Building the Linux Kernel

CS453: Operating systems

Overview

For this homework we will be doing two activities:

1. Cloning the Linux Kernel sources, building/installing the kernel
    a. We are going to use the linux-stable tree directly from Linus

```
git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git
```

2. Making a small change to the maximum number of threads that can be created and then rebuilding/reinstalling the kernel.

This will give us the experience of working with a large code base (16+ million lines of code!) You will also gain an understanding of the structure of the code for a real operating system that is used on a billion plus devices around the world.

Due to security concerns we can't have students installing a new kernel on the onyx cluster as it could cause the cluster to crash which would impact other classes. You **must** do this assignment on you own machine. If you don't have access to a machine or your laptop/desktop is not sufficiently powerful enough to complete this assignment please contact your instructor ASAP.

Finally the bulk of your time is going to be spent just downloading and building the kernel. There is very little coding to do. But getting everything downloading and building can take hours. So don't wait to get started on this assignment. For example building the kernel in a VM with 2 cores and 4 gigs of RAM allocated could take as long as 2 hours. This is normal, large industry projects take a very long time to build. Once you have built the kernel once, it should only take a few minutes to rebuild a new kernel with changes.

Tasks

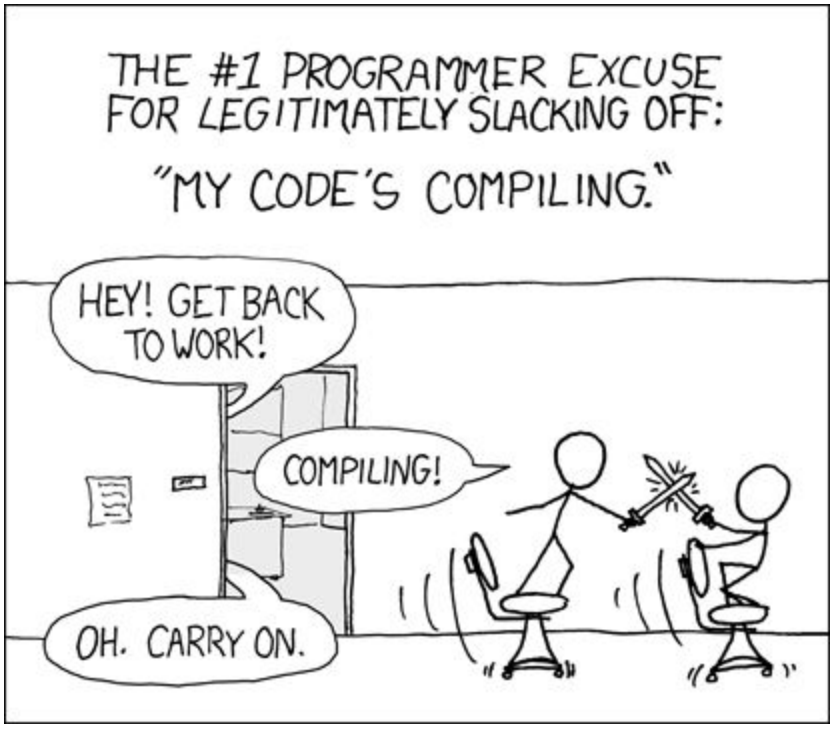
- Work through the provided kernel build instructions
- Note the value in **/proc/sys/kernel/threads-max**. You can see how many are allowed per process with the command **sudo ulimit -Ha** (we use sudo to run as superuser) and check for user processes entry.
- Modify the Linux kernel so it allows a user to run up to 95% of the maximum processes (instead of the default 50%).
- Rebuild and reboot into this kernel.

- You should now be able to create 95% of the maximum processes allowed (the value in **/proc/sys/kernel/threads-max**). Check with the command **sudo ulimit -Ha** and the value for user processes should now be higher. Hint: You will have to modify kernel/fork.c. Make sure to make a copy of the original before you modify it. The main function is **do_fork().**
- One strategy to find the correct location is to search for thread, threads, max, or /2
- Make sure to include the patch of code to fork.c that you used for the second part. Briefly, you can do this with the command:

  diff -Naur old-file new-file > fork-c.patch

Hints

- You will need at least 40G free to build the kernel. Make sure you have that much free space in your home partition! You should set your hard drive to be at least 100G to avoid running out of space.
- Give your virtual machine as much RAM as possible. So if your machine has 8 gigs of ram allocate 4-5 gigs to your Virtual Machine. This will improve the performance of the VM and make your kernel build faster.
- If you have an i5 or i7 processor allocate more than one core to the VM so you can get faster build times.
- Don't wait until the last minute to do this assignment. You will spend most of your time compiling the code. Remember the linux kernel has about 16+ million lines of code (you will only be compiling a subset of that 16 million lines however)!



Submission

Files committed to git (backpack)

Required files to submit through git (backpack), if you created more helper files that is fine.

1. Your patch file
2. Output of uname -a
3. README.md

Run the following commands

1. make clean
2. git add <file ...> (on each file!)
3. git commit -am “Finished homework”

Submit to Blackboard

You must submit the sha1 hash of your final commit on the correct branch. Your instructor needs this in order to troubleshoot any problems with submission that you may have.

- git rev-parse HEAD