signal of the child process



2. The detailed design is as follows:

1) Fork a child process: **fork()**

To fork a child process, we need to call the fork() function, which will create a child process and return a pid (process ID). Then, we can use pid to check if we are in a parent or child process.

2) Excute child process: **execve()**

To execute a child process, we need to call the exec function. execve() is one of the exec functions. We can pass the executable file name and arguments to execve() and make an execution.

3) Make parent process wait: **waitpid()**

To make parent process wait, we need to call the wait fuction. We should use WUNTRACED in the wait function waitpid() to be the wait flag.

4) The parent receives signals from the child and prints the signals.

Task 2

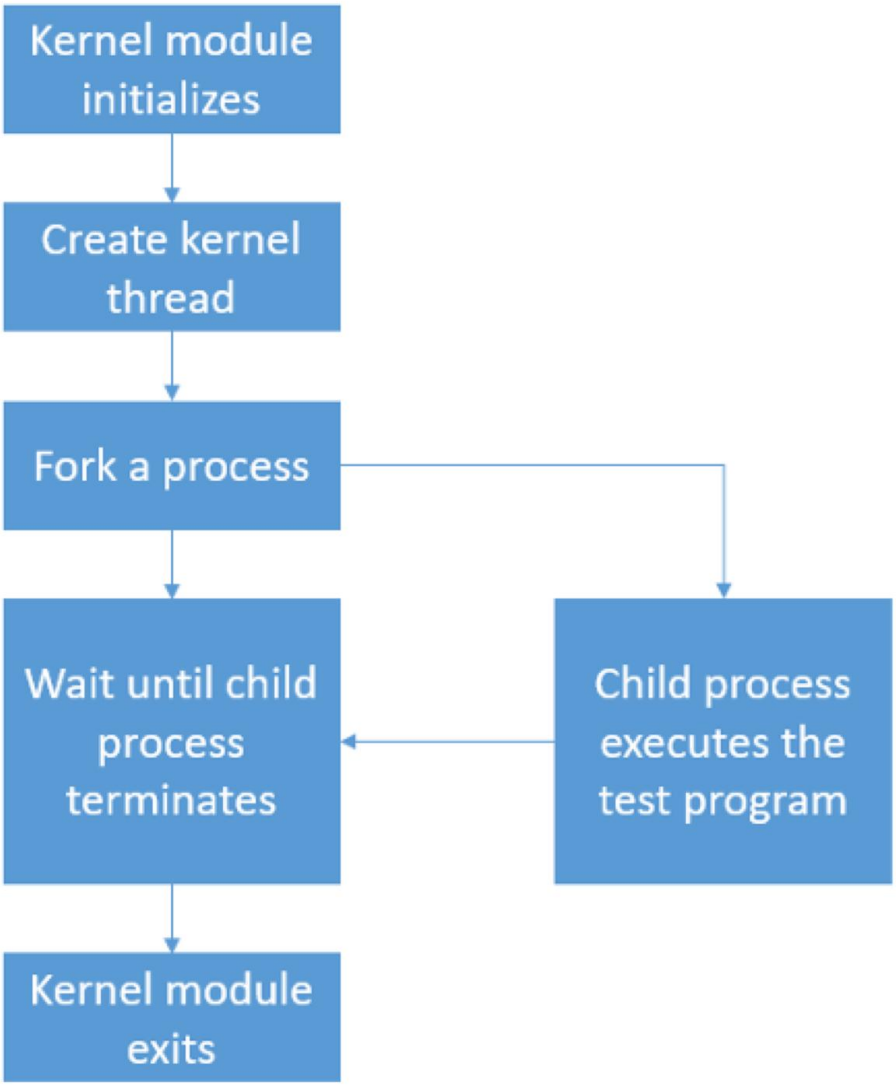
1. In Task 2, we should write a program (program2.c) that implement the functions below:

1) create a kernel thread and run my_fork function

2) fork a process in my_fork and execute the test programs

3) parent process wait until the child process terminates

4) parent process receive the signal raised by child process and print them



2. The detailed design is as follows:

1) Compile the Kernel: **EXPORT_SYMBOL()**

To make the program work, we need to use EXPORT_SYMBOL() and compile the kernel. We need to implement EXPORT_SYMBOL() in kernel source code to be used in the other modules. Then, export do_wait(), do_execve(), kernel_clone() and getname_kernel().

