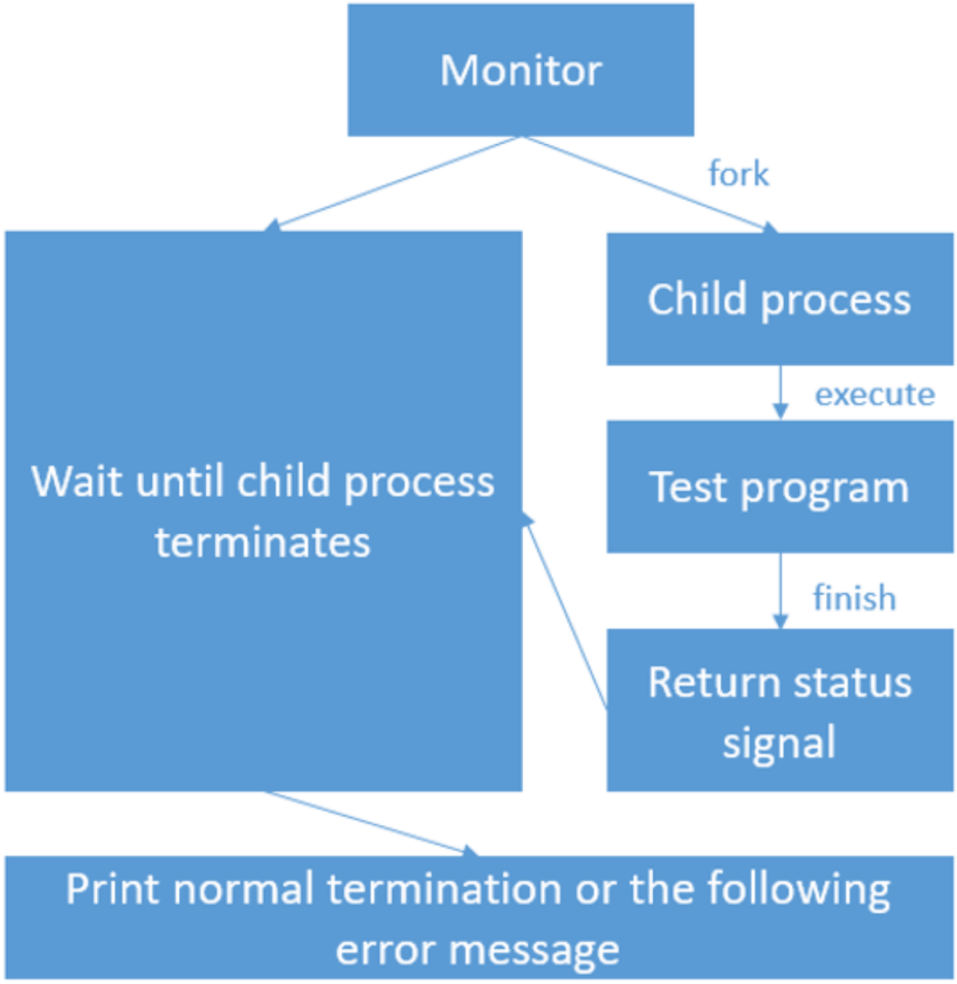
Name: Chen Junzhi
Student ID: 120090777

Program1:

Usage:

In user mode, fork a child process to execute the test program and print out the termination information and signal received. The flow chart is as followed.



