Introduction to Unix

Operating Systems

- A computer system cannot function without an operating system (OS).
- There are many different operating systems available for PCs, minicomputers, and mainframes.
  - The most common ones being Windows, Mac OS, and variations of UNIX (e.g. Linux)
- UNIX is available for many different hardware platforms
  - Most other operating systems are tied to a specific hardware family.
  - This is one of the first good things about UNIX.

Unix Overview

- UNIX allows more than one user to use the computer system at a time
  - a desirable feature for business systems.
- Power and Flexibility
  - Small number of basic elements that can be combined in an infinite variety of ways to suit the application
- Consistency in commands:
  - ls A* means list files beginning with ‘A’
  - rm A* means remove files beginning with ‘A’
- Majority of users are engaged in software development

Unix and Linux

- Late 1980s – two version of UNIX
  - 4.3BSD
  - System V Release 3
- Standardizing effort – POSIX (Portable Operating System)
  - POSIX committee produced standard 1003.1 by taking the intersection of System V and BSD
  - Set of library procedures – open, read, fork, ...
- Linux – Unix clone originally running on PC
  - Recent Version of Linux has features of a modern UNIX OS
- Unix and Linux have been adopted in industrial-strength business applications

Unix and Shell

- Mac OS and Windows - Graphical User Interface
- UNIX – command line interface, called *shell*
  - UNIX Graphical environment – X Windows

Logging In

- To use a UNIX system, you first log in with a username
  - a unique name that distinguishes you from the other users of the system.
- UNIX first asks you for your username by prompting you with the line “login:” and then asks for your password.
  - Depending on how your system is set up, you should then see either a $ or a > prompt.
- UNIX is case sensitive, so make sure that the case of the letters is matched exactly.

```
UNIX(r) System V Release 4.0
login: glass
Password:                 ...what I typed here is secret and doesn't show
Last login: Sun Feb 15 18:33:26 from dialin
cuse93:>
```

Shells

- When you first log into UNIX, you will see a prompt such as > displayed by the *shell*
- Shell is a program that acts as a middleman between you and the raw UNIX operating system.
- A shell lets you run programs, builds pipelines of processes, saves output to files, and runs more than one program at the same time.

Running A Utility

- To run a utility, simply enter its name and press the *Enter* key
- One sample utility is **date** which displays the current date and time

```
cuse93:> date                    ... run the date utility
Thu Mar 12 10:41:50 MST 1998
cuse93:>
```

Unix Utility Programs

- Large number of utility programs
- Divided into six categories
  - File and directory manipulation commands
    - Example: cp a b | ls *.*
  - Program development tools such as editors/compilers
    - Example: gcc (C compiler), make (maintain large programs whose source code consists of multiple files)
  - Text editing and processing
    - Example: pico (nano in Linux), emacs, vim, vi
  - Filters
    - Example: grep (extracts lines containing patterns), cut, paste,
  - System administration
    - Example: mount (mount file system)
  - Miscellaneous
    - Example: kill 1325 (kill a process), chmod (change privileges of a file)

Unix Utility Programs

Program	Typical use
cat	Concatenate multiple files to standard output
chmod	Change file protection mode
cp	Copy one or more files
cut	Cut columns of text from a file
grep	Search a file for some pattern
head	Extract the first lines of a file
ls	List directory
make	Compile files to build a binary
mkdir	Make a directory
od	Octal dump a file
paste	Paste columns of text into a file
pr	Format a file for printing
rm	Remove one or more files
rmdir	Remove a directory
sort	Sort a file of lines alphabetically
tail	Extract the last lines of a file
tr	Translate between character sets

Logging Out

- To leave the UNIX system, type the keyboard sequence *Control-D* at your shell prompt.
  - This tells your login shell that there is no more input for it to process, causing it to disconnect you from the UNIX system.
- Most systems then display a prompt and wait for another user to log in.

