Kernel Emulator Implementation

To implement the kernel emulator, we want to reuse the kernel code as much as possible and modify the kernel code as less as possible.

Keeping those in mind, we wrapped the kernel code and run as a Linux process with the following modification:

- Provide a new kernel entry point, but we can reuse the boot code to emulate the system booting stage. (Collect the boot info and set up the kernel objects for the roottask, etc.)
- After the kernel boots and entring the userland, we will only trap into the kernel routine via interrupt or exceptions. (In seL4, interrupts and exceptions are all handled in one routine and system calls which are entered using syscall instructions will enter a fast syscall routine) So we need to provide wrapper functions for the kernel interrupt handling routine and syscall handling routine.
- We need to emulate privilege instructions or bypass them.