Introduction to Pintos

Jin-Soo Kim (jinsookim@skku.edu)  
Computer Systems Laboratory  
Sungkyunkwan University  
http://csl.skku.edu
Welcome to Pintos!

What is Pintos?

- An instructional operating system
- Developed by Ben Pfaff @ Stanford U.
- A real, bootable OS for 80x86 architecture
  - Run on a regular IBM-compatible PC or an x86 simulator
- The original structure and form was inspired by the Nachos instructional OS from UC Berkeley (Java-based)
- A few of the sources files are derived from code used in the MIT’s advanced operating systems course
- Written in C language (with minimal assembly code)
What is Bochs?

- Open-source IA-32 emulator
- Simulates a complete Intel x86 computer in software
  - Interprets every instruction from power-up to reboot
  - Has device models for all of the standard PC peripherals: keyboard, mouse, VGA card/monitor, disks, timer, network, ...
  - Supports many different host platforms: x86, PowerPC, Alpha, Sun, and MIPS
- Runs most popular x86 OSes:
  - Windows 95/98/NT/2000/XP/Vista, Linux, BSDs, ...
- Written in C++
- Emulation, not virtualization
Bochs (2)

- Linux + Bochs
  - We will run Pintos using Bochs on Linux
  - Bochs makes it easy to develop and debug Pintos projects
Setting Up (1)

- **Install Linux distribution on your machine**
  - Debian, Fedora, Ubuntu, or whatever you like

- **Install development tools**
  - Including gcc, make, perl, gdb, and so on
  - GCC >= 4.0, binutils >= 2.13

- **Install development libraries, (for Bochs)**
  - Install X windows development libraries, if needed
    - For Debian, install xorg-dev package
  - Install curses development libraries, if needed
    - For Debian, install libncurses5-dev package
  - There could be additional libraries to install
Setting Up (2)

- **Install Pintos**
  - Download the Pintos package (pintos.tar.gz)
    - Available from [http://csl.skku.edu/SSE3044F11/Resources](http://csl.skku.edu/SSE3044F11/Resources)
    - Use this version only
  - Untar Pintos
    
    ```
    $ tar xvzf pintos.tar.gz
    ```
  - Build Pintos
    ```
    $ cd pintos/src/threads
    $ make
    ```
    - This will create the kernel image (kernel.bin) and the final OS disk image (os.dsk = loader.bin + kernel.bin) in ./build
Setting Up (3)

- **Install Bochs**
  - You need Bochs to run Pintos
  - Get the source code from [http://bochs.sourceforge.net](http://bochs.sourceforge.net)
    - Make sure you are downloading v2.2.6 (bochs-2.2.6.tar.gz)
    - You don’t have to untar the source code
  - Install Bochs
    - Must patch the Bochs source code for Pintos (Patches are available in pintos/src/misc)
    - Use the installation script provided by Pintos (pintos/src/misc/bochs-2.2.6-build.sh)
    - The script will untar, patch, configure, compile, and install Bochs
    - You need to be a superuser (root) to install Bochs in the system directory (e.g., /usr/local)
Setting Up (4)

- Install Bochs (cont’d)
  - Running the script:

```
行(0) 域(0) 觀(0) 司(0) 道(0) 老(0) (H)

gkm2164@ubuntu:~$ cd OSProject/
gkm2164@ubuntu:~/OSProject$ cd pintos/src/misc
gkm2164@ubuntu:~/OSProject/pintos/src/misc$ ls
bochs-2.2.6-big-endian.patch  bochs-2.2.6-paranoia.patch
bochs-2.2.6-build.sh          bochs-2.2.6-solaris-link.patch
bochs-2.2.6-gdbstub-ENN.patch bochs-2.2.6-solaris-tty.patch
bochs-2.2.6-jitter.patch      bochs-2.2.6-triple-fault.patch
bochs-2.2.6-ms-extensions.patch gcc-3.3.6-cross-howto
bochs-2.2.6-page-fault-segv.patch gdb-macros
gkm2164@ubuntu:~/OSProject/pintos/src/misc$ ./bochs-2.2.6-build.sh
usage: env SRC_DIR=<srcdir> PINTOS_DIR=<srcdir> DST_DIR=<dstdir> sh ./bochs-2.2.6-build.sh
where <srcdir> contains bochs-2.2.6.tar.gz
and <pintosdir> is the root of the pintos source tree
and <dstdir> is the installation prefix (e.g. /usr/local)
```
Setting Up (5)

