**Assignment 1: Virtual Machine and Linux**

CSE 330: Operating Systems - Fall 2023

Due by Friday, **September 8th 2023, 11:59 pm**

# Summary

The first assignment is a warm-up exercise that helps you become familiar with the virtual machine (VM) and Linux software that you will use extensively for your kernel development throughout this semester. In this exercise, you need to create a new VM using VirtualBox, install a Linux OS on the VM, and install the latest kernel in this Linux system.

# Description

Notes: The references included in the following description contain only high-level instructions for completing the tasks of this project; they are not exact step-by-step instructions. To get more help, attend the lectures, ask the instructor and TAs, and use your favorite search engine (there are lots of useful materials on the web!)

## Step 1: Create a new VM in VirtualBox.

We will use VMs extensively throughout the projects of this course. They make our lives much easier as kernel developers. We can conveniently test our new kernels without crashing the physical machines and take VM snapshots to save the progress of our work. In addition to completing this step as required in the project, I encourage you to play with your VM and get familiar with these useful features.

**Notes**:

* You can use either the VirtualBox in the instructional labs (*Hint: you can use the public computers on BYENG 2nd floor*) or install it on your own computer (it is free, supporting Windows, MacOS, and Linux). You can store your VM on a USB drive to make it portable.
* We will work on a 64-bit kernel in our projects, so make sure your VM is *64-bit* too.
* Kernel compilation is time-consuming, but it can be accelerated by using multiple CPUs to compile it in parallel, so give your VM as many cores as your physical machine has.
* Give your VM more than **50GB** of the virtual disk because building a new kernel requires a lot of space. But use dynamic allocation so that its actual storage usage grows as needed.
* **Download:** [**https://www.virtualbox.org/wiki/Downloads**](https://www.virtualbox.org/wiki/Downloads)
* **Reference**: “[VirtualBox End-user Documentation”](https://www.virtualbox.org/wiki/End-user_documentation)

## Step 2: Install Ubuntu 22.04.3 on your new VM.

Ubuntu is one of the major GNU/Linux distributions and it is quite user-friendly. In this step, you will install Ubuntu (version ***22.04.3 LTS***) on your new VMs which will be used for all the projects of this course. (LTS means that this version will be supported almost forever. 22.04.3 is the latest long-term support version of Ubuntu, released on **Aug 11, 2023**.)

**References**:

* “Ubuntu Desktop Guide”, URL: <https://help.ubuntu.com/lts/ubuntu-help/index.html>
* Ubuntu Desktop Download” URL: [Download Ubuntu Desktop | Download | Ubuntu](https://ubuntu.com/download/desktop)

## Step 3: Compile and install a new kernel on your new Ubuntu.

In this step, you will upgrade the kernel in your VM to the latest stable version which will be used as the basis for all your kernel development this semester. The below steps should be done inside the VM.

Basic instructions:

1. Download the long-term stable kernel (*6.4.11*) from:

<https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git/snapshot/linux-6.4.11.tar.gz>

1. Uncompress the file into a folder using tar -xvf linux-6.3.13.tar.gz .
2. Copy the kernel config file from your existing system to the kernel tree

The existing kernel config file can be found at /boot/config-$(uname -r) . Copy it in the folder you just uncompressed. Name this file as `**.config**`.

1. You may need to install the necessary dependencies such as***git, fakeroot, build-essential, ncurses-dev, xz-utils,******libssl-dev, bc, flex, libelf-dev, bison****.*

*Hint: You may need to figure out how to install these packages by apt or apt-get. You may also need to grant ‘sudo’ privileges to your current user.*

1. Bring the config file up to date: make oldconfig -j4  *(if your VM has 4 cores)* and press enter to accept all the default options.
2. If you need to make any kernel config changes, do the following and save your changes when prompted: make menuconfig .
3. For grading purposes, attach a unique local version to your kernel. For example, if your name is John Doe, your local version string should be “**CSE330Fall2023JohnDoe**”. It can be set using `***menuconfig`***, in “**General setup**” -> “**Local version**”
4. Check that the “Symmetric multi-processing support” option is enabled to allow your kernel to use the multiple cores that your VM has. You can understand the different options you can set in the kernel while browsing the menu for this option.
5. Disable the conflicting security certificates by executing the two commands below:

|  |
| --- |
| scripts/config --disable SYSTEM\_TRUSTED\_KEYS scripts/config --disable SYSTEM\_REVOCATION\_KEYS |

1. Build the kernel by executing the command make -j4 (this may take a long time when you do it for the first time) and press enter to accept all the default options.

