Trusted Firmware - M

Secure Partition Runtime Library Update

Summer Qin

2020.07.09

Contents

- Background – Brief Summary about SPRTL
- Updated Design – What's new
- Forecast – What can be added

Background - Brief summary about SPRTL

- The Secure Partition Runtime Library (SPRTL) is a shared library in SPE for Secure Partition runtime usage. In the initial design, it is put in a shared but all read-only region which can support limited function types, like “memset”, “memcpy”.

- Other SPRTL functions such as malloc() needs to access partition private data in an implicit way.

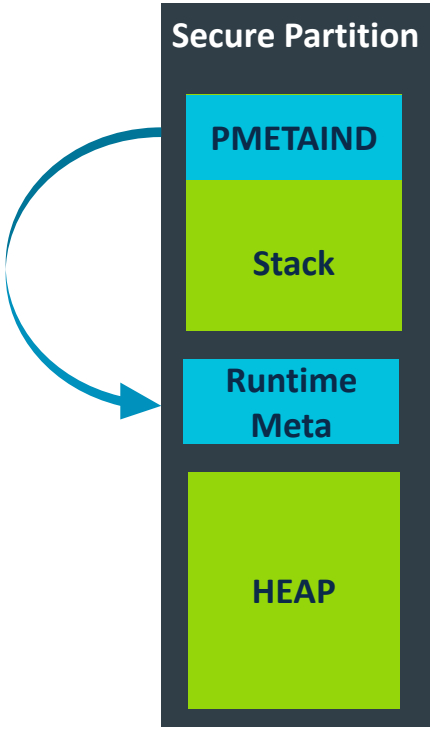
```
void *malloc(size_t n)
{
        return heap_alloc(this_partition_heap_inst, n)
}
```

Background - Solutions for functions with implicit parameters

- Solution 1: Put partition meta indicator at stack higher boundary

PMETAIND = (get_sp() & MASK) + STACK_SIZE – sizeof(PMETAIND);

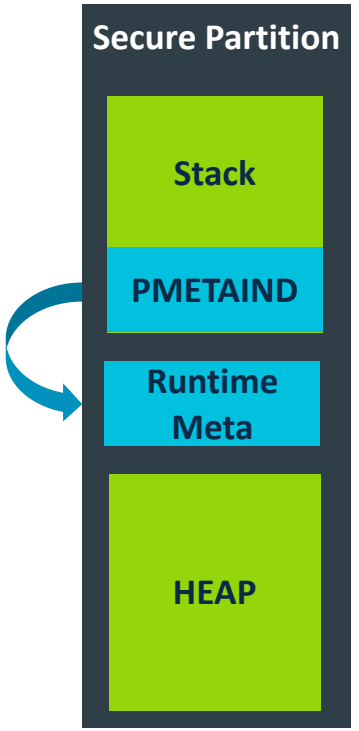
  - Due to stack address is not aligned on M-profile, cannot get meta address by tricks.
  - Unprivileged execution hard to get the stack pointer.