2) Fork a child process: **kernel_clone()**

To fork a child process, we can call kernel_clone() which will create a child process. It will return the pid of child process if the fork is successfully executed.

3) Execute child process: **do_execve()**

To execute child process, we can call the do_execve(). Then, do_execve() function was encapsulated in my_exec() function. We can pass the the pointer of my_exec() function as an argument to the kernel_clone().

4) Parent process wait child process: **do_wait()**

To make the parent process wait until child process terminates, we can use do_wait() will. The do_wait was encapsulated in the my_wait function. In the kernel mod, when the system call do_wait() is executed, exit.ko module will be loaded.

5) Parent process print the signal form child process: **my_wait()**

To print the signal from child process, we can use my_wait(). The do_wait() function will change the information in the wait_opts wo. We need to set the wo_flag of the wait options to make the parent process report when the child process stopped or terminated.

Development Environment

Linux, Linux Kernel, GCC Version

```
vagrant@csc3150:~$ cat /etc/issue
Ubuntu 16.04.7 LTS \n \l

vagrant@csc3150:~$ uname -r
5.10.27
vagrant@csc3150:~$ gcc --version
gcc (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609
```

Set up

Firstly, we install Vitrualbox and Vagrant. After installation, reboot the machine. Then, set the Virtualbox permissions to make it work properly. Set up a directory for csc3150 using **mkdir -p ~/csc3150**. Launch terminal and change current directory to the one you set up using **cd ~/csc3150**, then execute **vagrant init cyzhu/csc3150** and **vagrant up**. After download the system image, a Virtualbox window may pop up. Leave it open but put it aside. After the VM is set up, we can set up the ssh to connect to the VM. In terminal, execute **mkdir -p ~/.ssh && vagrant ssh-config >> ~/.ssh/config** and now we can connect to VM in terminal with **ssh default**.

Secondly, we set up Remote SSH plugin in VS Code. After installing it, go to the remote explorer tab, and find SSH Target called default. Click the icon to connect to the VM and launch a new window. After it finshes loading, start a terminal. In the terminal you just opened, install essential dependencies and libraries: **sudo apt update && sudo apt install -y**

build-essential. After it finishes, create a directory for the course: **mkdir -p ~/csc3150.** After everything, you can open a folder in the VS Code window to locate to the directory you create.Then, we try a hello world program on the VM. As usual, open a (remote) terminal in VS Code and compile the hello world program using **gcc hello.c -o hello** and try to run it.

Finally, we download source code from mirro: https://mirror.tuna.tsinghua.edu.cn/kernel/v5.x/ an install Dependency and development tools using **sudo apt-get install libncurses-dev gawk flex bison openssl libssl-dev dkms libelf-dev libudev-dev libpci-dev libiberty-dev autoconf llvm dwarves**. After that, we extract the source file to /home/seed/work using **cp linux-5.10.27.tar.gz /home/seed/work** and **cd /home/seed/work**. Then, we decompress the pachage using **sudo tar xvf linux-5.10.27.tar.gz**. After copying config from /boot to /home/seed/work/linux-5.10.27, we login root account and go to kernel source directory using **sudo su** and **cd /home/seed/work/linux-5.10.27**. In order to clean previous setting and start configuration we use **make mrproper** and **make clean**, as well as **make menuconfig**. Then, save the config and exit. After everything, we Build kernel Image and modules using **make –j$(nproc)** and install kernel modules as well as kernel by **make modules_install** and **make install**. We reboot to load a new kernel and check the version.