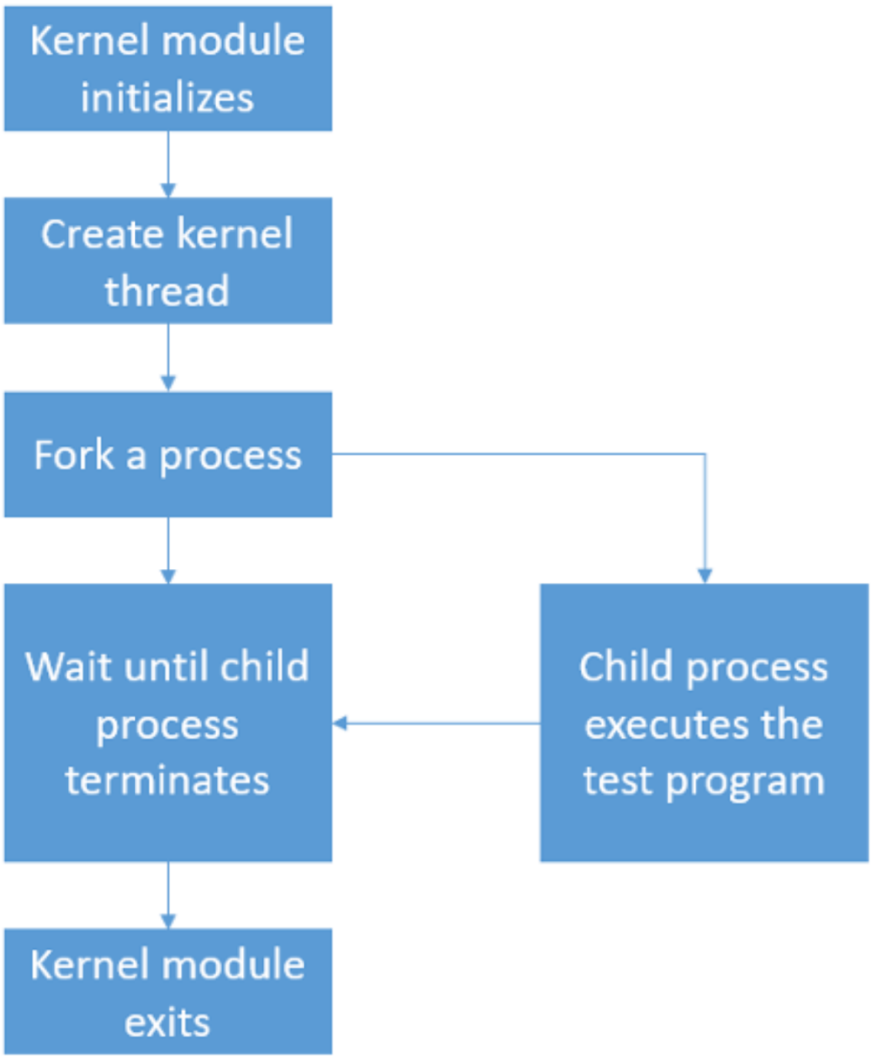
Implement Details:

1. Use fork() to fork a child process
2. Use execve() to execute the child process
3. Use waitpid() to keep parent process to wait for the child process to end:
   Argument: waitpid(pid, &state, WUNTRACED)
   WUNTRACED is used to terminate the child process when receive SIGSTOP
4. Use WIFEXITED(), WIFSIGNALED() and WIFSTOPPED() to handle different termination condition and print out the signal parent process receive.

Program2:

Usage:

In kernel mode, fork a child process to execute the test program and print out the termination condition and signals parent process receive. It is actually the LKM (loadable kernel module). The flowchart is as followed.



Implement Details:

1. Use do_execve() to implement my_execve() to execute the child process
2. Use do_wait() to implement my_wait() to keep the parent process waiting for the child process and capture the signal raising by the child process:

   The status of the child process is stored in wo.wo_stat. This program use this to print out the termination condition and signal parent process receive.
3. Implement my_fork() to fork the child process and execute it:

   Use kernel_thread() to fork a new process (my_execve() is used as a parameter of kernel_thread()).

   Use my_wait() to keep the parent process waiting for the child process and capture the signal raising by the child process.
4. Implement __init program2_init() to make the kernel work:

   Use kthread_create() to create a new thread in the kernel to carry out the fork

process (my_fork() is used as a parameter of kthread_create())

Development Environment:

OS version: Ubuntu 32-bit

kernel version: Linux-5.10.146

gcc –version: gcc (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609

How to execute the program:

Program 1:

$ cd /path program1 located in/

$ make

$ ./program1 ./test_program_name

(You can see the output now)

$make clean

Program 2:

Prepare in advance：

Ubuntu 5.10.146

Install the modules, export and extern the function including do_wait, do_execve, getname_kernel, do_exit and kernel_thread.

Then:

$ cd /path program2 located in/

$ make

$ sudo insmod program2.ko

$ sudo rmmod program2

$ dmesg | tail -n 10

(You can see the output now)

$ make clean

Development Environment set up and Compile Kernel:

Set up VM

· Follow the PPT of tutorial I, install virtualbox and vagrant, then reboot the machine.

· Set up a directory for csc3150

· Launch powershell with Administrator privilege and change current directory to the one you set up (e.g. cd D:\csc3150).

