

陈飞颺  
120090590

# Report

## How to design

### Task1

Receive a file name from user then execute the specified program. Note to compile all programs first.

After the fork, the parent and child execute different works due to the "if".

### Task2

I need to implement a similar function my\_fork() like the system call fork(). I create a new thread. I use kernel\_clone to create a new child process and use do\_execve() to make the child process execute specified function(my\_exec()). After the child terminates, the parent receive a signal and print it out.

## How to set up development environment

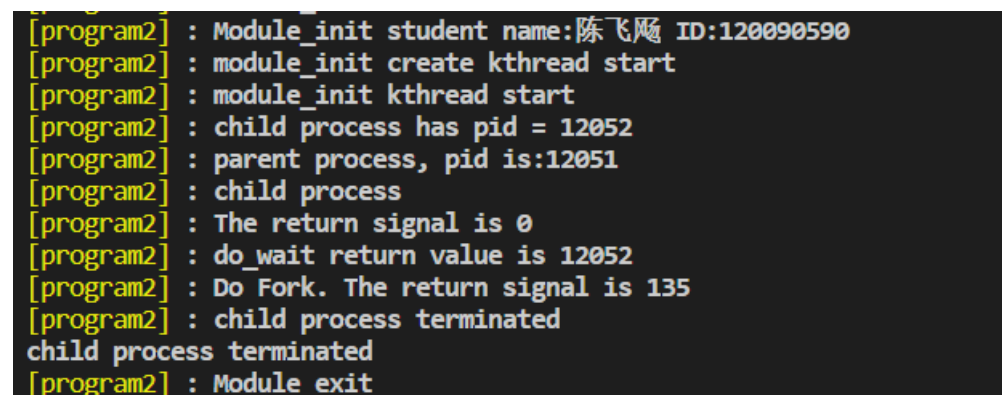
There are several main points.

1. for task1, I only need to deal with C compiler.
2. for task2, I need to modify several files in kernel source code like exit.c and exec.c. I need to add EXPORT\_SYMBOL(function\_name) after them and declare extern function\_name() in program2.c

The location of headers is a little strange. There are some headers under /usr/bin/linux. But actually the whole structure is in where I unzip the linux kernel source code.

To compile the kernel, I need to type make in the bash due to the given makefile. Then I will install the modules and the kernel to the virtual machine( the .ko and .o files). The reboot will update the kernel. Use uname -r to test the result.

## Screenshot



```
[program2] : Module_init student name:陈飞颺 ID:120090590
[program2] : module_init create kthread start
[program2] : module_init kthread start
[program2] : child process has pid = 12052
[program2] : parent process, pid is:12051
[program2] : child process
[program2] : The return signal is 0
[program2] : do_wait return value is 12052
[program2] : Do Fork. The return signal is 135
[program2] : child process terminated
child process terminated
[program2] : Module_exit
```

## What did I learn

1. How to read linux kernel source code. Understand the progress of building an OS.
2. How to compile and manipulate files in linux system. Learn to use makefiles and bash shell. Know the details of compilation.
3. The difference of kernel mode and user mode. In kernel mode we can only use some simpler functions (a lower level of system call). We can operate threads and processes directly. We will deal with signals ourselves. Our daily programs are usually running in user mode.