CS5250 – Advanced Operating Systems
AY2020/2021 Semester 2

Assignment 1

Deadline: Monday, 8 Feb 2021 • 11.59pm

1. Objectives

1. Learn how to create and use a virtual machine in VirtualBox.
2. Know how to configure and build a new Linux kernel image.
3. Do some exercises in Intel assembly programming.

2. Rules

1. It is fine to ask for "reasonable" amount of help from others, but ensure that you do all the tasks and write the report on your own. The University's policy on plagiarism applies here and any breaches will be dealt with severely.
2. Do not put your assignment in the public domain, say GitHub. I had to deal with complaints of "stealing" of solutions in the past.
3. For this assignment, you are asked to finish the tasks and write a report (check assignment section for more details).
4. Generate your report as a pdf file, name it as "*Name (Student Number) Assignment 1.pdf*". *Due to past difficulties I had with uncooperative students, failure to follow this convention will result in marks deducted.* Upload your report in the **Luminus folder "Submissions for Assignment 1"** of CS5250**.**
5. **Your report should include your solutions to Part B.**
6. **The deadline of Assignment 1 is Monday, 8 Feb 2021.** *Late assignments lose 4 marks per day.*

3. Tasks (30 marks)

Part A: Linux Kernel Installation (20 marks)

1. Complete the followings:

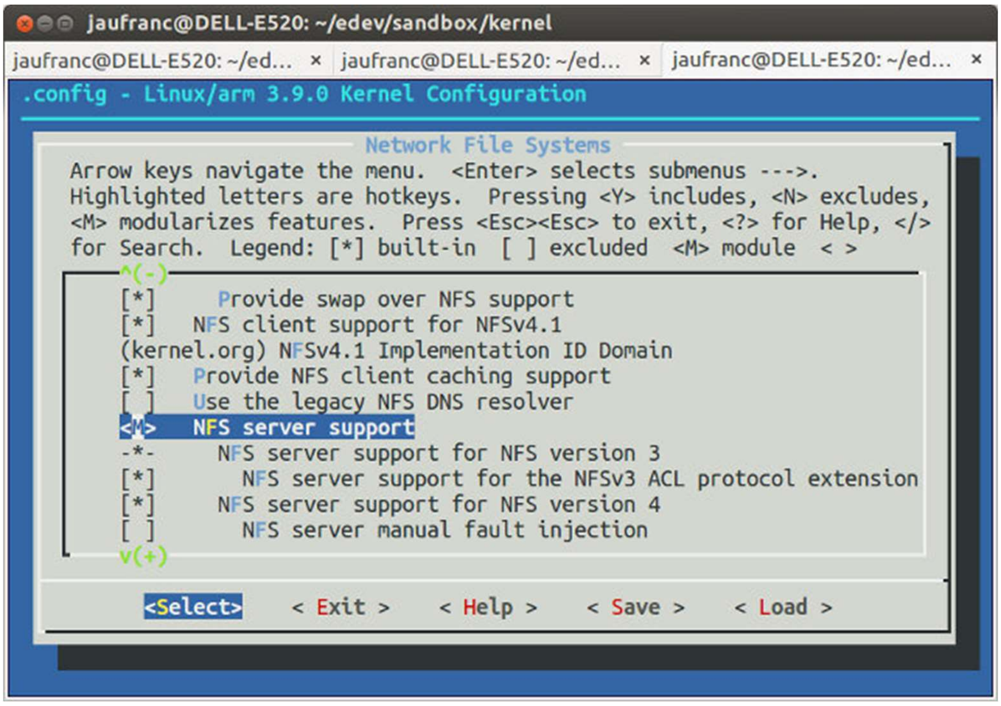
   a) Install the latest version of Virtualbox (https://www.virtualbox.org/wiki/Downloads)
   b) Install a Linux Ubuntu 20.10 guest machine (http://mirror.nus.edu.sg/ubuntu-ISO/20.10/)
   c) Compile and install a specific kernel in your guest machine

   You need to give the necessary screenshots (see the following for details) and explanations to show that you have gone through the entire workflow and understood what is happening.

You should also highlight any problems encountered and how you solved them. It is common to have problems during the setup stage. Try to figure it out yourself first. Always remember to save a snapshot before you try to take the next step so that you do not need to restart from the very beginning when your guest machine crashes. If something does go wrong, do not panic as you can always rollback and do it again. Remember to read the error messages carefully and search online for solutions. The abilities to search and debug is crucial in dealing with Linux.

However, if you cannot find solutions after you try all approaches you can think of, feel free to discuss it with your classmates or send an email to the lecturer.