· Execute “vagrant init cyzhu/csc3150”

· Then execute “vagrant up”. It may take a while to download the system image. After that a virtualbox window may pop up. Leave it open but put it aside.

Set up VS Code

·After installing, go to the remote explorer tab, click config in SSH-TARGETS.

·Go back to powershell (make sure you are still in the csc3150 directory) and execute vagrant ssh-config. Copy everything but the last line (Loglevel) to the ssh config and save the file, as is demostrated in the picture.

·Now you can find SSH Target called default (if not, may sure you have save the config or you can click the refresh button). Click the icon to connect to the VM and launch a new window.

·In the terminal you just opened, install essential dependencies and libraries: sudo apt update && sudo apt install -y build-essential (it may take a while). After it finishes, create a directory for the course: mkdir -p ~/csc3150. Then you can run your code there.

Compile kernel

·Download source code from “http://www.kernel.org”.

·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 auwoconf llvm dwarves”.

·Create file in “/home” : “home/seed/work”. Extract the source file to “/home/seed/work”

·Type following commands:

“$cp KERNEL_FILE.tar.xz /home/seed/work”

“$cd /home/seed/work”

“$sudo tar xvf KERNEL_FILE.tar.xz”

·Copy config from “/boot” to “/home/seed/work/KERNEL_FILE” (here mine is linux-5.10.146”

·Commands:

“$sudo su”

“cd /home/seed/work/KERNEL_FILE”

“$make mrproper”

“$make clean”

“$make menuconfig”

“$make bzImage -j$(nproc)”

"$make modules -j$(nproc)"

"$make modules_install"

"$make install"

·Reboot finally.

Output:

Program 1:

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 abort
Process start to fork
I'm the Parent Process, my pid = 85260
I'm the Child Process, my pid = 85261
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGABRT program


Parent process receives SIGCHLD signal
Child process get SIGABRT signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 alarm
Process start to fork
I'm the Parent Process, my pid = 85287
I'm the Child Process, my pid = 85288
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGALRM program


Parent process receives SIGCHLD signal
Child process get SIGALRM signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 bus
Process start to fork
I'm the Parent Process, my pid = 85362
I'm the Child Process, my pid = 85363
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGBUS program


Parent process receives SIGCHLD signal
Child process get SIGBUS signal
```

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 floating
Process start to fork
I'm the Parent Process, my pid = 85404
I'm the Child Process, my pid = 85405
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGFPE program


Parent process receives SIGCHLD signal
Child process get SIGFPE signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 hangup
Process start to fork
I'm the Parent Process, my pid = 85431
I'm the Child Process, my pid = 85432
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGHUP program


Parent process receives SIGCHLD signal
Child process get SIGHUP signal
```

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 illegal_instr
Process start to fork
I'm the Parent Process, my pid = 85507
I'm the Child Process, my pid = 85508
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGILL program


Parent process receives SIGCHLD signal
Child process get SIGILL signal
```

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 interrupt
Process start to fork
I'm the Parent Process, my pid = 85620
I'm the Child Process, my pid = 85621
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGINT program


Parent process receives SIGCHLD signal
Child process get SIGINT signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 kill
Process start to fork
I'm the Parent Process, my pid = 85659
I'm the Child Process, my pid = 85660
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGKILL program


Parent process receives SIGCHLD signal
Child process get SIGKILL signal
```

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 normal
Process start to fork
I'm the Parent Process, my pid = 85685
I'm the Child Process, my pid = 85686
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the normal program

------------CHILD PROCESS END------------

Parent process receives SIGCHLD signal
Normal termination with EXIT STATUS = 0
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 pipe
Process start to fork
I'm the Parent Process, my pid = 85723
I'm the Child Process, my pid = 85724
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGPIPE program


Parent process receives SIGCHLD signal
Child process get SIGPIPE signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 quit
Process start to fork
I'm the Child Process, my pid = 85762
Child process start to execute test program:
I'm the Parent Process, my pid = 85761
------------CHILD PROCESS START------------
This is the SIGQUIT program


