L41: Lab Setup

Lent Term 2015

L41 is taught through a blend of lectures and laboratory experiments. The purpose of the labs is threefold: to teach you about real-world operating systems, to teach you experimental methodology and practical skills, and to provide fodder for assessment. You will use tools such as DTrace to explore the behaviour of the system through 'potted' example programs that will trigger OS behaviours for you to investigate.

Each lab is structured in two parts: a set of optional exploration questions that will help you learn about OS performance and performance analysis, and a set of mandatory experimental questions that you will address in your lab report. The former are one possible way to approach the problem; you should feel free to use other techniques or approaches. Do ensure, however, that your lab report does address the assigned experimental questions.

Experimental platform

Our experimental platform is the open-source FreeBSD operating system running on the BeagleBone Black (BBB) board, described in the remainder of this handout.

The operating system: FreeBSD

We will be using the open-source FreeBSD operating system's ARMv7 port on the BeagleBone Black. FreeBSD is of particular interest due to its tight integration of a number of tracing and measurement tools (e.g., DTrace) and that it is built by default with the Clang/LLVM compiler suite, which make it easier to insert additional instrumentation. You can learn more about FreeBSD by visiting the FreeBSD Project's website:

`http://www.FreeBSD.org/`

The course text, *The Design and Implementation of the FreeBSD Operating System, Second Edition* will be a useful reference, covering concepts such as the process model, inter-process communications, and filesystems. There is also a section on the implementation of DTrace, which may be useful background material for the labs.

The board: BeagleBone Black

The BeagleBone Black board is based on a Texas Instruments System on Chip that employs a single-core ARM Cortex-A8 processor; it includes on-board flash, an Ethernet MAC, USB support, and an SD Card slot. This 32-bit processor is not the zippiest in the world – but it is widely deployed and fully functional with respect to many of the behaviours of interest for this course, including processor performance counters. In our configuration, FreeBSD boots off of the on-board SD Card.

We will access the board as a USB device from lab workstations or personal notebook computers. "USB target mode" allows the BBB to appear to be a set of USB devices to the attached workstation: an Ethernet device that can be used to SSH into the board, and a USB device that accesses a serial console. Power is also provided using the USB cable, although we also have external power adapters if you would prefer (i.e., if you want to leave the board running an experiment while you unplug your notebook from it). These instructions describe how to access the board from a Linux workstation in the lab. Other configurations, such as Mac or Windows notebooks, or FreeBSD workstations, may also work with suitable adjustment to client-side tools.

While the BBB is not enormously expensive, they are often back ordered, so acquiring replacements can be inconvenient. If you lose or damage your board (e.g., by dropping it), then we may ask you to pay to replace the board. It is fairly straight forward to "brick" the board by overwriting the contents of its on-board flash, the OS image on the SD Card, etc; this is discouraged. If this appears to have occured, please contact the module instructor for assistance. A small number of replacement boards are available if required.

It is easy to replace the SD Card in the board – should it appear to have become corrupted or entered an unusable state (they appear to be sensitive to power being removed from the device while an operation is in progress), let us know and we can provide a replacement. Note that any customisation to your board (e.g., SSH keys, `/data` partition) will not be present on the replacement card – do ensure that you keep the primary copy of any important data on your workstation.

Getting started

The BeagleBone Black will arrive in a compact plastic case, and with a pre-initialised SD Card holding the OS image. For the purposes of this course, you should not need to disassemble the case or eject the card. We will power and communicate with the board via a single USB cable. These directions assume that you will be working on an ACS Workstation running Ubuntu Linux. However, our teaching setup has also been used with varying degrees of success with other systems, including FreeBSD and Mac OS X[1] The BBB appears to the workstation operating system as two USB devices: a USB serial port, which reaches a console, and a USB Ethernet device, which can be used to SSH into the OS instance on the board.

When you plug in the BeagleBone Black, at least one blue LED should light up on the board. It can take a minute or so for the OS image to boot; once it does, the two USB devices will become visible on the workstation. You can connect to the console using the following command:

```
minicom -o -D /dev/ttyACM0
```

You should be able to log in as `guest` without a password. However, to log in via SSH, which will be much more convenient for streaming data or transferring files, you will need to set up an `authorized_keys` file in `guest`'s `~/.ssh` directory, (`/data`). If you already have a `~/.ssh/id\_rsa.pub` (or similar) file in your CL home directory, you can place that in the `authorized_keys` file. Otherwise, you can use `ssh-keygen` to generate a keypair first. Once the key is installed, you should be able to SSH to the BBB:

```
ssh guest@192.168.141.100
```

To run `dtrace`, you will need to switch to the `root` user's account using the `su` command, after which you should have a root shell prompt. When running as `root`, you should exercise caution: accidentally deleting system files may render your board unbootable or cause loss of your scripts or data. Now, run your first DTrace script:

```
dtrace -qn 'BEGIN { printf("Hello world"); exit(0); }'
```

To write the output of a script to a file, you can redirect standard output to a file:

```
cd /data
dtrace -qn 'BEGIN { printf("Hello world"); exit(0); }' > data.out
```

We have configured the root filesystem to be read-only by default, in order to reduce the chance of accidents, but also to reduce wear on the SD Card. The `/data` filesystem is writable, however, and is a good place to stick your scripts, data, compiler output, etc, so that it will persist across reboots. You will likely wish to copy files from the BeagleBone Black to your workstation for analysis – e.g., to load it into a spreadsheet. You may also want to edit scripts and source files on the workstation. You can copy files back and forth using `scp`; this command can be run on the Linux workstation to copy `data.out` to your current working directory:

```
scp guest@192.168.141.100:/data/data.out .
```

[1] We have had mixed experience with Mac OS X and Windows, as the OS image on the BBB implements 'USB target mode' – i.e., appears to be a set of USB devices – and device drivers on operating systems vary quite a bit. If you appear to have reliability problems with the serial console or Ethernet parts, it may well be a device-driver bug. You are welcome to help us debug the problem, but it may be simpler to work with the systems that we have tested with.