Analysing "Hello World" Roottask

Where to start with?

Let's start with inspecting a roottask printing "Hello World!". Notice that the following discussion will assume we use an x86_64 platform due to the current implementation. But we can extend the framework to be architecture independent without many changes.

What does a helloworld roottask need?

A runtime library

Set up the environment for an seL4 thread. (e.g. TLS region, POSIX syscall API redirection, IPC Buffer, etc.)

- seL4runtime

C library

Provide POSIX programming interfaces for the developer. (e.g. printf, open, read, etc.) Also the modified musllibc will redirect the POSIX syscalls to the handlers provided by seL4muslsys.

- musllibc
- seL4musllibcsys

seL4 system call library

Provide seL4 system call APIs.

- libseL4

seL4 kernel

Provide seL4 kernel functionalities and handle seL4 threads' requests.

- seL4 kernel