Analysing "Hello World" Roottask (cont)

How many seL4 syscalls needed by the helloworld roottask?

- **`seL4_SetTLSBase`** (set up the TLS region)
- **`seL4_DebugNameThread`** (needed if built with debug configuration)
- **`seL4_DebugPutChar`**

How to redirect those syscalls?

Dive deeper into how the above APIs are implemented. We can find they are actually wrappers around the raw seL4 syscalls. (e.g. x64_sys_send_recv() in syscall_syscalls.h) Those raw seL4 syscalls are wrapper around the ASM syscall instructions and they are **architecture dependent**.Other high level APIs provided by libseL4 eventually go here.

Hence we can redirect the system calls with the **minimal** modification in the original code by modifying the following raw syscall wrappers (Take x86_64 as an example).

`x64_sys_send`	invokes	**`seL4emu_sys_send`**
`x64_sys_reply`	invokes	**`seL4emu_sys_reply`**
`x64_sys_send_null`	invokes	**`seL4emu_sys_send_null`**
`x64_sys_recv`	invokes	**`seL4emu_sys_recv`**
`x64_sys_send_recv`	invokes	**`seL4emu_sys_send_recv`**
`x64_sys_nbsend_recv`	invokes	**`seL4emu_sys_nbsend_recv`**
`x64_sys_null`	invokes	**`seL4emu_sys_null`**