- **Install Bochs at Ubuntu 10.04**
  - `sudo apt-get install`
    - `patch`
    - `diff`
    - `g++`
    - `xorg-dev`
    - `ncurses-dev`
Setting Up (6)

- Test Bochs

```bash
$ bochs ; Put $DSTDIR/bin into your PATH
```
Setting Up (7)

- Setting pintos-gdb

```
$ vim pintos/src/utils/pintos-gdb
```

```bash
#!/bin/sh

# Path to GDB macros file. Customize for your site.
GDBMACROS=/home/gkm2164/OSProject/pintos/src/misc/gdb-macros

# Choose correct GDB.
if command -v i386-elf-gdb >/dev/null 2>&1; then
    GDB=i386-elf-gdb
else
    GDB=gdb
fi

# Run GDB.
if test -f "$GDBMACROS"; then
    exec $GDB -x "$GDBMACROS" "$@
else
    echo "*** $GDBMACROS does not exist ***"
    echo "*** Pintos GDB macros will not be available ***"
    exec $GDB "$@
fi
```
Setting Up (8)

- Run Pintos

```bash
$ cd pintos/src/threads
$ ../utils/pintos run alarm-multiple
```

```
(alarm-multiple) thread 0: duration=10, iteration=7, product=70
(alarm-multiple) thread 1: duration=20, iteration=4, product=80
(alarm-multiple) thread 3: duration=40, iteration=2, product=80
(alarm-multiple) thread 2: duration=30, iteration=3, product=90
(alarm-multiple) thread 4: duration=50, iteration=2, product=100
(alarm-multiple) thread 1: duration=20, iteration=5, product=100
(alarm-multiple) thread 2: duration=30, iteration=4, product=120
(alarm-multiple) thread 3: duration=40, iteration=3, product=120
(alarm-multiple) thread 1: duration=20, iteration=6, product=120
(alarm-multiple) thread 4: duration=50, iteration=7, product=140
(alarm-multiple) thread 2: duration=30, iteration=5, product=150
(alarm-multiple) thread 3: duration=40, iteration=4, product=160
(alarm-multiple) thread 2: duration=30, iteration=4, product=160
(alarm-multiple) thread 3: duration=40, iteration=3, product=150
(alarm-multiple) thread 4: duration=50, iteration=4, product=200
(alarm-multiple) thread 1: duration=20, iteration=5, product=200
(alarm-multiple) thread 2: duration=30, iteration=7, product=210
(alarm-multiple) thread 3: duration=40, iteration=6, product=240
(alarm-multiple) thread 4: duration=50, iteration=5, product=250
(alarm-multiple) thread 3: duration=40, iteration=7, product=280
(alarm-multiple) thread 4: duration=50, iteration=6, product=300
(alarm-multiple) thread 4: duration=50, iteration=7, product=350
(alarm-multiple) end
Execution of 'alarm-multiple' complete.
```
Project -1: Warming Up
Project -1 (1)

- Set up your own project environment
  - Install Linux
  - Install all the required tools
  - Install Pintos
  - Capture the screen shot of working Pintos
    
    $ pintos run alarm-multiple
Project -1 (2)

- Documentation
  - Specification of your environment
    - Linux distributions, versions of gcc, etc.
  - A screen shot of “alarm-multiple”

- Due:
  - Sep. 2, 11:59PM (NO slip day)
  - Submit via e-mail to sse3044@csl.skku.edu
  - Note: This is an individual project