Name: Wan Muhammad Rayhan Arwindra

NPM: 1806241210

Reflection-Week 1

On the first week of our course, we were introduced to the definition of system programming, and also various other basic concepts that we need to understand how system programs work. We first learnt about the differences between application and systems programming, and also the difference between scripting and programming languages. Then, we learnt about the kernel, and its tasks within the OS. Finally we learnt about some commands that we can use within a UNIX system, and its directory hierarchy.

When learning systems programming, we first need to know what a system program is. Systems programming is the act of developing a software to serve the hardware and software of the machine. We do not interact with system programs, our computer does. This is different to application programs, which directly services the user. Systems programming requires us to understand more about the hardware of the system, as we are creating the program for them.

Speaking of programing, we also need to know about the difference between scripting and programming. Generally, the difference between scripting and programming is that scripts don't need to be compiled like programs. Also, scripting does not require a lot of object oriented paradigms, and usually don't have a user interface. Popular scripting languages nowadays for example are python, JavaScript, and PHP.

Next, we learnt about the kernel, which is the core program of an operating system. The kernel is located within the OS, and it's the first program to run when the system is activated. The kernel has several tasks, namely scheduling the queue of processes, managing memory, networking, file system, and handling interrupts and system calls. The kernel's services cannot be accessed by the user directly, but instead must be accessed by invoking a system call.

A computer system has two modes, which are user mode and kernel mode. In the user mode, the system can run user applications such as word processors. When the system encounters an interrupt or a system call, the system then switches from user mode to kernel mode. In kernel mode, the system runs core operating system components. The system starts of in kernel mode initially, after the OS fully loads, then it executes user applications in user mode. The mode bit of a system is set to 1 when it's in the user mode. When the system changes from user mode to kernel mode, the mode bit is changed to 0.

In UNIX, users can be assigned to a certain group, and each group and user has particular permissions or privileges. Every user and group has an identification, which is called UID or GID respectively. UNIX kernel also has a command called “man”. What man does is it displays to us the manual of a certain command, and also the sections of that command. The syntax of man is man [SECTION] [COMMAND_NAME]. The manual sections in order from 1 to 9 are: commands, library functions, system calls, special files, file formats, miscellaneous, system admin commands, and kernel routines. So for example of you wish to view the chmod command, you can type “man 1 chmod”.

UNIX has a hierarchy of its directories called the Filesystem Hierarchy Standard or FHS for short. The hierarchy forms a tree data structure, where all files and directories are placed under the root directory “/”. To change directories in UNIX, you can use the cd command. For example, “cd folder” would change directory to the “folder” directory. Files and directories have a relative and absolute path. An absolute path is the path to a certain file from the root directory, or “/”. On the other hand, a relative path starts from a directory under the root directory.