arm

Background - Solutions for functions with implicit parameters

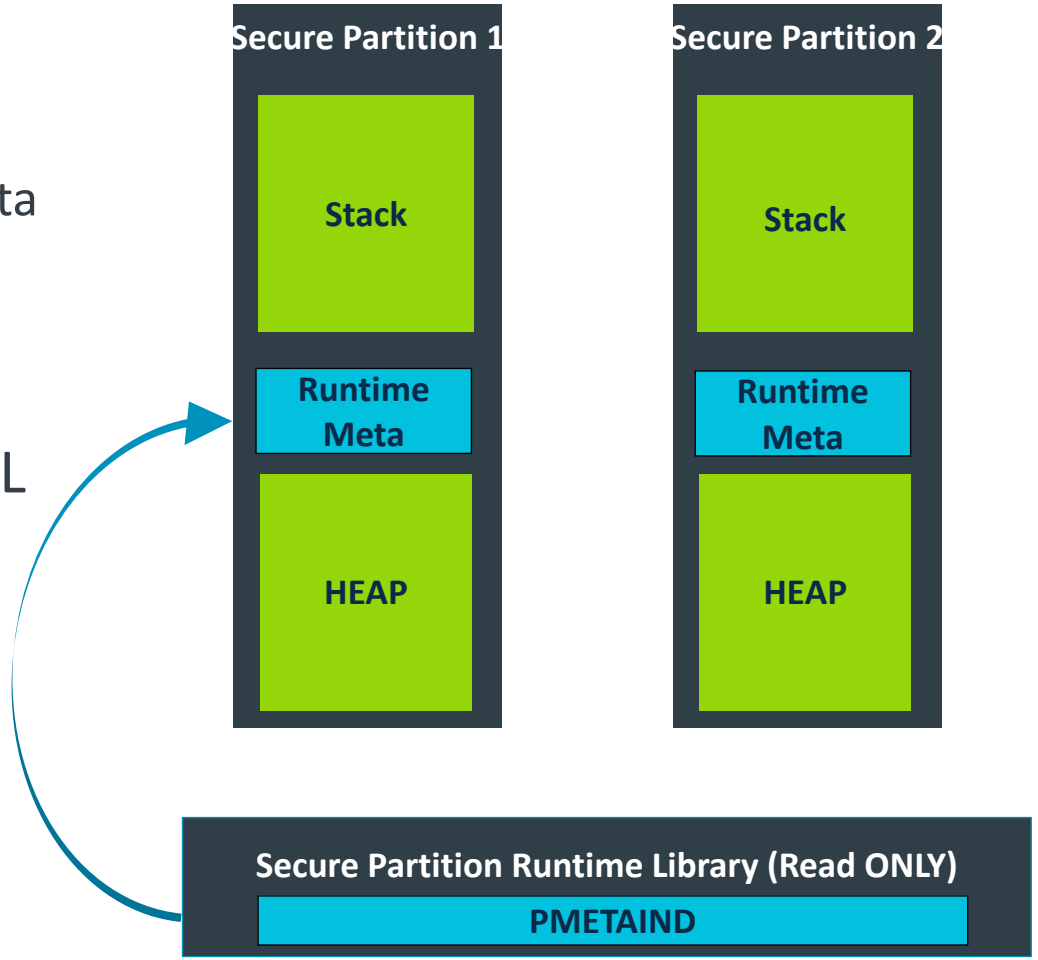
- Solution 2: Put partition meta indicator at stack lower boundary
  - Unprivileged execution hard to get the stack limit.
  - v6m, v7m does not have PSPLIMIT.



Updated Design - Add a specific indicator

- Partition runtime meta
  - sprtl_runtime_t: Per partition object in private data

- Define one read-only global variable in SPRTL
  - PMETAIND
  - A SPRTL visible and read-only variable for easy retrieving.
  - Pointing to the Partition's metadata.
  - Updated by SPM while scheduling.



arm

Updated Design - Runtime Partition Entry Wrapper

- A common partition entry wrapper (sprt_main) is required:
  - Mentioned in the first version of SPRTL design but didn't have chance to apply.
  - Do runtime initialization for partition (sprt_heap_init e.g.);
  - Then jump to developer provided actual partition entry.
  - "Invisible" to service developer's service code scope.
  - Tooling reports this as partition entry in the SPM partition instance, while actual partition entry is saved in metadata (Still a tooling behavior).

```
void sprt_main(void)
{
    struct sprt_runtime_t *meta = (struct sprt_runtime_t *)PMETAIND;

     sprt_heap_init(meta->heap_sa, meta->heap_sz);

    /* Call thread entry 'entry_point' */
    meta->thread_entry();

    /* should never return*/
}
```

Updated Design - Prototype links

https://review.trustedfirmware.org/c/TF-M/trusted-firmware-m/+/4546

https://review.trustedfirmware.org/c/TF-M/trusted-firmware-m/+/4547

https://review.trustedfirmware.org/c/TF-M/trusted-firmware-m/+/4644

https://review.trustedfirmware.org/c/TF-M/trusted-firmware-m/+/4647

Forecast - What can be added

- Other implied operations of partitions.

arm

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