

OS HW1

Operation system 107 fall

W.J. TSAI 蔡文錦 教授

TA 蘇聖雅 莊侑穎 劉晏 盧彥廷 黃資捷

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

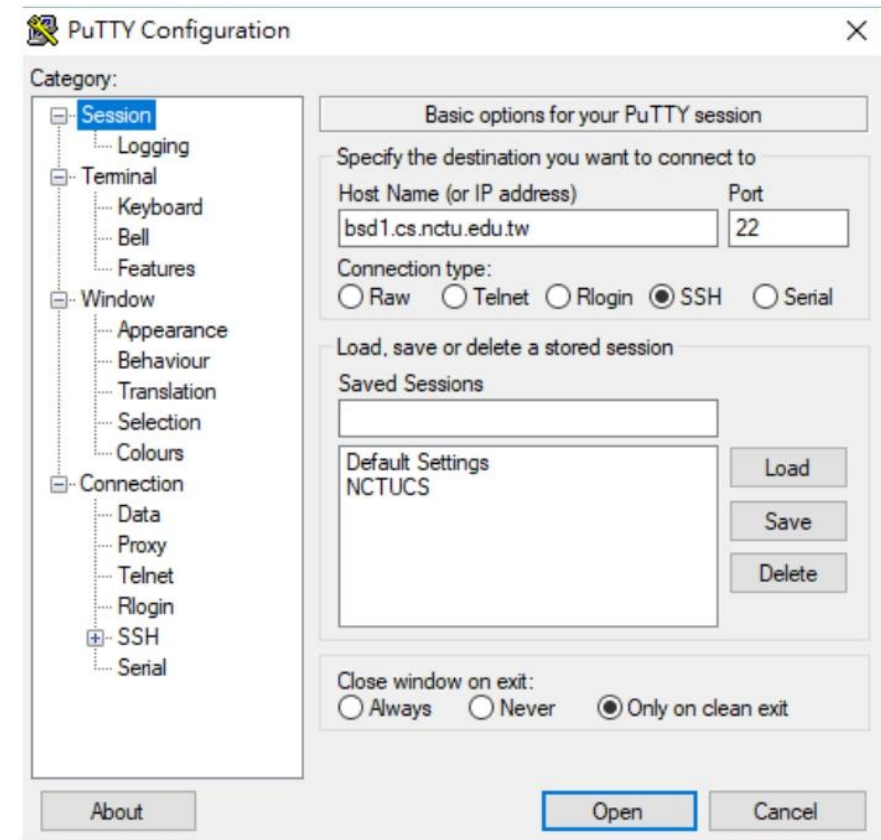
How to Use PuTTY

<https://goo.gl/8AJsPL>

Login

The default for SSH service is port 22

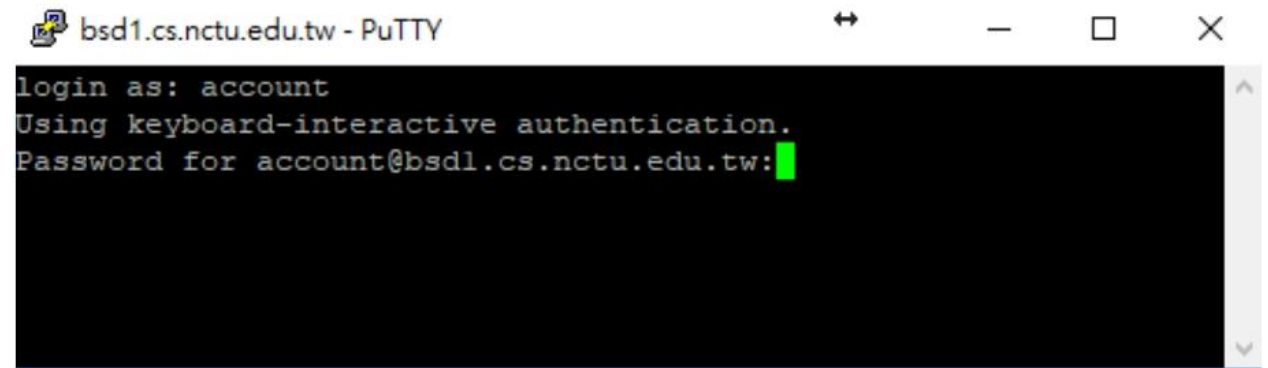
- bsd1.cs.nctu.edu.tw – bsd5.cs.nctu.edu.tw
- linux1.cs.nctu.edu.tw – linux6.cs.nctu.edu.tw



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

A screenshot of a PuTTY terminal window. The title bar reads "bsd1.cs.nctu.edu.tw - PuTTY". The terminal content shows a login sequence: "login as: account", "Using keyboard-interactive authentication.", and "Password for account@bsd1.cs.nctu.edu.tw:" followed by a green cursor. The terminal has a black background and white text. The window includes standard PuTTY controls like a double arrow icon, a minus icon, a square icon, and a close 'X' icon.