Output

Program1

1. Type **make** to make the files

2. Type **./program1** + executable file name (eg. ./normal)

The output is as below:

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./abort
Process start to fork
I'm the Parent Process, my pid = 21742
I'm the Child Process, my pid = 21743
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGABRT program

Parent process receives the SIGCHLD signal
child process get SIGABRT signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./alarm
Process start to fork
I'm the Parent Process, my pid = 21793
I'm the Child Process, my pid = 21794
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGALRM program

Parent process receives the SIGCHLD signal
child process get SIGALRM signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./bus
Process start to fork
I'm the Parent Process, my pid = 21844
I'm the Child Process, my pid = 21845
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGBUS program

Parent process receives the SIGCHLD signal
child process get SIGBUS signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./floating
Process start to fork
I'm the Parent Process, my pid = 21886
I'm the Child Process, my pid = 21887
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGFPE program

Parent process receives the SIGCHLD signal
child process get SIGFPE signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./hangup
Process start to fork
I'm the Parent Process, my pid = 21949
I'm the Child Process, my pid = 21950
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGHUP program

Parent process receives the SIGCHLD signal
child process get SIGHUP signal
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./illegal_instr
Process start to fork
I'm the Parent Process, my pid = 22011
I'm the Child Process, my pid = 22012
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGILL program

Parent process receives the SIGCHLD signal
child process get SIGILL signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./interrupt
Process start to fork
I'm the Parent Process, my pid = 22083
I'm the Child Process, my pid = 22084
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGINT program

Parent process receives the SIGCHLD signal
child process get SIGINT signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./kill
Process start to fork
I'm the Child Process, my pid = 22158
Child process start to execute test program:
I'm the Parent Process, my pid = 22157
------------CHILD PROCESS START------------
This is the SIGKILL program

Parent process receives the SIGCHLD signal
child process get SIGKILL signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./normal
Process start to fork
I'm the Parent Process, my pid = 22195
I'm the Child Process, my pid = 22196
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the normal program

------------CHILD PROCESS END------------
Parent process receives the SIGCHLD signal
Normal termination with EXIT STATUS = 0
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./pipe
Process start to fork
I'm the Parent Process, my pid = 22236
I'm the Child Process, my pid = 22237
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGPIPE program

Parent process receives the SIGCHLD signal
child process get SIGPIPE signal
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./quit
Process start to fork
I'm the Parent Process, my pid = 22250
I'm the Child Process, my pid = 22251
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGQUIT program

Parent process receives the SIGCHLD signal
child process get SIGQUIT signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./segment_fault
Process start to fork
I'm the Parent Process, my pid = 22301
I'm the Child Process, my pid = 22302
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSEGV program

Parent process receives the SIGCHLD signal
child process get SIGSEGV signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./stop
Process start to fork
I'm the Parent Process, my pid = 22328
I'm the Child Process, my pid = 22329
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSTOP program

Parent process receives the SIGCHLD signal
child process get SIGSTOP signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./terminate
Process start to fork
I'm the Parent Process, my pid = 22354
I'm the Child Process, my pid = 22355
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTERM program

Parent process receives the SIGCHLD signal
child process get SIGTERM signal
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program1$ ./program1 ./trap
Process start to fork
I'm the Parent Process, my pid = 22368
I'm the Child Process, my pid = 22369
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTRAP program

