OS LAB0: THE SETUP

Miro Mannino

01 August 2022

CLASS LOGISTICS

LINKS

The lab assignments, source code, and virtual machine images can be found in `azzadev.github.io/osbook`. The passcode is 'os2022'.

Getting help:

- Email to Azza (azza@nyu.edu) or Miro (miro.mannino@nyu.edu)
- Slack channel: join here.
- Book appointments:
  - Azza: `https://calendar.app.google/Zud7cR3xKKbdnsJ28`; `https://calendly.com/prof-azza`.
  - Miro: `https://calendar.app.google/unzLQ7BZmBijF7b47a`

LABS INTRODUCTION

LABS OVERVIEW

In these labs you will be developing components of an Operating System

- Boot-loader
- Preemptive and non-preemptive kernel module
- IPC infrastructure
- Virtual memory
- File system

You will be using bochs emulator to test your OS Implementation of the OS will be done using C and assembly

LAB0

LAB0 OVERVIEW

Setting up the development environment

- Setting up the VM
- Trying out bochs simulator

SETTING UP THE VM

- Download VirtualBox
- Download the provided VM image labpc-virtual.zip
- Add the VM image to VirtualBox using labpc-virtual.vbox
- Configure the VM to have good performances:
  - Give enough base memory if you can
  - Give more CPUs if you can and Enable PEA/NX
  - Give more video memory if you can
  - Enable 3D Acceleration if you can
  - In Storage options add as optical drive the VBoxGuestAdditions.iso
- Start the virtual machine

START THE VM

- Login using username ”student” and password ”cos318”
- Install the VirtualBox Guest Additions opening the optical drive
- In the optical drive click Open Autorun Prompt
- Install using ”root” as root password
- Restart
- Configure a shared folder if you want to edit files in your host and use the VM only for compiling and running
- If you have issues accessing the shared add the vboxsf group with sudo usermod –aG vboxsf student with the root account (you can do that in student’s terminal with the command su)

COMPILE THE CODE

- Open the terminal
- Go to folder ~/318/codes/project1
- Unpack the start-code: `tar -xvzf start_code_1.tar.gz`
- Go to the extracted folder: `cd start_code_1`
- Make it with `make`

RUN THE CODE

- Run the simulator using bochs in the terminal
- Begin the simulation
- The simulation as it is does nothing since it’s just going in a loop, notice how the VM might look a bit unresponsive
- Play with some of the code in the bootblock_example.s
    - Edit bootblock.s
    - Make
    - Run the simulation again

BOCHS DEBUGGING

Running bochs in debug mode (e.g. bochsdbg) you have access to few debugging commands:

- r to show the registers
- sreg to show the segment registers
- b set a breakpoint
- s step
- n next
- c continue
- d delete a breakpoint
- xp example memory at physical address
- u disassemble

More info: `https://bochs.sourceforge.io/doc/docbook/user/internal-debugger.html`

Questions?