Faculty of Computer Science & Engineering

Operating Systems

Nguyen Minh Tri
nmtribk@hcmut.edu.vn
302-B9

Ho Chi Minh City
University of Technology

Course overview

- What we will learn?
    - System Programming Techniques
    - Concurrency
    - Synchronization
    - Communication
    - Scheduling
    - Memory Management
- Environment: *nix systems (CentOS, Ubuntu, Mac OS?)

Assessments

- Assessments
  - Exams: 50%
  - Assignments: 30%
  - Lab works + Exercise: 20%

About me

- Contact:
  - Room 302B9, Main Campus, HCMUT
  - Room 710H6 (HPC Center), Linh Trung Campus, HCMUT
- Email:
  - nmtribk@hcmut.edu.vn
- Research interests:
  - High Performance Computing
  - Distributed Systems
  - Machine learning, deep learning, Data analysis…

Introduction to *nix OS

Kernel

- A kernel is a program that allocates and controls hardware resources in a system
- Note: Linux is a kernel, not an Operating System
- Linux Distributions (RedHat, Fedora, Debian, etc.) are operating system made from a software collection which based upon the Linux kernel and a package of management system (often GNU utilities)

Shell

- Shell is a command-line interpreter that allows users to direct the operation of the computer by entering commands as text.
- The most popular Shell today in *nix OS is bash (Bourne Again SHell)
- Other shells: Shell C (csh), Shell Korn (ksh), zsh, etc.
- Syntax: <command> <option> <argument>
- Exercise:
  - date
  - clear
  - echo hello, world!
  - man date

File System

- Everything in *nix systems is file.
- *nix uses an hierarchical, unified file system. Root (/) is the parent of all files
- File name is unique and described by the path from root
    - /home, /bin, /boot/, /etc …
- Exercise: Specify the path to /phd

File System

- Some special notations:
  - "." Working (or current) directory
  - ".." Parent directory of current directory
  - "~" Home directory
- Exercise: run and guess the functionality of following commands
  - pwd
  - ls
  - ls –a
  - ls --help
  - cd ..
  - cd /

File System

- Other useful commands
  - mkdir : Create new directory
  - mv : Move or change the name of a file (?)
  - cp : Copy file
  - rm : Remove file
  - rmdir : Remove empty directory

File System

- Wild cards: used as a substitute for any of a class of characters
  - * represent a group of characters including null.
  - ? only one characters
  - [..] range matching

Users

- Each user has his own identifier consisting of
  - UID (user ID): username
  - GID (group ID): the group in which user belongs to
- Get information about current user: type id
- Exercise:
  - who
  - whoami

Permission

- Each file belongs to only one user. Owner of a file has the right to allow or prevent other users from accessing, changing the content or executing his/her files.
- Three basic operations on files
  - Read (r): read a file; list file in directory
  - Write (w): write on file; create, rename, delete files in a directory
  - Execution (x): file can be executed; run execution file in a directory, read, write in a directory
- Permission are granted to 3 classes:
  - Owner of the file
  - Group of owner
  - Other (users)

Permission

- Permission of a file is represented by 9 bits:
  - First 3 bits: Owner permission
  - Next 3 bits: Group permission
  - Last 3 bits: Other permission
- In each of 3-bit group:
  - First bit: read permission
  - Second bit: write permission
  - Last bit: execution permission
- Using ls -l to see permission of files in a directory:
  - w: file can be written
  - r: file can be read
  - x: file can be executed
  - -: specific permission has not been assigned

Redirection

- Data direction could be treat as stream of characters. *nix systems have three standard input/output streams:
  - stdin: standard input, often comes from keyboard
  - stdout: standard output, often comes to screen
  - stderr: standard error output, often comes to screen
- Standard I/O direction could be redirected by using operators:
  - <   Redirect input direction
  - >   Redirect output direction
  - >>   Redirect output direction and append the output data to existing file (instead of clear the old content)

Pipe

- *nix systems allow data stream to go through multiple process for making efficient execution.
- Data go through processes in a pipe, the output of a process is the input of another.
- We use operator " | " to create a pipe which make data flow from the process on its left side to the process on its right side.
- Example:
    - ls -l /etc | grep "sys" | wc -l
- Exercise: explain the meaning of the command above.

Make File

- The *makefile* directs *make* on how to compile and link a program.
- When a source file is changed, it must be recompiled. If a file has changed, each source file that depend on this file must be recompiled to be safe.
- Rules:
  - target: dependencies

    system command(s)

Make File

- Example:

```
hello: main.o factorial.o hello.o
    g++ main.o factorial.o hello.o -o hello
main.o: main.cpp functions.h
    g++ -c main.cpp
factorial.o: factorial.cpp functions.h
    g++ -c factorial.cpp
hello.o: hello.cpp functions.h
    g++ -c hello.cpp
clean:
  rm edit main.o factorial.o hello.o
```

Learning materials

- Paul Cobbaut, “Linux System Administration” (free ebook)
- Evi Nemeth el al, “UNIX and Linux System Administration Handbook”, Pearson Education, Inc., 2011
- Steve Parker, “Shell Scripting”, John Wiley & Sons, Inc., 2011
- Arnold Robbins and Nelson H. F. Beebe, “Classic Shell Scripting”, O’Reilly Media Inc., 2005

Homeworks

1. Write a script to save your name (input) and system information into text file (ex1.txt)
   - $ ./ex1.sh <your name>
   - Check null input.
2. Write simple Makefile:
   - File: main.c, sum.h, sum.c, sub.h, sub.c

End

Thanks!