CSE 4/521 – Introduction to Operating Systems
Pintos Overview
University at Buffalo, Summer 2018

Objective:

- To describe the fundamentals of Pintos Operating System.
- To establish the grounds of what needs to be done for programming assignments on Pintos.

Fundamental Steps:

1. Download the .ova file available at : https://jerryajay.com/cse-4-521/ (by clicking the Pintos VM Download button).
2. In VirtualBox software, from the File menu, choose "Import Appliance" and point to the downloaded "UB-pintos.ova". You do not need to change any other settings for this procedure. This will import the image and create a Virtual Machine on your computer. For this operation you need about 9GB free space.
3. Now choose UB-Pintos from the left pane, and click on "Start", your virtual machine should now boot and get you to a login page:

```
Username: os-class
Password: cse421521
```

4. To test if everything works, open a terminal (Ctrl+Alt+T), and enter:

```
cd $PINTOSDIR/src/threads && make
cd build && pintos run alarm-multiple
```

If you see the 'bochs' window pop up and some output related to threads, you are good to go!

Overview of Expectations for Programming assignments:

There is much to learn about navigating through Pintos code. First programming assignment starts shortly, and is based on threads. The programming assignments in Pintos OS would not require you to understand every line. The intuition of which section is important (and which is not) comes through experience. So, make sure to utilize your time to gather this experience, before you can think of how to improve implementation.

Refer Pintos manual https://web.stanford.edu/class/cs140/projects/pintos/pintos.html (skipping parts that are specific to Stanford CS140) regularly. Some concerns you might be having might have already been answered there (since they are the creators of this OS).

You will read, modify, and write a lot of Pintos source code. You'll have to compile the code, and run it on the emulator (bochs) to see the output. The code actually runs as if it is on a x86 hardware.

Credits:

Prof. Tevfik Kosar
Mr. Farshad Ghanei