Parent process receives the SIGCHLD signal
child process get SIGTRAP signal
```

Program2

1. Type **make** to make files
2. Sign in the root account using **sudo su**
3. Insert the module **insmod**, remove the module **rmmod**, check the log by **dmesg**

The output is as below:

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 9
[11918.688794] [program2] : module_init {Zhouzerui} {120090533}
[11918.692750] [program2] : module_init create kthread start
[11918.697102] [program2] : module_init kthread start
[11918.700772] [program2] : THe child process has pid = 18951
[11918.703408] [program2] : This is the parent process, pid = 18950
[11918.706200] [program2] : child process
[11923.704675] [program2] : child process runs normally
[11923.708443] [program2] : The return signal is 0
[11927.339954] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[ 9800.111058] [program2] : module_init {Zhouzerui} {120090533}
[ 9800.112756] [program2] : module_init create kthread start
[ 9800.115377] [program2] : module_init kthread start
[ 9800.117117] [program2] : THe child process has pid = 14144
[ 9800.119360] [program2] : This is the parent process, pid = 14143
[ 9800.123641] [program2] : child process
[ 9800.123652] [program2] : get SIGHUP signal
[ 9800.125473] [program2] : child process is hung up
[ 9800.128034] [program2] : The return signal is 1
[ 9802.048395] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[ 9886.581743] [program2] : module_init {Zhouzerui} {120090533}
[ 9886.584522] [program2] : module_init create kthread start
[ 9886.587057] [program2] : module_init kthread start
[ 9886.589861] [program2] : THe child process has pid = 14207
[ 9886.591885] [program2] : This is the parent process, pid = 14206
[ 9886.594945] [program2] : child process
[ 9886.594967] [program2] : get SIGINT signal
[ 9886.596543] [program2] : child process gets interrupt from keyboard
[ 9886.598604] [program2] : The return signal is 2
[ 9889.981568] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[ 9949.407079] [program2] : module_init {Zhouzerui} {120090533}
[ 9949.411726] [program2] : module_init create kthread start
[ 9949.415398] [program2] : module_init kthread start
[ 9949.418442] [program2] : THe child process has pid = 14284
[ 9949.420976] [program2] : This is the parent process, pid = 14283
[ 9949.423326] [program2] : child process
[ 9949.662655] [program2] : get SIGQUIT signal
[ 9949.667560] [program2] : child process quits
[ 9949.668945] [program2] : The return signal is 3
[ 9951.836241] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10087.448206] [program2] : module_init {Zhouzerui} {120090533}
[10087.452000] [program2] : module_init create kthread start
[10087.455366] [program2] : module_init kthread start
[10087.458391] [program2] : THe child process has pid = 14422
[10087.462823] [program2] : This is the parent process, pid = 14421
[10087.466824] [program2] : child process
[10087.637524] [program2] : get SIGILL signal
[10087.642429] [program2] : child process encounters illegal instruction
[10087.647761] [program2] : The return signal is 4
[10089.244969] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10141.994478] [program2] : module_init {Zhouzerui} {120090533}
[10141.998499] [program2] : module_init create kthread start
[10142.002612] [program2] : module_init kthread start
[10142.004975] [program2] : THe child process has pid = 14476
[10142.007124] [program2] : This is the parent process, pid = 14475
[10142.010229] [program2] : child process
[10142.182157] [program2] : get SIGTRAP signal
[10142.186545] [program2] : child process encounters breaking point for debugging
[10142.192161] [program2] : The return signal is 5
[10143.921319] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10202.290153] [program2] : module_init {Zhouzerui} {120090533}
[10202.292511] [program2] : module_init create kthread start
[10202.295353] [program2] : module_init kthread start
[10202.297996] [program2] : THe child process has pid = 14557
[10202.301285] [program2] : This is the parent process, pid = 14556
[10202.304397] [program2] : child process
[10202.413517] [program2] : get SIGABRT signal
[10202.415456] [program2] : child process encounters abnormal termination
[10202.417269] [program2] : The return signal is 6
[10205.479423] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10254.229778] [program2] : module_init {Zhouzerui} {120090533}
[10254.234240] [program2] : module_init create kthread start
[10254.274660] [program2] : module_init kthread start
[10254.276740] [program2] : THe child process has pid = 14612
[10254.295958] [program2] : This is the parent process, pid = 14611
[10254.309082] [program2] : child process
[10254.507966] [program2] : get SIGBUS signal
[10254.509725] [program2] : child process encounters bus error
[10254.511774] [program2] : The return signal is 7
[10255.809796] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10345.709395] [program2] : module_init {Zhouzerui} {120090533}
[10345.712006] [program2] : module_init create kthread start
[10345.715238] [program2] : module_init kthread start
