OS HW1

Operation system 108 fall

W.J. TSAI 蔡文錦 教授

TA 潘璿任 游依杰 簡育聲 石瑾旋

PREWORK

Login Tools

- PuTTY

Editors

- vim

FTP Tools

- FileZilla Client

PuTTY

Download PuTTY

https://goo.gl/rM4Scb

Alternative binary files

The installer packages above will provide all of these (except PuTTYtel), but you can download

(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

putty.exe (the SSH and Telnet client itself)

32-bit:	putty.exe	(or by FTP)	(signature)
64-bit:	putty.exe	(or by FTP)	(signature)

PuTTY

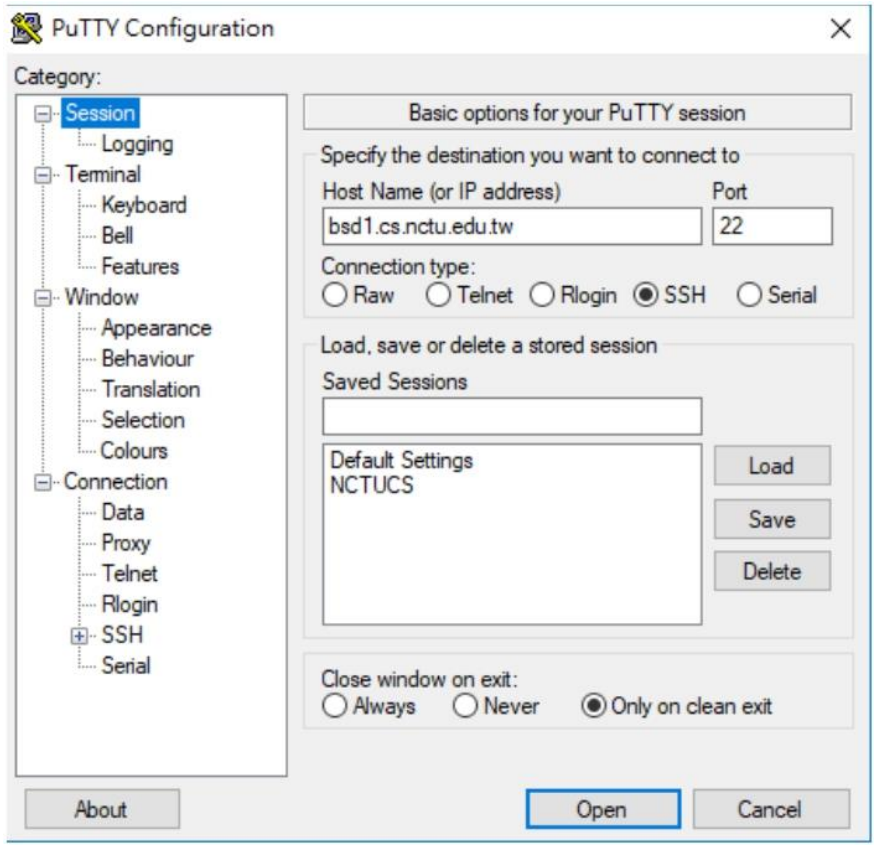
How to Use PuTTY

https://goo.gl/8AJsPL

Login

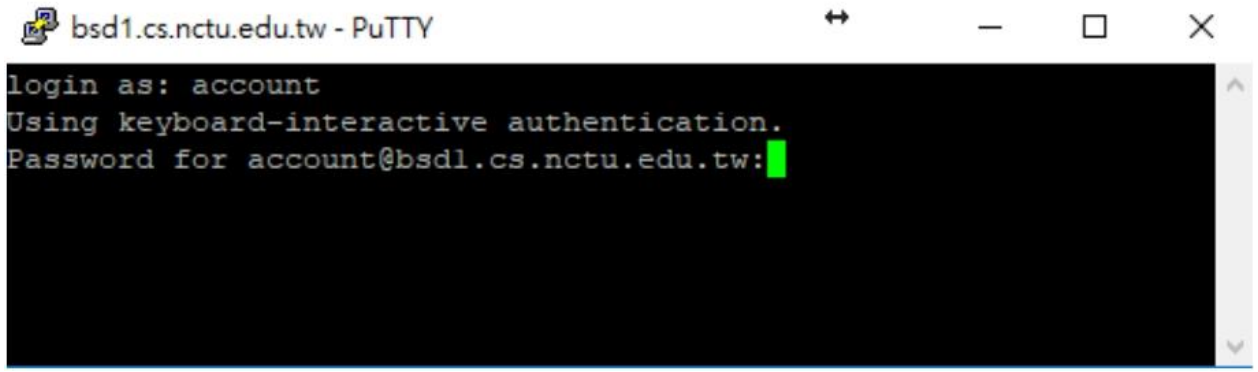
The default for SSH service is port 22

- bsd1.cs.nctu.edu.tw – bsd5.cs.nctu.edu.tw
- linux1.cs.nctu.edu.tw – linux6.cs.nctu.edu.tw



