EE 107S: Assignment 6

Due: Saturday, October 21, 2017 at 11:59 PM

Introduction

The purpose of this assignment is to familiarize yourself with building and installing a complicated project from source. The assignment itself may seem straightforward, but you'll find that you may run into inexplicable issues. This is common when working on any large open source project.

For this assignment you may work with up to one partner.

Assignment

Linux is a completely open-source kernel. Your job is to build the Linux kernel from source. First download and extract the Linux source code using the following commands:

Note: See the Using the LRC machines section **before** running the commands if you plan on using the LRC machines (recommended).

```
wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.13.7.tar.xz
tar xvf linux-4.13.7.tar.xz
```

From there it's up to you! Remember, a good place to start is always look for a file called `README`. The steps for building a project are generally configuring, building, and installing. For this project you will not install the new kernel.

Note on configuring the kernel

So far all of the projects we have built from source have had a `configure` script, but this will not be the case with every project. The Linux kernel uses a different method for configuring since it is massively customizeable. You will learn more about configuring the Linux kernel in the `README`. There are several methods of configuring it, but if you choose the correct one, it should be a matter of running a single command that takes a couple of seconds to run.

Using the LRC machines

While the Linux kernel is not as complex to build as some other open source projects, it still does take significant resources. The source and intermediate build files will add up to about 1.3 GB. Furthermore, the build may take significant time on less powerful CPUs. As such, I recommend using the LRC machines to build the kernel. They have 24 powerful cores and somewhere between 70 and 128 GB of RAM depending on the machine.

If you do choose to use the LRC machines, you should note that your home directory is limited to 1 GB of space. You should create a temporary directory in `/tmp/<username>/`, where you can

work on this assignment.

Please be considerate of others using the LRC machines! The machines are shared between all ECE students, and some research groups rely on the machines to run simulations and licensed tools. When you are performing the build, **do not use more than 8 threads**.

If you follow all these directions and everything goes smoothly, the build should take approximately 2 minutes on the LRC machines.

Bonus Assignment

Dolphin is a cross-platform, open source, GameCube and Wii emulator. Your job is to build and install Dolphin from source. First, download the Dolphin source code using the command:

```
git clone https://github.com/dolphin-emu/dolphin.git
```

Then follow the build instructions for a Linux Global Build that are reproduced (slightly modified) below.

```
mkdir build
cd build
cmake ..
make -j8              ; build with 8 threads
sudo make install
./dolphin-emu
```

Unlike the main assignment, I would recommend using your virtual machine or a local installation of Linux instead of the LRC machines. Dolphin requires more resources and will take longer to compile than the kernel, but it also has a significant number of dependencies. Since you do not have root access to the LRC machines, you will be unable to easily install any of those dependencies.

The most immediate dependency is a program called CMake. CMake is a build system that generates a `Makefile`. One of the steps CMake will perform is to make sure all dependencies necessary are installed. You will spend most of your time making it through the `cmake ..` command. Once you have built and installed the project, you may use the command `./dolphin-emu` to launch Dolphin.

And...that's it!

Useful notes

- Since Dolphin is a large project, it will take some time to build (for reference, it took me 10 minutes with an i5-6600K @ 4.6 GHz with 4 physical cores dedicated to the virtual machine). Luckily, CMake generates a Makefile that shows the percentage complete after each individual file is compiled. I would recommend working on the more powerful of the two machines between partners.

- It is often difficult to understand the errors that CMake or `make` will generate. It may be helpful to search some of the words in the error message and then tack on "dev tools ubuntu". For example, you may search "x11 dev tools ubuntu" during this assignment.
- There are lots of additional features you can build into Dolphin if you want (you may notice some dependencies are skipped along the way and features are disabled instead of causing an error). However, for this assignment you just need to be able to get through the basic build completely.

Submission

You will submit a report describing the steps necessary to build the Linux kernel (and Dolphin if you choose). The report can simply be a list of packages you had to install along the way and the commands used to build the project. If you've forgotten the commands you ran, you can type in the command `history`. In the report you may also describe packages you had particular difficulty installing or anything else you learned along the way. Submit one report per group to Canvas and include both partners' names on it.