*Hint: you can check your compilation by the command make after above make -j4. It should tell ‘Kernel: arch/x86/boot/bzImage is ready’.*

1. Build the modules by executing the command make modules -j4
2. Install the new kernel modules by executing the command make modules\_install -j4.

Hint: If permission is denied, you can add sudo to get permission.

1. Install the new kernel image by executing the command make install -j4.
2. Configure the GRUB boot-loader so it can load the new kernel image that you just built.

By running the update-grub command GRUB should find all the bootable kernels and add them to the GRUB menu.

Restart Linux and select the new kernel’s entry from the GRUB menu list.

By default the GRUB menu will not be shown; to see it, in /etc/default/grub change GRUB\_TIMEOUT=-1 (or long press shift when reboot) and run update-grub again; then when you reboot, you should see the menu.

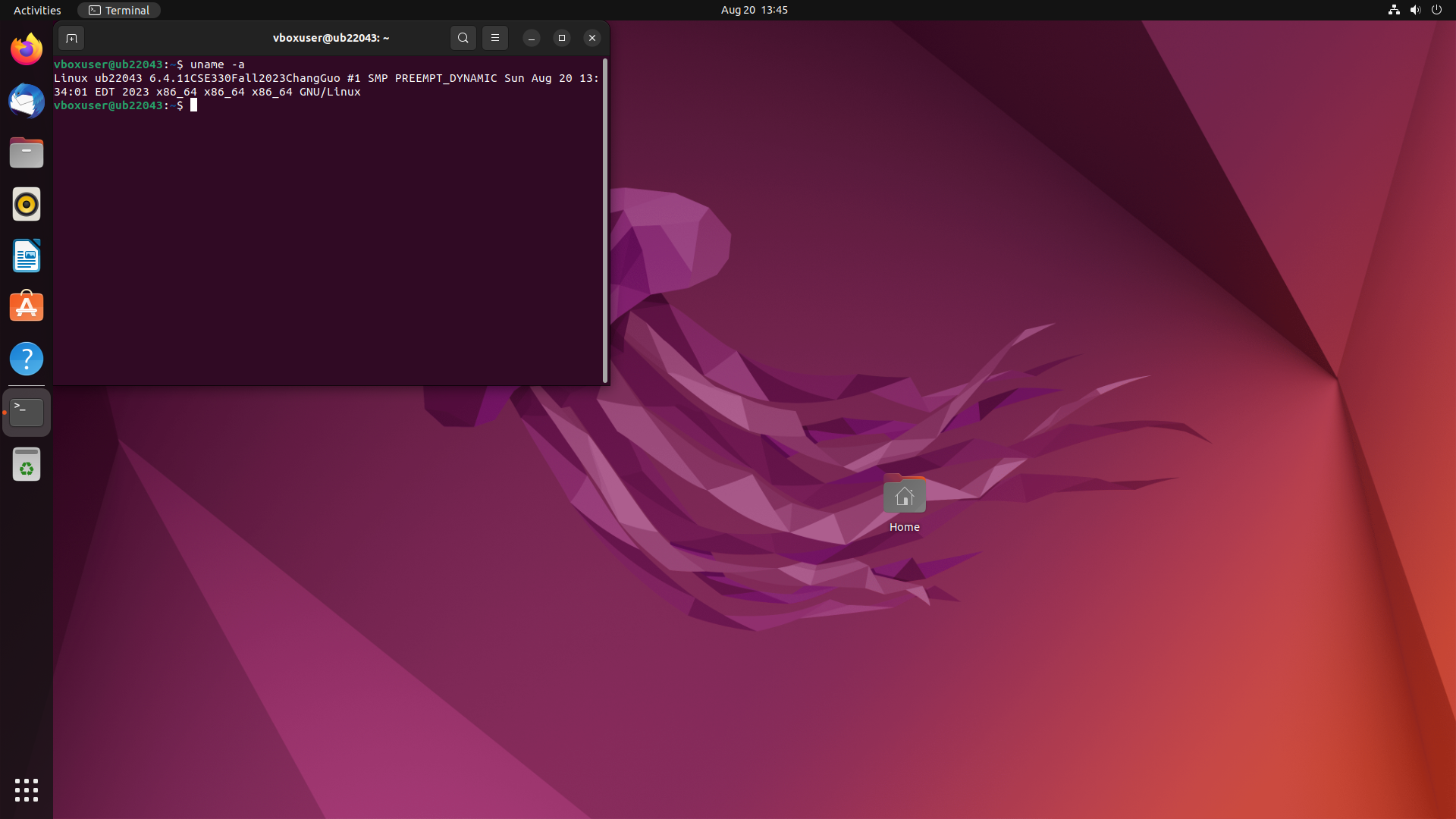
*Hint: grub requires you to run as root.*

