Name: Wan Muhammad Rayhan Arwindra

NPM: 1806241210

This week, we learnt about boot sequence, kernel compilation, and bootscript. A boot sequence is the sequential tasks done by a system to load the operating system, this usually happens when the machine is turned on. Kernel compilation is the act of creating our own custom kernel, after this process is done we can select this kernel to be run during the booting of the system. Finally, bootscripts are scripts which are run after the system successfully boots by the init program.

There are a couple of components handling the boot sequence, which are the BIOS, MBR, GRUB, Kernel, Init, and Runlevel. When the machine is switched on, the BIOS is loaded to perform a system checkup. Then, it loads the stage 1 of grub from the MBR, which is the first 512 bytes on the disk storage. Afterwards, it loads stage 1.5 and stage 2 of grub, which is the file system contained in the first 30kb of the disk, and the menu consisting of the list of kernels. After selecting the kernel and loading it, it then mounts the root directory of the file system and execute init. The init process then runs boot scripts.

There are three types of kernels, each having their own features, advantages and disadvantages. The monolithic kernel has almost all devices running in kernel mode, it's simple and quick but is instable. The microkernel has most services and drivers running in the user mode, it's more stable and secure but slower than the monolithic kernel. Finally, the hybrid kernel takes the best of both worlds. It transforms the monolithic kernel to run like a microkernel, meaning that it has both the performance of the monolithic kernel and the stability of the microkernel.

We can create our own kernel to run in our system by compiling a custom kernel. The first thing we need to do is install dependencies, we can do this by typing "sudo apt-get install kernel-package libncurses5-dev flex bison libssl-dev build-essential". Then, we need to get the latest linux package, which we can get from kambing.cs.ui.ac.id. We can then get the package by typing wget http://kambing.cs.ui.ac.id/linux/v4.x/linux-4.15.1.tar.xz, and unpacking it by typing tar xJvf linux-4.15.1.tar.xz. Then, inside the unpacked directory, we need to copy a working config file into a ".config" file. Then, we can load that config by using the menuconfig, and selecting load, and then selecting save. Then, we clean the packages by typing make-kpkg clean and compile it by typing make-kpkg –append-to-version=[VERSION_NAME] kernel_image kernel_headers. After this process is done (it took 90 minutes in my case), we need to install the linux-headers and image by the dpkg -i command. Then, we need to install the modules, by make modules_install. Then, we need to navigate to the /boot directory, and update the initramfs and grub. Finally, we can restart the system, and select our new kernel from the grub options.

A boot script or startup script can be made by using systemd and cron job. To make a startup script in systemd, we first need to navigate to the /lib/systemd/system directory. Then, we create a service in that directory, with filename extension of .service. We then start that service with the command systemctl start [SERVICE_NAME].service. We can track the service by typing systemctl status hello.service, and stop it with systemctl stop hello.service. To make a script in cron job, we first run the command crontab -e, and typing in the command @reboot /root/[SCRIPT_NAME] start. We can then restart the system, and the service would run.