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.

You will use:

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

`fork()`: create child process

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

`wait()`

...

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

#define 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("qsh>");
        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).

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 [/u/gcs/106/0656613] -sysul205- ./shell
osh>cat hi.txt
1234osh>ls
      hi.txt      shell      simple-shell.c ssh
osh>date
Fri Sep 21 21:42:08 CST 2018
osh>ps
  PID TT  STAT   TIME COMMAND
35914  9   Is    0:00.05 /bin/tcsh -l
36059  9   S+    0:00.00 ./shell
36065  9   R+    0:00.00 ps
osh>ps -f
  PID TT  STAT   TIME COMMAND
35914  9   Is    0:00.05 /bin/tcsh -l
36059  9   S+    0:00.00 ./shell
36090  9   R+    0:00.00 ps -f
osh>exit
bsd1 [/u/gcs/106/0656613] -sysul205- █
```

Example

```
bsd1 [/u/gcs/106/0656613] -sysul205- ./shell
osh>ps -ael
```

UID	PID	PPID	CPU	PRI	NI	VSZ	RSS	MWCHAN	STAT	TT	TIME	COMMAND
65534	704	1	0	20	0	6304	2084	accept	I	v0-	0:00.01	/usr/local/et
0	797	1	0	52	0	6412	2244	ttyin	Is+	v0	0:00.00	/usr/libexec/
0	798	1	0	52	0	6412	2244	ttyin	Is+	v1	0:00.00	/usr/libexec/
0	799	1	0	52	0	6412	2244	ttyin	Is+	v2	0:00.00	/usr/libexec/
0	800	1	0	52	0	6412	2244	ttyin	Is+	v3	0:00.00	/usr/libexec/
0	801	1	0	52	0	6412	2244	ttyin	Is+	v4	0:00.00	/usr/libexec/
0	802	1	0	52	0	6412	2244	ttyin	Is+	v5	0:00.00	/usr/libexec/
0	803	1	0	52	0	6412	2244	ttyin	Is+	v6	0:00.00	/usr/libexec/
0	804	1	0	52	0	6412	2244	ttyin	Is+	v7	0:00.00	/usr/libexec/
16287	34602	34598	0	20	0	13960	6100	ttyin	Is+	0	0:00.07	/bin/tcsh -l
16911	77094	77093	0	20	0	13640	5872	ttyin	Is+	1	0:00.06	/bin/tcsh -l
14274	31611	31610	0	52	0	17112	10000	ttyin	Is+	2	0:01.29	/bin/zsh -l
16990	31487	31486	0	52	0	13640	5812	ttyin	Is+	3	0:00.04	/bin/tcsh -l
13634	96468	96441	0	20	0	13640	6252	ttyin	Is+	4	0:00.05	/bin/tcsh -l
16990	31726	31725	0	22	0	13640	5804	ttyin	Is+	5	0:00.02	-tcsh (tcsh)
15184	52240	52239	0	21	0	13640	5888	pause	Is	7	0:00.03	-tcsh (tcsh)
15184	52243	52240	0	20	0	16500	8576	select	I+	7	0:00.07	ssh csduty
15184	52358	52239	0	20	0	13640	5892	pause	Is	8	0:00.04	-tcsh (tcsh)
15184	53000	52358	0	20	0	16500	8476	select	I+	8	0:00.03	ssh csduty
15197	35914	35913	0	20	0	13640	5820	pause	Ss	9	0:00.05	SSH_CLIENT=116
15197	36176	35914	0	20	0	6284	2080	wait	S+	9	0:00.00	SSH_CLIENT=116

Receive “-ael” as args and execute

Example

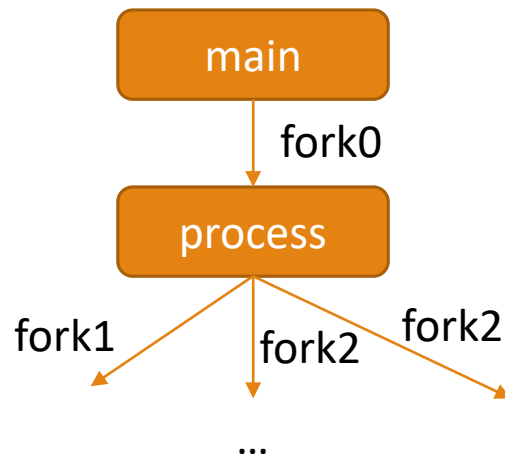
PID 36227 becomes a zombie
(because `ps -f &` will let child
process and parent process run
concurrently, meaning that the
parent process didn't call "wait"
for the child)