1. Upon next login, check the current running kernel version using uname -a and verify that you are using your new kernel.

Sample Output:

**Choose ‘Advanced options for Ubuntu’**

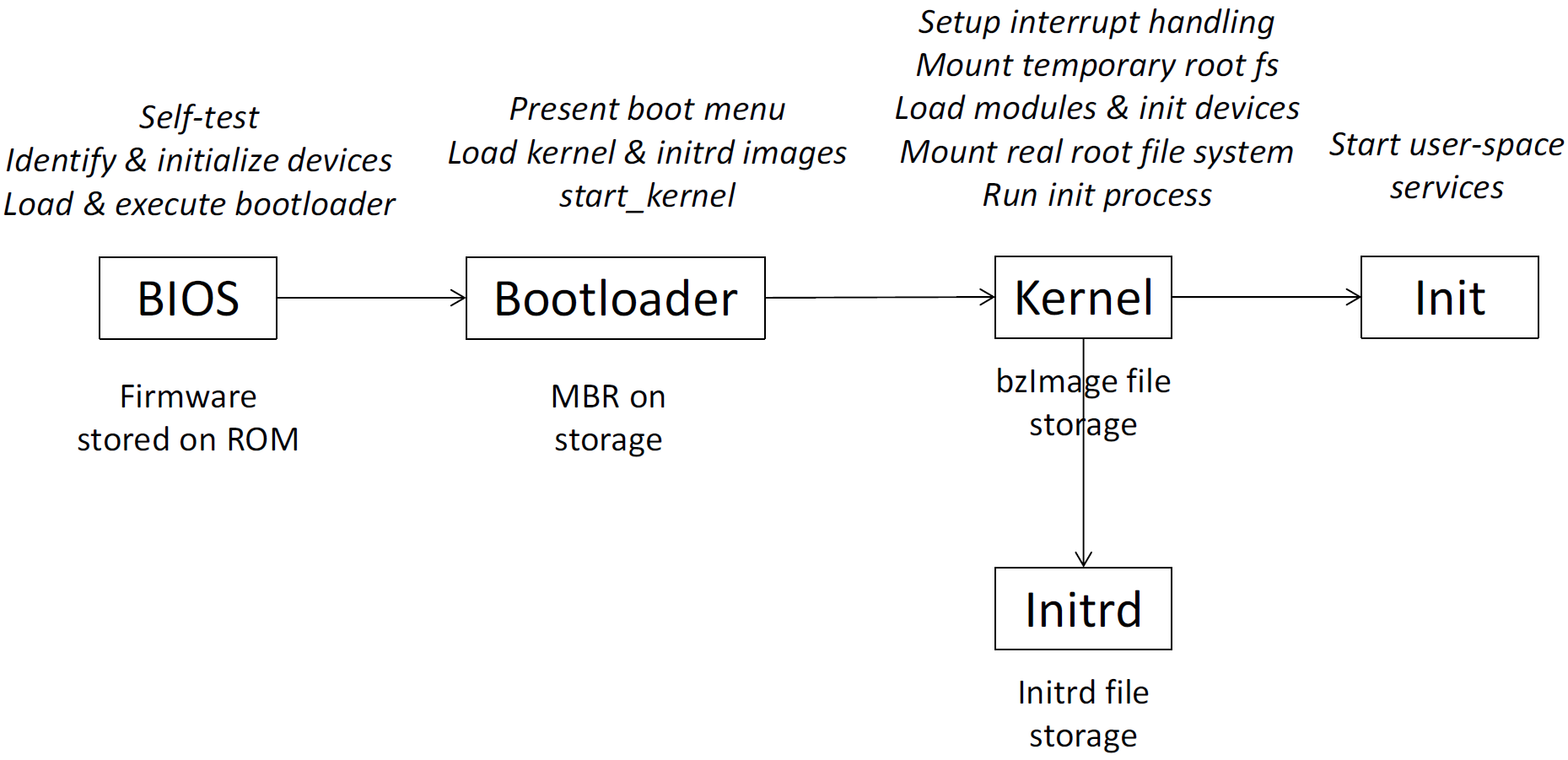
****

****

**Tips:**

* You should finish the above three steps in *three hours* or so. The following link would be helpful: <https://phoenixnap.com/kb/build-linux-kernel>
* Command-line tools are important for kernel development. For example, the above Step 3 requires the use of a variety of command-line tools. To help you become familiar with the command-line interface, try to do everything from a shell terminal, instead of using the GUI.
* If you don’t know what command to use, the quickest way to find is through a search engine. If you don’t understand the usage of a command, the easiest way is to run “*man*” followed by the command name.
* There are many Linux command-line references on the Web. Here is a good one: <http://wireless.ictp.it/school_2003/docs/linux/linux_guides/linux_command_reference.pdf>
* Some steps mentioned above require administrative privileges, e.g., using *apt-get* to install a missing package and using *make install* to install the new kernel image. When you run these commands as a regular user, you need to add “*sudo*” in front of the command, which gives you the necessary privileges to run the command (assuming you are a *sudo* user; check the man page of sudo for more information). You can also look at the Ubuntu documentation for sudo at: <https://help.ubuntu.com/community/RootSudo>
* Kernel compilation is quite time-consuming, so do it on a good computer (*Hint: you can use the public computers on BYENG 2nd floor*) and store your VM on a fast drive when you do it for the first time. You can also speed it up using parallel compilation using “-j”. Run “*man make*” and you can find the explanation of this option.