PuTTY

Command



- clear – clear the screen
- ls – list directory contents
- mv – move files or directories
- mkdir – create directories
- rm – remove files or directories
- chmod – change file system modes of files or directories

- Other instruction Reference
  - http://linux.vbird.org/linux_basic/redhat6.1/linux_06command.php#filesystem

FileZilla

- Upload File to Workstation
- Login

主機: bsd1.cs.nctu.edu.tw

協定: SFTP

登入型態: 一般

使用者: 計中申請帳號

密碼: 計中申請密碼

HW 1-1

Finish “hw1_1.c” in order to design a C program to serve as a shell interface that accepts user commands then execute each command in a separate process.

UNIX shells typically allow the child process to run in the background or concurrently, so if a ampersand(&) at the end of the command means the parent and child processes will run concurrently.

Important System Call:

read(STDIN_FILENO, inputBuffer, MAX_LINE): read command line

fork(): create child process

execvp(char *command, char *params[]): execute system calls

waitpid()

…

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

#defile MAX_LINE 80

int main(void)
{
    char *arg[MAX_LINE/2+1]; /*command line arguments*/
    int should_run = 1; /*flag to determine when to exit program*/

    while(should_run){
        print("osh>");
        fflush(stdout);

        /**
        * your code!
        * After reading user input, the step are:
        * (1) fork a child process using fork()
        * (2) the child process will invoke execvp()
        * (3) if command included &, parent will invoke wait()
        */
    }

    return 0;
}
```

HW 1-1

- Change directory

  $cd your/folder/

- Compile

  $gcc -o shell hw1_1.c

- Execute

  $./shell

- **You need**

1. finish "hw1_1.c" as a shell interface.

2. user can keep entering the command until he/she enters "exit".(a command include the command itself and its parameters).

3. if a user enter "&", the shell should let child run in the background (means child and parent run concurrently).

4.Your shell needs to support following commands: cat, ls, date, ps –f, ps –f &, exit(you can refer to pages 9, 10, 11).

Example

Show the content of hi.txt file

Show the file in the directory

Show the date

List all processes in current shell

Perform a full-format processes list

Enter “exit” to finish shell

bsd1.cs.nctu.edu.tw - PuTTY

```
bsd1 [/u/gcs/106/0656649/OS] -hjpan91- ./shell
osh>cat hi.txt
1234osh>ls
hi.txt    hw1_1.c   hw1_1.cpp hw1_3.cpp shell
osh>date
Tue Sep 24 15:46:11 CST 2019
osh>ps
  PID TT  STAT    TIME COMMAND
94548  7  Ss   0:00.04 /bin/tcsh -l
94619  7  S+   0:00.01 ./shell
94623  7  R+   0:00.00 ps
osh>ps -f
  PID TT  STAT    TIME COMMAND
94548  7  Ss   0:00.04 /bin/tcsh -l
94619  7  S+   0:00.01 ./shell
94624  7  R+   0:00.00 ps -f
osh>exit
bsd1 [/u/gcs/106/0656649/OS] -hjpan91-
```

Example

bsd1.cs.nctu.edu.tw - PuTTY