[10345.717878] [program2] : THe child process has pid = 14666
[10345.720178] [program2] : This is the parent process, pid = 14665
[10345.722536] [program2] : child process
[10345.835298] [program2] : get SIGFPE signal
[10345.836378] [program2] : child process encounters floating-point exception
[10345.838012] [program2] : The return signal is 8
[10347.564661] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10401.593215] [program2] : module_init {Zhouzerui} {120090533}
[10401.597719] [program2] : module_init create kthread start
[10401.601063] [program2] : module_init kthread start
[10401.604649] [program2] : THe child process has pid = 14729
[10401.606951] [program2] : This is the parent process, pid = 14728
[10401.610321] [program2] : child process
[10401.610331] [program2] : get SIGKILL signal
[10401.612935] [program2] : child process encounters forced-process termination
[10401.615071] [program2] : The return signal is 9
[10404.375925] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10453.753972] [program2] : module_init {Zhouzerui} {120090533}
[10453.757946] [program2] : module_init create kthread start
[10453.760935] [program2] : module_init kthread start
[10453.763677] [program2] : THe child process has pid = 14787
[10453.767460] [program2] : This is the parent process, pid = 14786
[10453.771350] [program2] : child process
[10453.962529] [program2] : get SIGSEGV signal
[10453.964250] [program2] : child process refers to invalid memory
[10453.966508] [program2] : The return signal is 11
[10456.709079] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10561.825503] [program2] : module_init {Zhouzerui} {120090533}
[10561.828925] [program2] : module_init create kthread start
[10561.831610] [program2] : module_init kthread start
[10561.857106] [program2] : THe child process has pid = 15235
[10561.873721] [program2] : This is the parent process, pid = 15234
[10561.876270] [program2] : child process
[10561.876281] [program2] : get SIGPIPE signal
[10561.878736] [program2] : child process writes to pipe with no readers
[10561.880446] [program2] : The return signal is 13
[10563.698523] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10682.455695] [program2] : module_init {Zhouzerui} {120090533}
[10682.458721] [program2] : module_init create kthread start
[10682.461068] [program2] : module_init kthread start
[10682.464329] [program2] : THe child process has pid = 15681
[10682.467921] [program2] : This is the parent process, pid = 15680
[10682.471459] [program2] : child process
[10682.471467] [program2] : get SIGALRM signal
[10682.472952] [program2] : child process is alarmed by real-timerclock
[10682.475450] [program2] : The return signal is 14
[10684.469429] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10736.638144] [program2] : module_init {Zhouzerui} {120090533}
[10736.641080] [program2] : module_init create kthread start
[10736.645892] [program2] : module_init kthread start
[10736.650177] [program2] : THe child process has pid = 15722
[10736.652613] [program2] : This is the parent process, pid = 15721
[10736.655520] [program2] : child process
[10736.655525] [program2] : get SIGTERM signal
[10736.657120] [program2] : child process terminates
[10736.659253] [program2] : The return signal is 15
[10738.486831] [program2] : module_exit./my
```

```
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo insmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ sudo rmmod program2.ko
vagrant@csc3150:~/csc3150/Assignment_1_120090533/source/program2$ dmesg | tail -n 10
[10778.948151] [program2] : module_init {Zhouzerui} {120090533}
[10778.952473] [program2] : module_init create kthread start
[10778.957398] [program2] : module_init kthread start
[10778.960722] [program2] : THe child process has pid = 15774
[10778.963614] [program2] : This is the parent process, pid = 15773
[10778.966990] [program2] : child process
[10778.966994] [program2] : get SIGSTOP signal
[10778.968915] [program2] : child process stops
[10778.971536] [program2] : The return signal is 19
[10780.432476] [program2] : module_exit./my
```

What did I learned

1. I learned how to download a virtual machine and configure the environment for it.

2. I learned to set up Remote SSH plugin in VS Code, which brought great convenience.

3. I learned how to download the kernel and compile it properly.

4. In Task1, I learned how to create child process and how to reieve their signal. I also learned how to use fork() function to fork a child process, use exec function execve() to execute a child process, as well as use wait fuction waitpid() to make parent process wait for the child process.

5. In Task2, I learned how to modify kernel files and recompile the kernel. I also learned how to insert and remove modules, such as kernel_clone() which will create a child process, do_execve() which will execute child process, do_wait() which will make the parent process wait.

In conclusion, this project gave me practice about process and kernel, which has increased my coding skills and experience. In addition, I think my ability to solve problems independently was improved, such as using search engines.