  Here's an example:

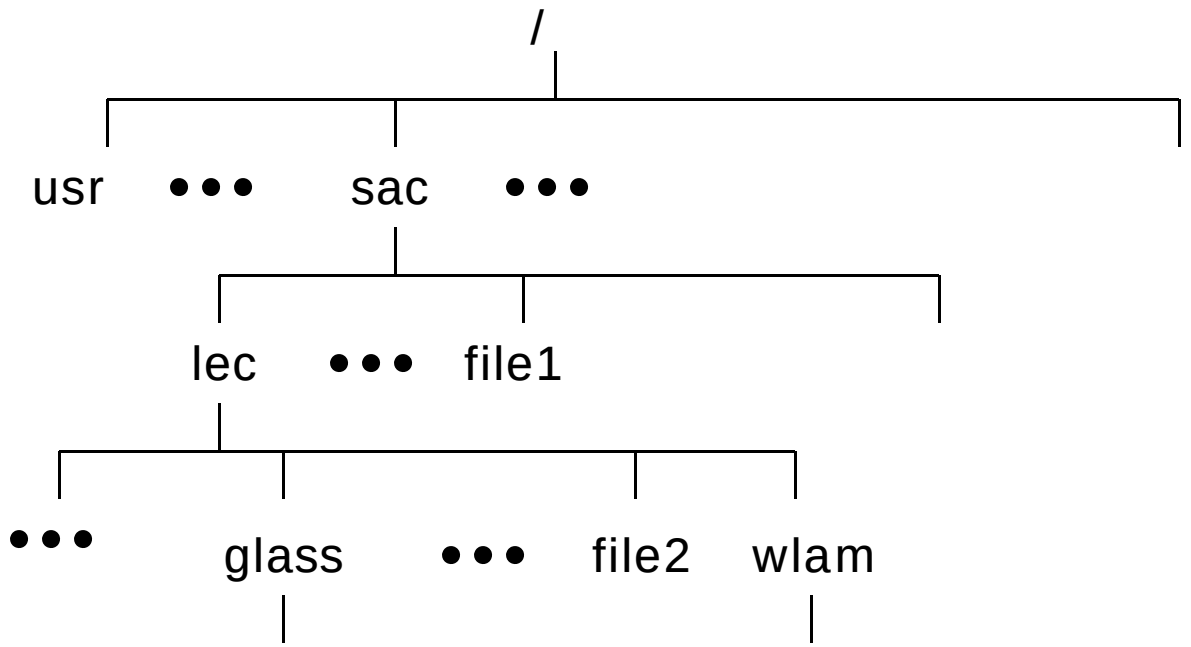
  ```
  cuse93:> ^D                    ... I'm done!
  UNIX(r) System V Release 4.0
  login:          ...wait for another user to log in.
  ```
- Note: The shell can be set to ignore ^D for logout, since you might type it by accident. In this case, you in, type the "logout" command instead.

Directory and File

- A hard disk in Unix is typically shared by different users
- Directory – similar to “folder” in Windows
- The root directory is specified by a symbol “/”
- Under a directory
  - Store files
  - Contains sub-directories

Directories

- The directories are organized in a hierarchical structure (similar to Windows)



Pathname

- Pathname -
  - A sequence of directory names that leads you through the hierarchy from a starting directory (e.g. root “/”) to a target (directory or file)
- Some sample pathnames
  - /sac/file1 – a file
  - /sac/lec – a directory
  - /sac/lec/file2 – a file

Home Directory

- When you first login, you are automatically located at a certain location of a hard disk called *home directory*
  - Every user has a different home directory
  - The home directory is set by the system administrator
  - For example, after login, the user is located at /sac/lec/glass

```
UNIX(r) System V Release 4.0
login: glass
Password:                 …what I typed here is secret and doesn't show
Last login: Sun Feb 15 18:33:26 from dialin
cuse93:/sac/lec/glass>
```

