arm

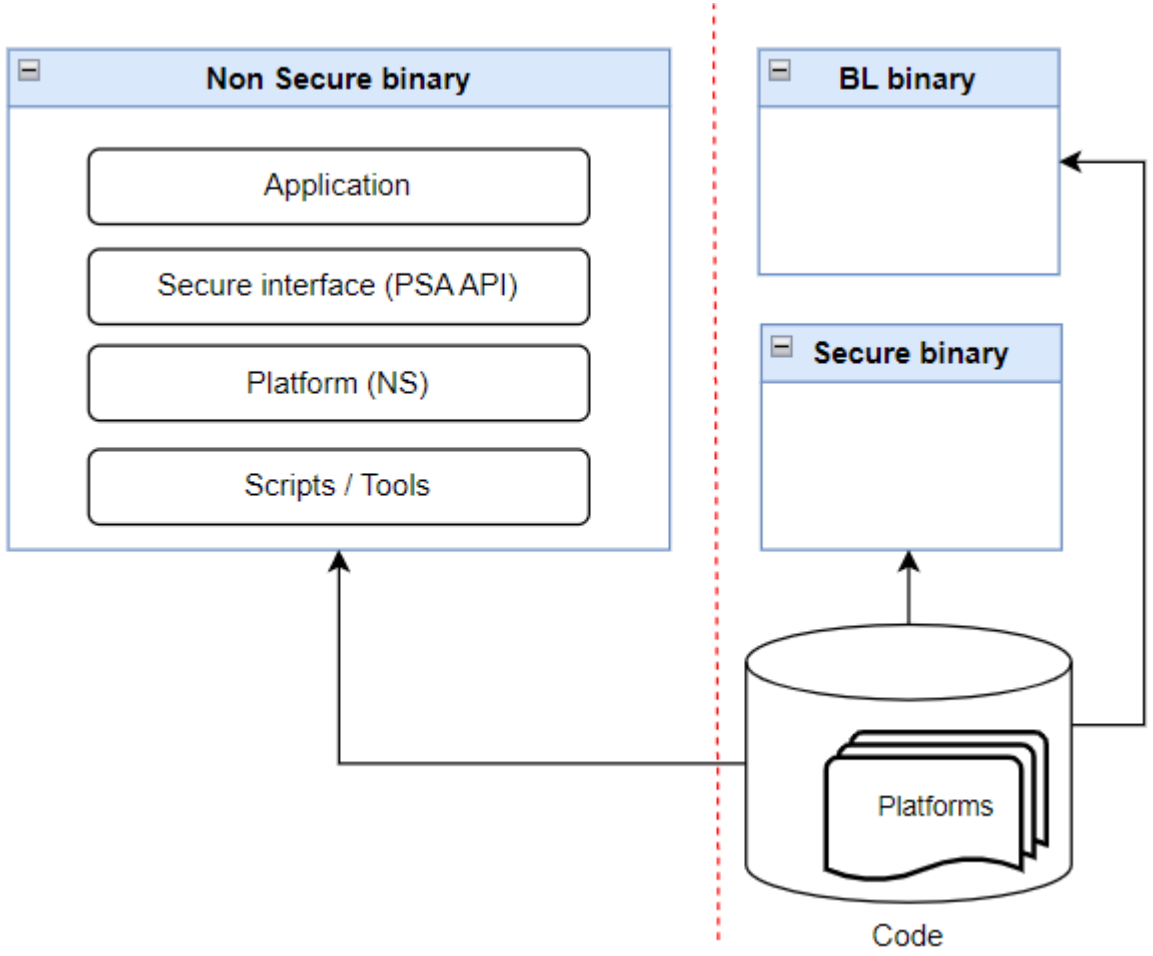
TF-M split build

continue

Anton Komlev
May 2023

Background and problem definition

- Building S, BL2 and NS sides together, starting from S. Reverted dependency
- Supporting all config combinations leads to :
  - Large and complex configuration set
  - Tricks in CMake build script to support different CPUs on S and NS
  - High entrance barrier for development either sides (S, NS)
  - Maintenance difficulty
  - Error-prone and vulnerable to side-effects
- Can we relax assumptions safely?
  - BL1, BL2 and S configurations are mainly defined by HW platform
  - Do not share codebase between NS and S directly



Split build alternative

2 semi-independent projects

TF-M = Secure side

- Mainly defined by HW platform. Highest priority
  - Memory layouts
  - Peripherals and drivers
  - CPUs
  - Platform sources for S and NS sides
- User defines
  - Partitions
  - Toolchain
- Outputs = exports = installs
  - PSA interface
  - BLs, S binaries
  - Bin image tools (signing, merging)
  - NS platform code
  - NS toolchain