Parent process receives SIGCHLD signal
Child process get SIGQUIT signal
```

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 segment_fault
Process start to fork
I'm the Parent Process, my pid = 85788
I'm the Child Process, my pid = 85789
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSEGV program


Parent process receives SIGCHLD signal
Child process get SIGSEGV signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 stop
Process start to fork
I'm the Parent Process, my pid = 85815
I'm the Child Process, my pid = 85816
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSTOP program


Parent process receives SIGCHLD signal
child process get SIGSTOP signal
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 terminate
Process start to fork
I'm the Parent Process, my pid = 85853
I'm the Child Process, my pid = 85854
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTERM program


Parent process receives SIGCHLD signal
Child process get SIGTERM signal
```

```
csc3150@csc3150:~/csc3150/HW1/source/program1$ ./program1 trap
Process start to fork
I'm the Parent Process, my pid = 85867
I'm the Child Process, my pid = 85868
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTRAP program


Parent process receives SIGCHLD signal
Child process get SIGTRAP signal
```

Program2:

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[  932.598227] [program2] : Module_init {Junzhi Chen} {120090777}
[  932.598229] [program2] : Module_init create kthread start
[  932.599447] [program2] : module_init kthread starts
[  932.599792] [program2] : The child process has pid =4228
[  932.599794] [program2] : This is the parent process, pid=4227
[  932.599977] [program2] : child process
[  934.603400] [program2] : get SIGALARM signal
[  934.603405] [program2] : child process has alarm error
[  934.603407] [program2] : The return signal is 14
[  937.399866] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 1167.607788] [program2] : Module_init {Junzhi Chen} {120090777}
[ 1167.607790] [program2] : Module_init create kthread start
[ 1167.608392] [program2] : module_init kthread starts
[ 1167.608679] [program2] : The child process has pid =4979
[ 1167.608680] [program2] : This is the parent process, pid=4978
[ 1167.608856] [program2] : child process
[ 1167.723923] [program2] : get SIGBUS signal
[ 1167.723925] [program2] : child process terminated
[ 1167.723925] [program2] : The return signal is 7
[ 1173.941570] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 2004.614517] [program2] : Module_init {Junzhi Chen} {120090777}
[ 2004.614534] [program2] : Module_init create kthread start
[ 2004.616009] [program2] : module_init kthread starts
[ 2004.616479] [program2] : The child process has pid =10840
[ 2004.616481] [program2] : This is the parent process, pid=10839
[ 2004.617040] [program2] : child process
[ 2004.765335] [program2] : get SIGILL signal
[ 2004.765337] [program2] : child process has illegal instruction error
[ 2004.765337] [program2] : The return signal is 4
[ 2007.837414] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 2098.509601] [program2] : Module_init {Junzhi Chen} {120090777}
[ 2098.509603] [program2] : Module_init create kthread start
[ 2098.510330] [program2] : module_init kthread starts
[ 2098.510480] [program2] : The child process has pid =11520
[ 2098.510482] [program2] : This is the parent process, pid=11519
[ 2098.510865] [program2] : child process
[ 2098.511666] [program2] : get SIGINT signal
[ 2098.511667] [program2] : terminal interrupt
[ 2098.511667] [program2] : The return signal is 2
[ 2101.569631] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 2220.911215] [program2] : Module_init {Junzhi Chen} {120090777}
[ 2220.911217] [program2] : Module_init create kthread start
[ 2220.911969] [program2] : module_init kthread starts
[ 2220.912316] [program2] : The child process has pid =12617
[ 2220.912318] [program2] : This is the parent process, pid=12616
[ 2220.912642] [program2] : child process
[ 2220.914313] [program2] : get SIGKILL signal
[ 2220.914315] [program2] : child process is killed
[ 2220.914316] [program2] : The return signal is 9
[ 2225.118135] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 2415.057784] [program2] : Module_init {Junzhi Chen} {120090777}
[ 2415.057786] [program2] : Module_init create kthread start
[ 2415.058569] [program2] : module_init kthread starts
[ 2415.058802] [program2] : The child process has pid =13956
[ 2415.058803] [program2] : This is the parent process, pid=13955
[ 2415.059079] [program2] : child process
[ 2415.060095] [program2] : get SIGPIPE signal
[ 2415.060097] [program2] : child process has pipe error
[ 2415.060097] [program2] : The return signal is 13
[ 2418.347732] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 2499.466112] [program2] : Module_init {Junzhi Chen} {120090777}
[ 2499.466114] [program2] : Module_init create kthread start
[ 2499.466848] [program2] : module_init kthread starts
[ 2499.467226] [program2] : The child process has pid =14623
[ 2499.467227] [program2] : This is the parent process, pid=14622
[ 2499.467691] [program2] : child process
[ 2499.581360] [program2] : get SIGQUIT signal
[ 2499.581362] [program2] : terminal quit
[ 2499.581362] [program2] : The return signal is 3
[ 2504.332018] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 2579.209064] [program2] : Module_init {Junzhi Chen} {120090777}
[ 2579.209066] [program2] : Module_init create kthread start
[ 2579.209636] [program2] : module_init kthread starts
[ 2579.210072] [program2] : The child process has pid =15303
[ 2579.210073] [program2] : This is the parent process, pid=15302
[ 2579.210403] [program2] : child process
[ 2579.318784] [program2] : get SIGSEGV signal
[ 2579.318785] [program2] : child process has segmentation fault error
[ 2579.318785] [program2] : The return signal is 11
[ 2582.263456] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 3046.764908] [program2] : Module_init {Junzhi Chen} {120090777}
[ 3046.764909] [program2] : Module_init create kthread start
[ 3046.765627] [program2] : module_init kthread starts
[ 3046.765953] [program2] : The child process has pid =17329
[ 3046.765954] [program2] : This is the parent process, pid=17328
[ 3046.766287] [program2] : child process
[ 3046.767395] [program2] : get SIGSTOP signal
[ 3046.767396] [program2] : child process stopped
[ 3046.767397] [program2] : The return signal is 19
[ 3049.738454] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 3139.012531] [program2] : Module_init {Junzhi Chen} {120090777}
[ 3139.012533] [program2] : Module_init create kthread start
[ 3139.013091] [program2] : module_init kthread starts
[ 3139.013524] [program2] : The child process has pid =18078
[ 3139.013525] [program2] : This is the parent process, pid=18077
[ 3139.013803] [program2] : child process
[ 3139.014868] [program2] : get SIGTERM signal
[ 3139.014869] [program2] : child process is terminated
[ 3139.014870] [program2] : The return signal is 15
[ 3146.014970] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 1682.296145] [program2] : Module_init {Junzhi Chen} {120090777}
[ 1682.296147] [program2] : Module_init create kthread start
[ 1682.296763] [program2] : module_init kthread starts
[ 1682.297035] [program2] : The child process has pid =9366
[ 1682.297036] [program2] : This is the parent process, pid=9365
[ 1682.297381] [program2] : child process
[ 1682.452961] [program2] : get SIGFPE signal
[ 1682.452963] [program2] : child process terminated
[ 1682.452964] [program2] : The return signal is 8
[ 1692.977258] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 3213.343685] [program2] : Module_init {Junzhi Chen} {120090777}
[ 3213.343686] [program2] : Module_init create kthread start
[ 3213.344690] [program2] : module_init kthread starts
[ 3213.344982] [program2] : The child process has pid =18791
[ 3213.344983] [program2] : This is the parent process, pid=18790
[ 3213.345233] [program2] : child process
[ 3213.483609] [program2] : get SIGTRAP signal
[ 3213.483610] [program2] : child process has trap error
[ 3213.483611] [program2] : The return signal is 5
[ 3216.242687] [program2] : Module_exit./my
vagrant@csc3150:~/csc3150/HW1/source/program2$
```

