CMSI 387-01

OPERATING SYSTEMS

Spring 2012

Assignment 0308

So here's the big thing — kernel building and modification! Enjoy the ride **:)**

Outcomes

This assignment will affect your proficiency measures for outcomes *1a*, *1b*, *2e*, *3a–3c*, and *4a–4f*.

Not For Submission

If you haven't already done so, read Chapters 1 and 2 in SGG.

For Submission

Figure out how to modify, compile, install, and demonstrate an Ubuntu 11.10 kernel. The core concepts behind this activity are generally the same across the board, for any flavor of any operating system. But as with many technical activities, the devil is in the details. And it *is* quite a devil. Put another way, this is perhaps the mother of all "Hello world!" variants.

The end of SGG Chapter 2 spells out a generalized set of steps for modifying and building a Linux kernel. This information is useful but will need to be adjusted, for reasons ranging from timeliness to the exact software involved.

The web is our ally here, but even there you will need to know what you're doing: you will find all kinds of information there, ranging from good to bad to ugly. But it's better than not having any information at all.

In broad strokes, the tasks you are to perform are:

- Install whatever prerequisites are necessary to modify, compile, install, and demonstrate an Ubuntu 11.10 kernel.
- Acquire the latest kernel source code.
- Configure, build, and install a new kernel from that code.
- Add a new system call to the kernel source code (for this assignment, a system call that emits "Hello world!" will suffice).
- Write and build a C program that demonstrates the existence and functionality of your new system call.

What to Turn In

To show that you did the work (and knew what you were doing), your deliverables are:

- One or more web pages, uploaded to *public_html/ubuntu-kernel-howto* on *my.cs.lmu.edu*, documenting *precisely* how to perform this task, from a vanilla Ubuntu 11.10 install to the finished product.

  Your web pages should be as "turnkey" as possible, meaning that a user who follows your instructions step by step should end up with the same results that you did. Screenshots or exact command input/output will definitely help here.

  Your web pages should also demonstrate your own knowledge of the process: include explanatory sections where necessary, and state what might vary (identifiers, names, labels) based on user preferences or situations.
- When all is said and done, you will have your own customized version of the kernel source code. Learn how to derive a *git patch* from your changed version, and commit that to *homework/kernel*. Remember, this is the *patch*, and not the entire source tree — that would be, um, unwieldy (to say the least).

For fun, we can do a web search after the deadline to see whose how-to page(s) are ranked highest based on the search term. Bragging rights to the highest-ranked submission **:)**