2. During the “make menuconfig” stage, there are three different choices, built-in, excluded and module. (i) What do they mean? (ii) Which are the ones that will appear in the kernel image? (2 marks each for (i) and (ii)).


jaufranc@DELL-E520: ~/edev/sandbox/kernel
jaufranc@DELL-E520: ~/ed... ×
jaufranc@DELL-E520: ~/ed... ×
jaufranc@DELL-E520: ~/ed... ×

```
.config - Linux/arm 3.9.0 Kernel Configuration
                    Network File Systems
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
      ^(-)
      [*]     Provide swap over NFS support
      [*]   NFS client support for NFSv4.1
      (kernel.org) NFSv4.1 Implementation ID Domain
      [*]   Provide NFS client caching support
      [ ]   Use the legacy NFS DNS resolver
      <M>   NFS server support
      -*-     NFS server support for NFS version 3
      [*]       NFS server support for the NFSv3 ACL protocol extension
      [*]     NFS server support for NFS version 4
      [ ]       NFS server manual fault injection
      v(+)

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

(This screenshot is a sample from the web. It may not correspond exactly to what you will see.)

3. Try to build the smallest kernel image that will boot up successful on your virtual machine. By this I mean that it should boot to the same login screen as your original kernel and you should be able to log in. Give the commands you need, the screenshots of the size of the kernel image you build, giving this kernel a different name from that built-in Question 1. Remember to keep the original kernel in Question 1 for the next assignment. The one with the smallest kernel will win a small prize. List the options that you disabled. How small is your final kernel compared to the default one? (4 marks)

Detailed steps for Part A

Install Virtualbox on your machine

Virtualbox is an Open Source Software that can run on most common platforms. Download and install the latest version of the software on your PC/Mac/laptop if you do not have one.

https://www.virtualbox.org/

Install a Linux guest machine

1. Download an ISO file of Linux Ubuntu 20.10 from http://mirror.nus.edu.sg/ubuntu-ISO/20.10. You also can download the iso file from ubuntu official website. Make sure you download the *desktop* version and *amd64* for 64-bit machine. (“i386” refers to the 32-bit version.)
2. Set up a new virtual machine
    a. Start Virtual Box
    b. Click “New” button in the Oracle VM VirtualBox Manager. Use the setting listed below
        i. VM Name and OS Type
            1. Name: Input your student number
            2. OS Type:
                a. Operating System: Linux
                b. Version: Ubuntu
        ii. Memory
            1. Base memory Size: 1024MB (you can change this value later)
        iii. Virtual Hard Disk
            1. Create a virtual hard disk now. (Choose at least 40.00GB)
            2. Click Create.
3. Start your virtual machine by click “**Start**” on VirtualBox Manager.
    a. You should choose the ISO file you downloaded and then click continue.
    b. Install the Ubuntu as guided.
    c. Set your username as your full name and your password as you like.
    d. Open the terminal in your guest machine, output the OS, the kernel version and also your MAC address and **give a screenshot** including all the information.
4. If you broke something in the virtual machine that cannot be recovered. Just repeat the steps above to get a new one. However, the work done already will be lost.

Build a specific kernel

1. Download linux-5.10.6.tar.xz from (https://cdn.kernel.org/pub/linux/kernel/v5.x/). Unzip the file using the “tar” command, and enter the directory in terminal using “`cd`” command.
2. `make menuconfig`

    a. If you see some errors such as Linux Error: curses.h: No such file or directory; can not find flex, bison, etc, this means that there is no specific header in your machine and you need to figure out how to install that header. It will be easy to search this question online and get reasonable approaches.

3. `make`
4. `make modules_install`
5. `make install`
6. Set up bootloader. Now the kernel image is in the right place. However, the bootloader does not know it yet.
    a. The order of OS booting is in `/boot/grub/grub.cfg`. By editing this file, you can change the default OS and the order of different kernel versions.
    b. Try figuring out how to edit this file. **Give screenshots** of you changes.
        i. The original setting in the file and some Google search will be of great help.
        ii. Root privilege is needed as well.
    c. Creating backup or making snapshot before this step is recommended if this is the first time you do it.
7. Reboot the virtual machine and enjoy your new kernel.
    a. During reboot check whether boot option appears as you assume.
    b. If everything goes well, you should see the login prompt at this time.
    c. Login and type in the command `uname -a`, you will see the kernel version has changed to the new one. **Give a screenshot** in the report.
    d. You just finished configure, build and install the kernel. **Congratulations!**

8. Task 3 is essentially revisiting steps 2 and 3. Try to reconfigure the kernel such that it can still boot like the one in step 7 but is significantly smaller.

Part B: x86 Assembly Programming (10 marks)

There are online tools that can do some of the following. But avoid using them because you will not learn anything by using them. Assignments are meant to be part of the module's learning journey. Instant answers defeat this purpose.

1. What is the Intel64 (i.e., 64-bit processor) instruction corresponding to these bytes, showing each step of your decoding process? (4 marks)

   `4d 29 44 7a a7 (base 16)`

2. What is the 32-bit instruction encoding for the following instruction. You need to show each step of your workings. (2 marks)

   `addl 8(%esp), %ecx`

3. Disassemble the following IA32 assembly program by hand and recover the C program that performs the same task. (4 marks)

```
.section        .rodata
label_C0:
        .string "%d\n"
        .text
        .globl  unknown_func
unknown_func:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $40, %esp
        movl    $0, -12(%ebp)
        jmp     label_2
label_5:
        movl    8(%ebp), %eax
        movzbl  (%eax), %eax
        cmpb    $101, %al
        jne     label_3
        jmp     label_4
label_3:
        addl    $1, -12(%ebp)
        addl    $1, 8(%ebp)
label_2:
        movl    8(%ebp), %eax
        movzbl  (%eax), %eax
        testb   %al, %al
        jne     label_5
label_4:
        movl    -12(%ebp), %eax
        movl    %eax, 4(%esp)
        movl    $label_C0, (%esp)
        call    printf
        leave
        ret
```

Hope you can have fun and learn something from the assignment.