Kernel Compilation

Kernel Core: /usr/src/kernel/

arch/x86/entry/syscalls/syscall_64.tbl
add our syscalls to syscall table

include/linux/syscalls.h
define asmlinkage for our syscalls

SystemCalls/test_call.c
implementation of our test syscall
(if module attached, call function associated with STUB (syscallModule.c)

(kernel) Makefile
tell compiler to compile SystemCalls folder when compiling kernel

SystemCalls/syscallModule.c
defines functions for STUBs referenced in system call implementation (test_call.c)

SystemCalls/Makefile
compiles kernel module
compiles our syscall implementation

Kernel Compilation

- Because we modified files internal to the kernel to add our system calls, we must compile the entire kernel (~ hours)
- Once the kernel files are setup correctly, we will only need to recompile the kernel module during the rest of the development (~ seconds)
- Before compiling, make sure you added all 3 system calls as defined in the project description
  - You should NOT be adding a call named “test_call”

Development Tools

- Install tools needed to compile kernel:

  > sudo apt-get install

  - build-essential
  - libncurses-dev
  - bison
  - flex
  - libssl-dev
  - libelf-dev

Menuconfig

> make menuconfig

- Provides gui for editing kernel compilation config file
- Can remove unnecessary components to allow faster compilation, but make sure you don’t accidentally remove something you need
- If no changed desired, use esc to exit and save

Building

> sudo make -j $(nproc)

- -j flag allows you to specify number of processors to use during compilation
  - Dramatically speeds up computation time
- nproc provides the number of processing units available

Install

- sudo make modules_install
- sudo make install

- Installs the kernel
- Creates necessary boot/grub files

Reboot and Pray

- Confirm you are in your new kernel

  > uname -r (should return 4.19.98)

- You should only need to compile the kernel once to set up linkage / file pointers for new system calls

- Implementation of system call functionality is done in the kernel module which can be compiled independently as done in part 2