Project 1 Yi-Jung Chiang

CS 544

Spring 2017

Abstract

In this project I compiled version 3.14.26 of the Linux kernel and ran it in a virtual machine using qemu.Moreover, I completed a project implementing producer-consumer problem.

I. Log of Commands

This is a selection of the commands I ran to clone and build the Linux kernel on the os-class server. Not all of them were run in the same shell. Note that I use several aliases for git, vim, and other common commands.

```
mkdir /scratch/spring2017/10-01/
```

I ran this to create a directory at /scratch/spring2017/10-01
Next I cloned the linux project into this newly created directory with:

```
$ git clone git://git.yoctoproject.org/linux-yocto-3.14
```

Then, I checked out the correct branch with:

```
$ git checkout v3.14.26
```

Next I configured environment with the command:

```
$ source /scratch/opt/environment-setup-i586-poky-linux
```

In the respective directory, I ran the command into debug mode:

```
$ qemu-system-i386 -gdb tcp::5078 -S -nographic -kernel bzImage-qemux86.bin -drive file=
```

And now with qemu running in debug mode, I start up another terminal (the VM)and ran the commands to source and target the VM:

```
$ source/scratch/opt/environment-setup-i586-poky-linux
$ $GDB
$ target remote :5078
$ continue
```

Go back to the initial window, QEMU/YOCTO should be building. But before we can use Yocto, we need to build it first. To do so I ran the following commands to copy the correct .config file and build it with 4 threads.

```
$ cp /scratch/spring2017/files/config-3.14.26-yocto-qemu.config
$ make -j4 all
```

Finally make sure Yocto correctly boots up in Qemu, I ran the above commands to run qemu in debug mode then lauched, logged in to root and ran the following command to get the following response.

```
$ uname -r
$ 3.14.26ltsi-yocto-standard
```

II. QEMU Flags

- `-gdb tcp::5078` Open to connection with the GNU Debugger over TCP on port 5078. GDB can also use a custom protocol to communicate with qemu..
- `-nographic` Disable graphical output and redirect the emulated serial port to the console.
- `-kernel bzImage-qemux86.bin` Use bzImage as kernel image. The kernel can be either a Linux kernel or in multiboot format.
- `-drive file=core-image-lsb-sdk-qemux86.ext3,if=virtio` Define a new drive with the file `core-image-lsb-sdk-qemux86.ext3` over the virtio interface.
- `-enable-kvm`This option is only available if KVM support is enabled when compiling.
- `-net none` Don’t configure any network devices.
- `-usb` Enable USB driver.
- `-localtime` Use the system’s local time.
- `-no-reboot` Exit instead of rebooting.
- `-append "root=/dev/vda rw console=ttyS0 debug"` Use the provided commands as the kernel command line.

III. Concurrency Questions

What do you think the main point of this assignment is?

In assignment 1,there are two main points.The first point is how to build the Linux Kernel and run it by using QEMU. The second point is how to use pthread APIs and review the concept of producer-consumer problem.

How did you personally approach the problem? Design decisions, algorithm, etc.

Before I started this assignment, I reviewed pthread functions and concept of producer-consumer problem. After I had more ideas about how to finish code, I started to write program. I choosed a mutex to complete this program because semaphores are more useful when only a fixed number of threads can share a resource, while a mutex can only be held by one thread at a time. It also allowed me to stick to the pthread API without using the posix semaphore API. Next I defined the data structures for the project. I made sure to pick meaningful data types for the item structure within the bounds of the description of the assignment. In this case, a thread cannot sleep for a negative period of time, so using a signed integer for a waiting period is an unwise decision.

How did you ensure your solution was correct? Testing details, for instance.

I am sure that my solution was correct through printing many trace statements and values throughout the whole development process. To visually evaluate my solution, I print out the consumption value every time the consumer takes out an item. I also display the whole buffer for each iteration of the whole process so the user can visualize the consumer going through the entire buffer, index by index. Meanwhile the producer is right behind the consumer, generating a new value after a value is taken out. This solution ensures that the consumer never draws from an empty buffer and the producer never adds to a full buffer.

What did you learn?

This is my first time to use Latex. Although Latex is very complicated, I learned about hoe to use Latex build nifty citations with bibtex. Also, I learned about how to build PDF documents without `pdflatex`. I had the opportunity to review the pthreads API and reviewed concept of producer-consumer problem. I also got learned few new arguments for qemu. Moreover, I learned how to run vm by using terminal.

However, I do spend lots of time finishing this assignment.

IV. Work Log

Author	**Date**	**Message**
Yi-Jung Chiang	2017-04-18	Review producer-consumer problem and Pthread functions
Yi-Jung Chiang	2017-04-18	Implementing producer-consumer problem
Yi-Jung Chiang	2017-04-19	Write report and learn how to use Latex
Yi-Jung Chiang	2017-04-19	Write make file
Yi-Jung Chiang	2017-04-20	Build kernel
Yi-Jung Chiang	2017-04-21	Finish report