```
bsd1 [/u/gcs/106/0656649/OS] -hjpan91- ./shell
osh>ps -ael
  UID   PID  PPID CPU PRI NI     VSZ   RSS MWCHAN STAT TT     TIME COMMAND
    0   535     1   0  20  0    6412  1604 ttyin  Is+  v0  0:00.00  /usr/libexec/get
65534   716     1   0  20  0    6352  2136 accept I    v0- 0:00.05  /usr/local/etc/p
    0   811     1   0  52  0    6412  2196 ttyin  Is+  v1  0:00.00  /usr/libexec/get
    0   812     1   0  52  0    6412  2196 ttyin  Is+  v2  0:00.00  /usr/libexec/get
    0   813     1   0  52  0    6412  2196 ttyin  Is+  v3  0:00.01  /usr/libexec/get
    0   814     1   0  52  0    6412  2196 ttyin  Is+  v4  0:00.00  /usr/libexec/get
    0   815     1   0  52  0    6412  2196 ttyin  Is+  v5  0:00.00  /usr/libexec/get
    0   816     1   0  52  0    6412  2196 ttyin  Is+  v6  0:00.00  /usr/libexec/get
    0   817     1   0  52  0    6412  2196 ttyin  Is+  v7  0:00.00  /usr/libexec/get
16287 42957 42956   0  20  0    7864  3728 pause  Is    0  0:00.42  /bin/tcsh -l
16287 95047 42957   0  20  0   22748 15188 select I+    0  0:02.06  mutt
16287 94361 94360   0  23  0    7864  3616 ttyin  Is+   1  0:00.35  /bin/tcsh -l
14301 58772 58771   0  20  0    7544  3976 ttyin  Is+   2  0:00.60  /bin/csh -l
18601 89028 89027   0  20  0    7544  4160 ttyin  Is+   3  0:00.34  /bin/tcsh -l
14301 72224 72222   0  20  0    7544  3308 ttyin  Is+   5  0:00.06  /bin/csh -l
10094 96657 96656   0  20  0    7544  3312 ttyin  Is+   6  0:00.64  /bin/tcsh -l
18026 94548 94547   0  21  0    7544  3980 pause  Is    7  0:00.05 SSH_CLIENT=140.11
18026 96118 94548   0  20  0    6284  1800 wait   S+    7  0:00.00 SSH_CLIENT=140.11
18026 96121 96118   0  20  0    6996  2804 -      R+    7  0:00.00 SSH_CLIENT=140.11
18601 48366 48365   0  20  0    9592  4108 ttyin  Is+   8  0:00.10  /bin/tcsh -l
