0. Introduction

Welcome to Pintos. Pintos is a simple operating system framework for the 80x86 architecture. It supports kernel threads, loading and running user programs, and a file system, but it implements all of these in a very simple way. In the Pintos projects, you will strengthen its support by yourself.

Pintos could, theoretically, run on a regular IBM-compatible PC. Unfortunately, it is impractical to supply every SE 223 student a dedicated PC for use with Pintos. Therefore, we will run Pintos projects in a system simulator, that is, a program that simulates an 80x86 CPU and its peripheral devices accurately enough that unmodified operating systems and software can run under it. In class we will use the Bochs and QEMU simulators. Pintos has also been tested with VMware Player.

This project is hard. SE 223 has a reputation of taking a lot of time, and deservedly so. We will do what we can to reduce the workload, such as providing a lot of support material, but there is plenty of hard work that needs to be done. We welcome your feedback. If you have suggestions on how we can reduce the unnecessary overhead of assignments, cutting them down to the important underlying issues, please let us know.

This chapter explains how to get started working with Pintos. You should read the entire chapter before you start work on any of the projects.

0.1 Getting Started

To get started, you'll have a machine that Pintos can be built on. You may use the Solaris or Linux machines. We will test your code on these machines, and the instructions given here assume this environment. We cannot provide support for installing and working on Pintos on your own machine, but we provide instructions for doing so nonetheless.

0.1.1 Install Pintos with a Virtual Machine

The best way to install Pintos on your machine is to use a pre-installed copy on a Linux operating system via VirtualBox VM (actually, we have export the virtual disk as a format that VMWare Player can use to setup a virtual machine, but we **do not** respect you to do so). If you have not used a VM before, the idea is that you will be running an entire operating system on inside a program (VirtualBox) that looks to the virtual guest operating system like real hardware. We provide downloadable image of the guest OS with everything installed, so that you only need to do minimal setup to get started.

1. First, download Oracle's VirtualBox, either from the VirtualBox website at https://www.virtualbox.org/wiki/Downloads, or from Oracle's download page at http://www.oracle.com/technetwork/server-storage/virtualbox/downloads/index.html. Choose the version for the operating system you are running on your computer.

2. Once the download completes, run the installer to install VirtualBox. You will need administrator access on your computer to do so.

3. You may also find it necessary to download and install the VirtualBox Extension Pack to paly around with the VirtualBox just like a real machine. With this Extension Pack, your VirtualBox will support for USB 2.0 devices, VirtualBox RDP and PXE boot for Intel cards.

4. And then, download our linux virtual machine image from ftp://my.ss.sysu.edu.cn/~wh/course_down/, Note that this download is approximately 1.6GB.

5. Unzip the file into a convenient directory. The unzipped files are about 1.7GB, so make sure you have enough disk space available. If you are a Windows XP user and have trouble opening the zip file, try WinZip (shareware) or 7-Zip (free, open source) instead of the built-in zip support in Windows.

6. Once you have unzipped the VM, double-click on the file "Pintos.ovf" This should open the VM in VirtualBox and you can forget the rest of this chapter and start to writing your own Pintos Project!

Note: Our VM's provided username is student and password is 123456. And the Pintos's source folder is in ~/SE223/.

However, we recommend you spend some time to have a look at what we have installed for you: * Pintos Source Code. This include the basic code of Pintos.

- GCC. Version 4.0 or later is preferred. Version 3.3 or later should work. If the host machine has an 80x86 processor, then GCC should be available as gcc; otherwise, an 80x86 cross-compiler should be available as i386- elf-gcc. A sample set of commands for installing GCC 3.3.6 as a cross-compiler are included in ‘src/misc/gcc-3.3.6-cross-howto’.

- GNU binutils. Pintos uses addr2line, ar, ld, objcopy, and ranlib. If the host machine is not an 80x86, versions targeting 80x86 should be available with an ‘i386-elf-’ prefix.

- Perl. Version 5.8.0 or later is preferred. Version 5.6.1 or later should work.

- Autoconf. Autoconf is an extensible package of M4 macros that produce shell scripts to automatically configure software source code packages.

- GNU make, version 3.80 or later.

- GDB. GDB is helpful in de- bugging. If the host machine is not an 80x86, a version of GDB targeting 80x86 should be available as ‘i386-elf-gdb’.

- Bochs, version 2.2.6.

- Git. Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

- GNU Emacs. GNU Emacs is an extensible, customizable text editor—and more. "Emacs is a refugee

from the long dead culture of LispMachines. It's an asylum seeker in the UnixCulture."

- Vim. Vim is an advanced text editor that seeks to provide the power of the de-facto Unix editor 'Vi', with a more complete feature set.

0.1.2 Install Pintos on your own machine

If you decide to install Pintos on your own machine so that you can have a better performance, first you should have a working Unix-like systems(but not Mac OS). It has been most extensively tested on GNU/Linux, in particular the Debian and Ubuntu distributions, and Solaris. It is not designed to install under any form of Windows.

Then you need to install a Pintos development environment including the list as described before except the GUN Emacs and Vim(however, we recommend you to use one of them). Once these prerequisites are available, follow these instructions to install Pintos:

1. Get a copy of the archive of Pintos Source Code. you can extract the source for Pintos into a directory named **'pintos/src'**, by executing `zcat /usr/class/cs140/pintos/pintos.tar.gz | tar x`
2. Configure Bochs by passing `--enable-gdb-stub --with-nogui` to the ./configure shell script.
3. Go to the folder **'pintos/src/utils'**, edit the Makefile to replace `LDFLAGS` with `LDLIBS` and make.
4. Copy **'backtrace'**, **'pintos'**, **'pintos-gdb'**, **'pintos-mkdisk'**, **'pintos-set-cmdline'**, and **'Pintos.pm'**, **'squish-pty'** into the default **PATH**.
5. Install **'src/misc/gdb-macros'** in a public location. Then use a text editor to edit the installed copy of **'pintos-gdb'**, changing the definition of `GDBMACROS` to point to where you installed **'gdb-macros'**. Test the installation by running `pintos-gdb` without any arguments. If it does not complain about missing **'gdb-macros'**, it is installed correctly.
6. Pintos should now be ready for use. If you have the Pintos reference solutions, which are provided only to faculty and their teaching assistants, then you may test your installation by running `make check` in the top-level **'tests'** directory. The tests take between 20 minutes and 1 hour to run, depending on the speed of your hardware.

0.2 Assignment: Hello, Pintos

0.2.1 Description

Once your Pintos have been ready for use, you can have a look inside **'src/threads/init.c'**, we have provided an empty function named **void hello_pintos(void)**. After Pintos boot completes, our code will create a new thread that runs this function if you pass the argument `run hello-pintos` to our Pintos. For more specifically, you can folow the steps to run this: 1. cd to **'src/threads/'** and run `make`. 2. run `pintos -q run hello-pintos` in this folder. 3. you will find Pintos should star up and run function

"void hello_pintos(void)".

0.2.2 Requirement

This requirement is quite simple. You just need to modify the function "void hello_pinots(void)" to print a message like "Hello, your name !", e.g. "Hello, jeason!".

When you have finished this, please download the DESIGNDOC and archive your code and DESIGNDOC into a **zip** file and upload to the ftp://my.ss.sysu.edu.cn/~wh/homework*upload/13%BC%B6%20%B2%D9%D7%F7%CF%B5%CD%B3/assignment*1/