Unix - Making A Directory : mkdir

- We can use the **mkdir** utility to create a new directory.
- An example of the **mkdir** utility:
- *Utility:* **mkdir** *newDirectoryName*

```
cuse93:> mkdir reverse      ... create a directory called “reverse”
cuse93:>
```

Unix - Moving To A Directory : cd

- In general, it's a good idea to move your shell into a directory if you intend to do a lot of work there. To do this, use the **cd** command.

```
cuse93:> cd reverse          ... go to the reverse directory
```

- **cd** isn't actually a UNIX utility, but instead is an example of a shell built-in command. Your shell recognizes **cd** as a special keyword and executes it directly.
- *Shell Command:* **cd** [*directoryName*]
- The **cd** (change directory) shell command changes a shell's current working directory to *directoryName.* If the *directoryName* argument is omitted, the shell is moved to its owner's home directory.

Editing Programs: Using an Editor

- **pico (nano in Linux)** Editor
  - General Command

Write editor contents to a file	[Ctrl] o
Save the file and exit pico	[Ctrl] x
Spell Check	[Ctrl] t
Justify the text	[Ctrl] j

  - Moving around in your file

Move one character to the right	[Ctrl] f or right arrow key
Move one character to the left	[Ctrl] b or left arrow key
Move up one line	[Ctrl] p or up arrow key
Move down one line	[Ctrl] n or down arrow key

- Other editors such as **emacs** and **vim** can be used

Unix - Listing The Contents Of A Directory: ls

- We can use the **ls** utility to list the name and other information about a file or a directory.
- Suppose that we have used an editor to edit the program “reverse.c”

```
cuse93:> ls                              ... list all files in current directory
reverse.c
cuse93:>  ls -l reverse.c                ... long listing of “reverse.c”
-rw-r--r--  1  glass  106   Jan 30 19:46   reverse.c
```

- The “-l” is an option for the **ls** utility

Unix - Listing A File: cat

- We can use the **cat** utility to display the content of a text file.

```
cuse93:> cat reverse.c          ... display the contents of the "reverse.c"
/* reverse.c */
#include <stdio.h>

/* Function prototype */
void reverse (char* before, char* after);

/*****************************************/
int main()
{
   char str[100];  /* buffer to hold reversed string */
   reverse("cat",str);  /* reverse the string "cat" */
   printf ("reverse("cat") = %s\n", str);
                              :
                              :
```

Unix - Listing A File: more

- **cat** is good for listing small files, but doesn't pause between full screens of output.
- The **more** utility is better suited for larger files and contains advanced facilities such as the ability to scroll backward through a file.

```
cuse93:>  more reverse.c          ... display the contents of the "reverse.c"
/*  reverse.c  */
#include <stdio.h>

/*  Function prototype  */
void reverse (char* before, char* after);

/*****************************************/
int main()
                              :
                              :
```

Connect to Linux (from Windows)

- SSH (Secure Shell) can connect to remote machines
- Windows need third-party tools such as PuTTY
- For using GUI, need to start a X Server on Windows
  - VcXsrv, an open-source utility, can achieve

Connect to Linux (from Windows)

- For this course, we have 3 Linux servers:
  - linux03.se.cuhk.edu.hk
  - linux04.se.cuhk.edu.hk
  - linux05.se.cuhk.edu.hk
- To connect these servers outside CUHK, you need to connect to CUHK VPN

https://www.itsc.cuhk.edu.hk/all-it/wifi-and-network/cuhk-vpn/

Connect to Linux (from Windows)

- Start a local X Server using VcXsrv
- Run VcXsrc
  - run/start Xlaunch which will ask for some configurations
  - just take all the default options and click Next until you see the Finish button
  - click Finish and a small X icon will appear in the system tray
- Run putty.exe
  - Enable X11 forwarding
  - Host Name: e.g. linux03.se.cuhk.edu.hk