15184 19780 19779   0  20  0    7544  4100 ttyin  Is+   9  0:00.01  -tcsh (tcsh)
15184 19830 19779   0  20  0    7544  4100 ttyin  Is+  10  0:00.02  -tcsh (tcsh)
12793 33243 33242   0  20  0    7544  3324 pause  Is   11  0:00.36  /bin/tcsh
12793 50730 33243   0  31  0 1486156 94328 uwait  I+   11  0:02.87  ./test2
18601 94337 94336   0  52  0    7544  4104 ttyin  Is+  13  0:00.04  /bin/tcsh -l
18601 94409 94408   0  20  0    9592  4456 pause  Is   14  0:00.26  /bin/tcsh -l
18601 95945 94409   0  20  0    7412  3880 ttyin  S+   14  0:00.07  vi hw01_b.c
15463 23726 23725   0  25  0    7544  3292 ttyin  Is+  17  0:00.01  -tcsh (tcsh)
osh>
```

Receive “-ael” as args and execute

Example

bsd1.cs.nctu.edu.tw - PuTTY

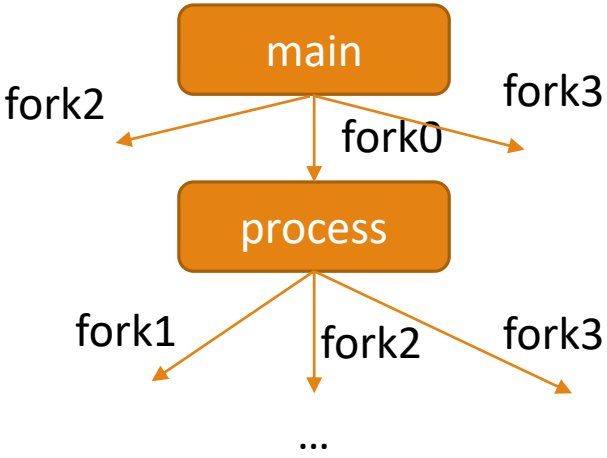
```
bsd1 [/u/gcs/106/0656649] -hjpan91- cd OS
bsd1 [/u/gcs/106/0656649/OS] -hjpan91- g++ -o shell hw1_1.cpp
bsd1 [/u/gcs/106/0656649/OS] -hjpan91- ./shell
osh>ps -f &
osh>  PID TT  STAT    TIME COMMAND
96229  4  Ss   0:00.05 /bin/tcsh -l
96320  4  S+   0:00.01 ./shell
96322  4  R+   0:00.00 ps -f
ps -f &
osh>  PID TT  STAT    TIME COMMAND
96229  4  Ss   0:00.05 /bin/tcsh -l
96320  4  S+   0:00.01 ./shell
96322  4  Z+   0:00.00 <defunct>
96323  4  R+   0:00.00 ps -f
ps -f
  PID TT  STAT    TIME COMMAND
96229  4  Ss   0:00.05 /bin/tcsh -l
96320  4  S+   0:00.01 ./shell
96322  4  Z+   0:00.00 <defunct>
96323  4  Z+   0:00.00 <defunct>
96324  4  R+   0:00.00 ps -f
osh>
```

PID 96322 becomes a zombie (because ps –f **&** will let child process and parent process run concurrently, meaning that the parent process didn't call "wait" for the child)

HW 1-2

Please draw the tree format according the code on the report(OS_document.docx).

You need to clarify which fork(fork0, fork1, fork2 or fork3) the process been made by, for instance:



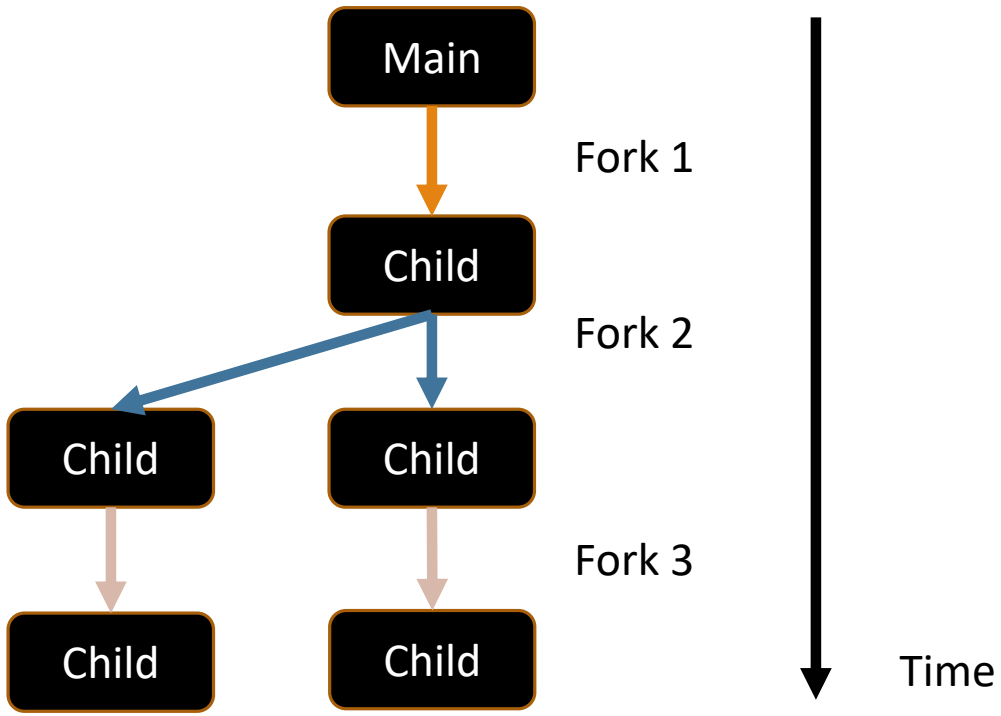