```
bsd1.cs.nctu.edu.tw - PuTTY
bsd1 [/u/gcs/106/0656613] -sysul205- ./shell
osh>ps -f &
osh> PID TT STAT TIME COMMAND
35914 9 Ss 0:00.05 /bin/tcsh -l
36226 9 S+ 0:00.00 ./shell
36227 9 R+ 0:00.00 ps -f
ps -f &
osh> PID TT STAT TIME COMMAND
35914 9 Ss 0:00.05 /bin/tcsh -l
36226 9 S+ 0:00.00 ./shell
36227 9 Z+ 0:00.00 <defunct>
36228 9 R+ 0:00.00 ps -f
ps -f
PID TT STAT TIME COMMAND
35914 9 Is 0:00.05 /bin/tcsh -l
36226 9 S+ 0:00.00 ./shell
36229 9 R+ 0:00.00 ps -f
osh>
```

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:

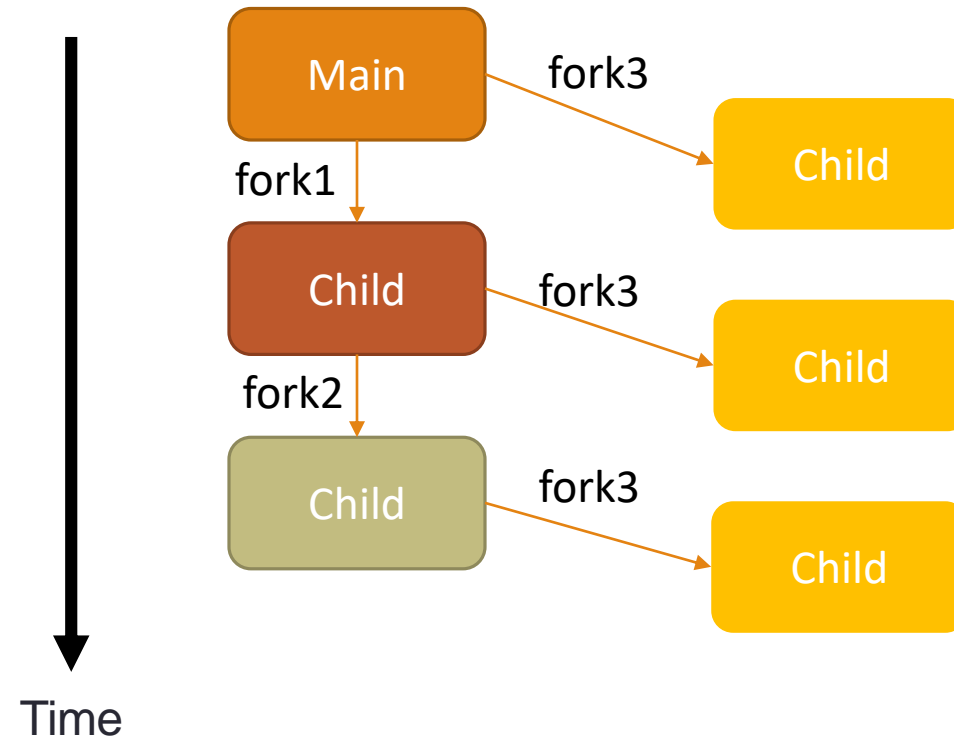


```
#include <stdio.h>
#include <unistd.h>

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

HW 1-3

Write a program which uses `fork()` to produce the following tree format (namely, your code should have only 3 `fork()`)



HW 1-3

Output format

Main process
Total 5 Children
The **format & fork order**
have to be same.
(Fork 1 2 3 3 3)