```
vagrant@csc3150:~/csc3150/HW1/source/program2$ dmesg | tail -n 10
[ 1905.336719] [program2] : Module_init {Junzhi Chen} {120090777}
[ 1905.336721] [program2] : Module_init create kthread start
[ 1905.337235] [program2] : module_init kthread starts
[ 1905.338160] [program2] : The child process has pid =10112
[ 1905.338161] [program2] : This is the parent process, pid=10110
[ 1905.338477] [program2] : child process
[ 1905.339539] [program2] : get SIGHUP signal
[ 1905.339540] [program2] : child process is hung up
[ 1905.339540] [program2] : The return signal is 1
[ 1907.976888] [program2] : Module_exit./my
```

What did I learn from the tasks:

1. How to use kernel API like do_wait(), do_execve() and so on.
2. Have a deeper understanding of how OS execute the process
3. Environment setting is the most difficult part for me. I first finish my program1 in my M1 Mac but fail to compile the kernel for the program2. I borrow another Windows computer to finish my program2(This is the reason why my output screen shot looks different in program1 and program2). Through compiling kernel, I practice my skills in using linux command line and knows my computer better.
4. When learning the usage of the kernel API, I have seen some source code from kernel. This is the first time I learn to work with others code to implement my program.
5. Understand the concept of kernel, fork, parent process and child process deeper.