**Reference**: Linux boot process



**Figure 1. Linux boot process**

# Submission Requirements & Guidelines

Assignment 1 is due by **11:59pm – Friday, September 8th 2023**. Submit the project work following the below guidelines.

1. Submit the following (Maintain your code and file on the provided private GitHub repository in the [CSE330 Operating Systems - Fall 2023](https://github.com/CSE330-FALL-2023) Organization)
   1. **GitHub submission:** README file, listing the following:
2. Anything that you would like the TAs to consider for grading your submission, especially if you failed to work out the example screenshot.
3. For example, the command you used in each step, the problem you met, and your solution.
4. If your screenshot is wrong, TAs would grade based on your README file.
   1. **Canvas Submission:** Screenshots of the outputs.
5. The screenshots should include the outputs from Step **3**
6. You can use VirtualBox’s screenshot function to take a screenshot (to take a screenshot in VirtualBox, go to View-> Take Screenshot).
7. Create a .zip file with all your submission files. Name the zip file following the below-naming convention. **“Assignment-1-<Your Last Name>.zip”**
8. **Do not** submit any other source code
9. **Do not** submit any binary

# Policies

1. Late submissions will *absolutely not* be graded (unless you have verifiable proof of emergency). It is much better to submit partial work on time and get partial credit for your work than to submit late for no credit.
2. Every group needs to *work independently*on this exercise. We encourage high-level discussions among students to help each other understand the concepts and principles. However, a code-level discussion is prohibited and plagiarism will directly lead to failure of this course. We will use anti-plagiarism tools to detect violations of this policy.