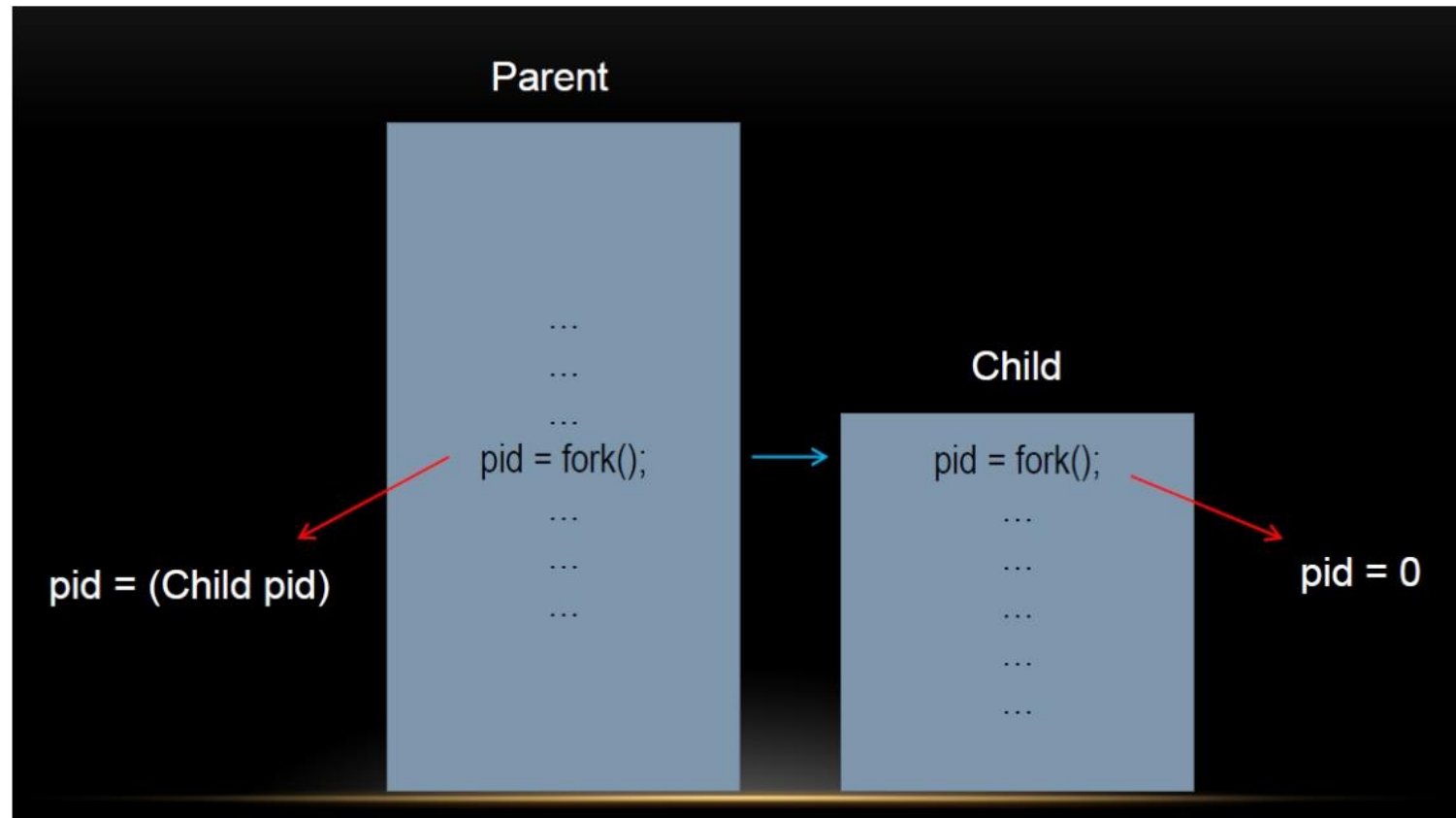
```
bsd1.cs.nctu.edu.tw - PuTTY
bsd1 [/u/gcs/106/0656613] -sysul205- ./hw1_3
Main process id : 40513.
Fork1, I'm the child 40514, my parent is 40513.
Fork2, I'm the child 40515, my parent is 40514.
Fork3, I'm the child 40516, my parent is 40515.
Fork3, I'm the child 40517, my parent is 40514.
Fork3, I'm the child 40518, my parent is 40513.
bsd1 [/u/gcs/106/0656613] -sysul205- █
```

Hint:

Parent Process has to wait until **Child Process** finishes, then **exit**.

Use PID to identify parent and child.

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 : 00000000_hw1.zip).

Deadline: 2018/10/14 (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.