Operating Systems: Lab 1 preparation report

January 25, 2009

Please answer the following questions in a `.txt` file, and submit the file to the fire system.

Preparing the working space

Please dedicate a directory for this assignment and download all the source program files (see `www.ce.chalmers.se/edu/course/EDA092/lab/`).

Question 1

What do the flags `-Wall -g` mean for `gcc`?

The man pages

Please study the manual pages of the following system calls: `fork, clone, creat, open, close, wait, stat, signal, pipe, dup, exit, execvp, chdir`.

Question 2

For each of the system calls above briefly describe its function, its arguments, the required include files, and all possible return values.

Experiment with the following lines of code. What happens if you put some code after Point #1?

```
pid_t pid;
if ( (pid = fork()) == -1)
/* react to the reported error */
else if (pid == 0)
/* POINT #1:  Insert some code here */
```

Question 3

Study the following program. What does it do?

```
#include <stdio.h>
#include <signal.h>

void  sigtest(int sig) {
    if(sig == SIGINT)
        /* POINT #1: User pressed a key
        combination, but which one? */
    else if(sig == SIGTSTP)
       /* POINT #2: And here? */
    else
        /* All other signals go here. */
}

int  main(void) {
     signal(SIGINT, sigtest);
     while (1)
          pause();
}
```

Which key combinations are being handled in Points #1 and #2 respectively? Are indeed both key combinations handled right now? If your answer is yes, explain why. If your answer is no, explain how you can fix it.

Question 4

The following program creates a child process and establishes a pipe between parent and child processes.

```
#include <unistd.h>
#include <stdio.h>

#define MAXLINE 100

int main(void){
   char line[MAXLINE];
   int p[2], pid;

   pipe(p);

   switch(pid = fork()){
   case -1:
```

```
            fprintf(stderr, "fork error \n");
            exit(2);
      case 0:  /* child code */
            close(p[0]);
            write(p[1], "Hello World\n", 12);
            break;
      default:   /* parent code */
            close(p[1]);
            read(p[0],line,MAXLINE);
            wait(NULL);
      }

      exit(0);
}
```

What happens in the parent and child processes? How do they interact? What is the meaning of the `close(p[0])` and `close(p[1])` commands?

Resource hygiene

On Unix computer operating systems, a zombie process or defunct process is a process that has completed execution but still has an entry in the process table, this entry being still needed to allow the process that started the zombie process to read its exit status.

Question 5

- Describe the output of the following command and the scenarios for which the output is not empty.

  ```
  % ( ps aux | grep defunct )
  ```

- Write a command that kills a `defunct` or zombie process. When should we use such a command? Please explain what the following command does.

  ```
  % ps aux | grep -v "^root" | wc
  -l
  ```

- Read the page `help ulimit`, and explain its function. Please design a technique that restricts processes from executing programs that proliferate zombie. Can you extend that technique to other resources?