```
#include <stdio.h>
#include <unistd.h>

using namespace std;

int main()
{
    pid_t pid;
    pid = fork(); //fork0
    for(int i=0;i<2;i++)
    {
        if(pid==0)
        {
            pid = fork(); //fork1
        }
        else if(pid>0)
        {
            pid = fork(); //fork2
        }
        else
        {
            printf("Error!");
        }
    }
    pid = fork(); //fork3
    return 0;
}
```

HW 1-3

Write a program which uses fork() to produce the following tree format.



HW 1-3

Your program must output messages in the format as below

Main Process ID

Total 5 Children

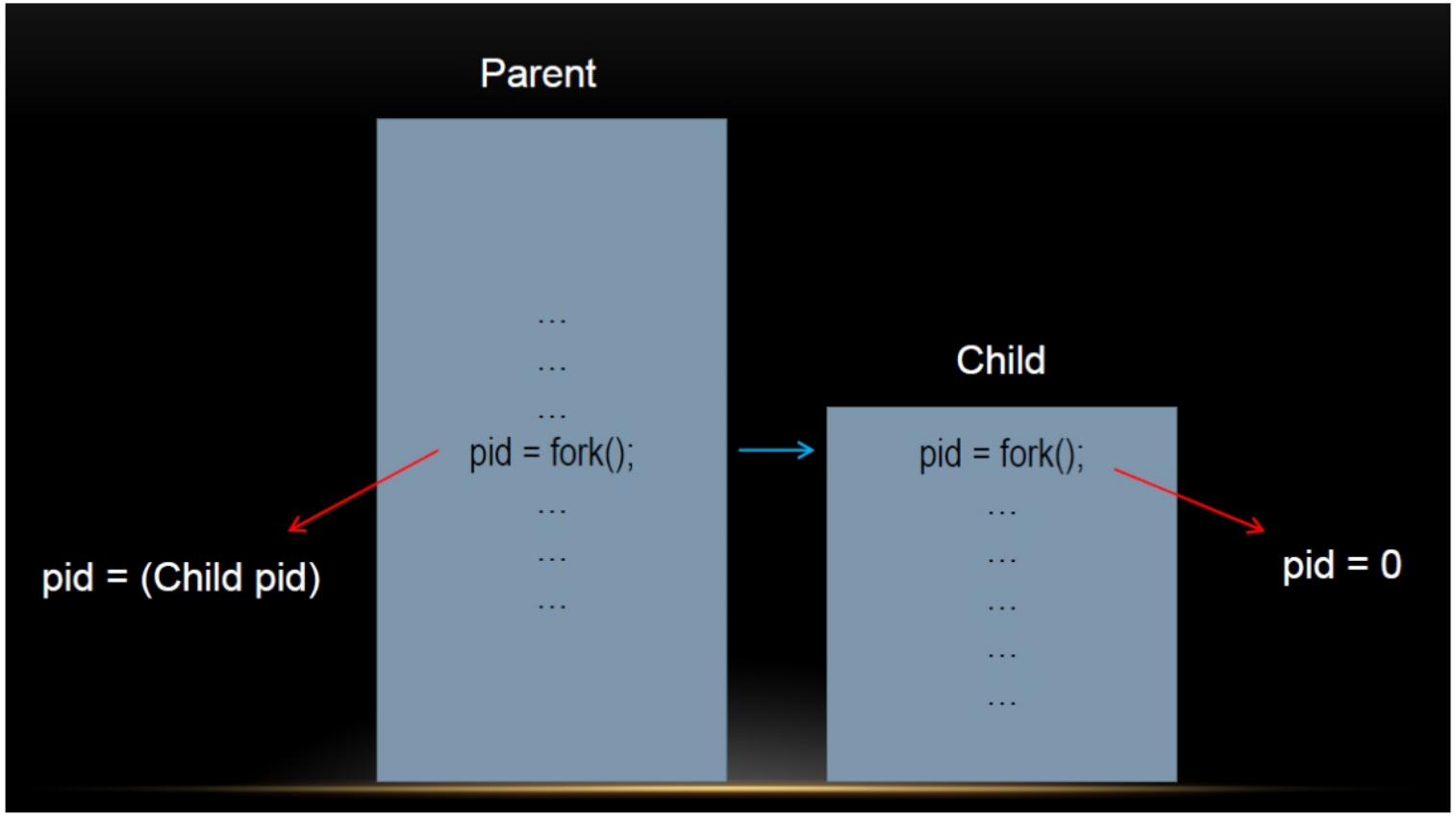
The format & fork order have to be same.
(Fork 1 2 2 3 3)

```
16:07          @bsd1 [~/106_OS/hw1] >./a.out
Main Process ID : 29428

Fork 1. I'm the child 29429, my parent is 29428.
Fork 2. I'm the child 29431, my parent is 29429.
Fork 2. I'm the child 29432, my parent is 29429.
Fork 3. I'm the child 29433, my parent is 29432.
Fork 3. I'm the child 29434, my parent is 29431.
```

Hint:
Parent Process has to wait until Child Process finish, then exit.

Hint



Submission and Grade

Filename format please according : hw1-1.c, hw1-3.c (or .cpp), OS_report.docx.
Put two *.c(*.cpp) files and a *.docx report into same compressed file named **StudentID_hw1.zip** (ex : 0000000_hw1.zip).

Deadline: 2019/10/13 (SUN) PM11:59

a. Total score: 100pts. **COPY WILL GET A 0 POINT!**

b. hw1-1 score: code 40pts, report Q1 10pts

c. hw1-2 score: report Q2 20pts

d. hw1-3 score: code 20pts, report Q3 10pts

e. Report: format is in OS_report.docx. YOU NEED TO FINISH EVERY PART OF REPORT TO GET SCORE!

Rules

0. Use NCTU CS Workstation as your programming environment
1. Use only C/C++, OTHER LANGUAGES WILL GET 0 POINT!
2. **Incorrect filename format** will get -5 pts
3. **Incorrect output format** will get -5 pts
4. DELAYED SUBMISSION WILL GET 0 POINT!

*If you have any question, just send email to Tas by new E3.