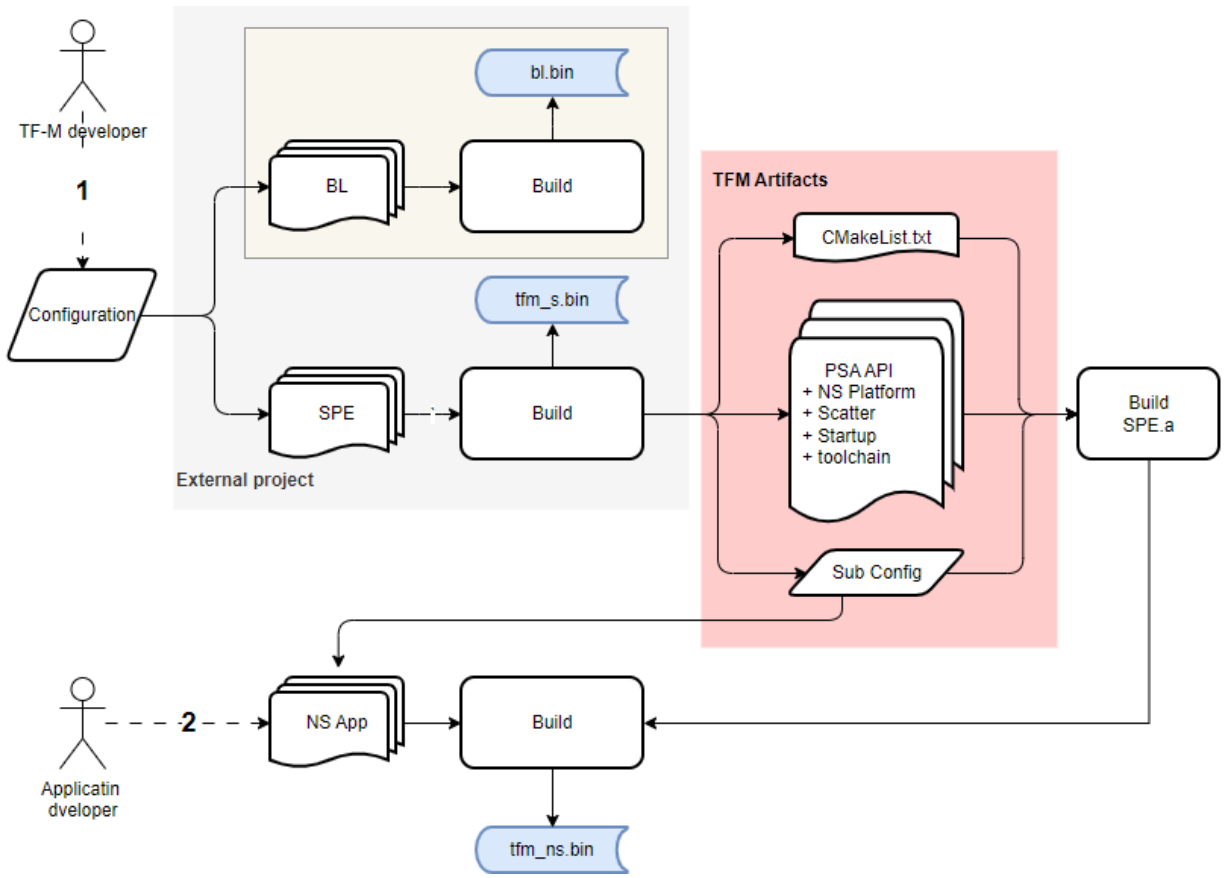
Platform specific

Application = Non-Secure side

- An application code
- Builds and links with NS platform sources
- Combines with BL, S binaries
- BLs and S
  - Stays the same
  - OEM can ship it in binaries

Implementation

- Extend installation script (install.cmake) to export
  - Common NS platform files
  - CMakeLists.txt for SPE
- A Platform shall export
  - NS platform sources
  - Linker script (Scatter)
  - Startup file
- Initially this is non-intrusive, optional feature.
  - The legacy way supported in parallel
  - Each platform can support the new building way independently
  - Small change in code. Main change in approach
- PoC is here: https://review.trustedfirmware.org/c/TF-M/trusted-firmware-m/+/20960



Pros & Cons

Pros

- Hides S side complexity and simplify TF-M application (NS) development
- Native separation of compile flags for S and NS
  - Clean Cmake scripts
  - Floating point support
  - usage of V8.1 PAT-PCI feature
- NS side does not worry on BL1, BL2, S
- Simplify IDE support
- Works in parallel with the current build process

Cons

- Change build process. Build S and NS separately
- ?

Questions / issues

- What is the option set for export to NS side?
  - Global or platform specific?
- Need headers (*.h) dependency cleanup.
  Headers are mixed/coupled between S and NS sides.
- include paths in some files
  - #include **platform/include/**tfm_plat_defs.h
  - #include **timer_cmsdk/**timer_cmsdk.h
- Need to stub some CMake targets for compatibility. Ex:
  - platform_region_defs
  - platform_common_interface
  - tfm_fih_headers
  - tfm_ns_interface
- Generate Platform_region_defs targets for NS by settings on ?

arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה