seL4: Untyped

廖东海

ctrlz.donghai@gmail.com

Background

- Physical memory
  - 除了一小部分静态的内核内存，seL4 所有的物理内存都在用户态被管理。
  - 在启动时由 seL4 创建的对象的 capability ，以及由 seL4 管理的其他的物理内存资源，在启动后都会传递给 root task。
- Untyped memory
  - Untyped memory 是一块特定大小的连续物理内存块，表示未被使用。
- untyped capabilities
  - 是 untyped memory 的 capability。
  - untyped capabilities 可以被 retyped。

Initial state

- seL4_BootInfo 对 root task 描述了所有的 untyped capabilities，包括他们的大小，是否是 device untyped，以及对应的物理地址。

```
    printf("    CSlot    \tPaddr           \tSize\tType\n");
    for (seL4_CPtr slot = info->untyped.start; slot != info->untyped.end; slot++) {
        seL4_UntypedDesc *desc = &info->untypedList[slot - info->untyped.start];
        printf("%8p\t%16p\t2^%d\t%s\n", (void *) slot, (void *) desc->paddr, desc->sizeBits, desc->isDevice ? "device untype
d" : "untyped");
    }
```

- untyped capabilities 有一个 bool 的属性，指明对应的内存对象是否可以被内核写
  - 内存对象可能不是 RAM 而是其他 Device，或者它可能位于内核无法寻址到的 RAM 区域。

Initial state

- seL4_BootInfo 对 root task 描述了所有的 untyped capabilities，包括他们的大小，是否是 device untyped，以及对应的物理地址。

```
Booting all finished, dropped to user space
    CSlot       Paddr                   Size    Type
   0x137                   0           2^20    device untyped
   0x138          0x1ffe0000           2^17    device untyped
   0x139          0x20000000           2^29    device untyped
   0x13a          0x40000000           2^30    device untyped
   0x13b          0x80000000           2^30    device untyped
   0x13c          0xc0000000           2^29    device untyped
   0x13d          0xe0000000           2^28    device untyped
   0x13e          0xf0000000           2^27    device untyped
   0x13f          0xf8000000           2^26    device untyped
   0x140          0xfc000000           2^25    device untyped
   0x141          0xfe000000           2^23    device untyped
   0x142          0xfe800000           2^22    device untyped
   0x143          0xfec01000           2^12    device untyped
```

- untyped capabilities 有一个 bool 的属性，指明对应的内存对象是否可以被内核写
  - 内存对象可能不是 RAM 而是其他 Device，或者它可能位于内核无法寻址到的 RAM 区域。

Retyping

- 只要untyped capability对应的memory大小允许，memory可以被拆分成许多新的object。
- 使用seL4_Untyped_Retype函数对untyped capability进行创建新的capability。
  - 同时也是会对相应的untyped object进行retype。
  - 新的capability是原来untyped capability的child。
  - children按顺序获得有效内存，且内存是对齐的。
  - 例如，创建4k object以后创建16k object，这样有12k就被浪费了。

```
error = seL4_Untyped_Retype(parent_untyped, // the untyped capability to retype
                            seL4_UntypedObject, // type
                            untyped_size_bits,  //size
                            seL4_CapInitThreadCNode, // root
                            0, // node_index
                            0, // node_depth
                            child_untyped, // node_offset
                            1 // num_caps
                            );
```

Object Type

- seL4 中的每个对象都有指定的类型，所有类型的定义可以在 libsel4 中找到。
  - API Object：
    - seL4_UntypedObject
    - seL4_TCBObject
    - seL4_EndpointObject
    - seL4_NotificationObject
    - seL4_CapTableObject
  - 有些类型是硬件架构特定的。
- Untyped Object Type
  - Untyped对象用于转化为其他类型的对象。
  - Untyped也可以转化为小的Untyped对象。
  - Child untyped对象可以进一步retype。

Object Size

- size 参数指定新对象的大小。这个参数对不同的对象类型有不同的含义
  - 大多数 seL4 对象是固定大小。对此，该参数会被忽略。
  - seL4_UntypedObject，允许可变的大小， 指定的对象大小为 2^{size}字节。
  - seL4_CapTableObject 是可变大小， size 指定的是 Slot 的数量，数量为 2^{size} 个。
  - 可以通过Revoke回收之前分配出去的Untyped Object。
- Root, node_index & node_depth
  - 用于指定目标Cnode。
- Node_offset
  - 用于指定目标CSlot。
- Num_caps
  - seL4_Untyped_Retype 接口可以一次性 retype 生成多个 capabilities ，生成 cap 的数量由这个参数指定。

seL4: Mapping

廖东海

ctrlz.donghai@gmail.com

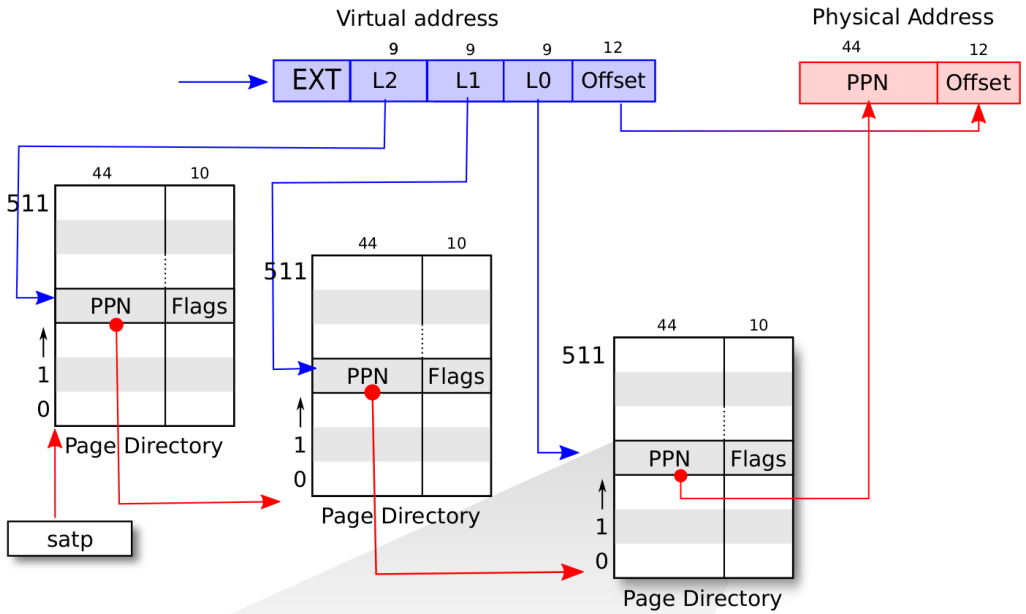
Background

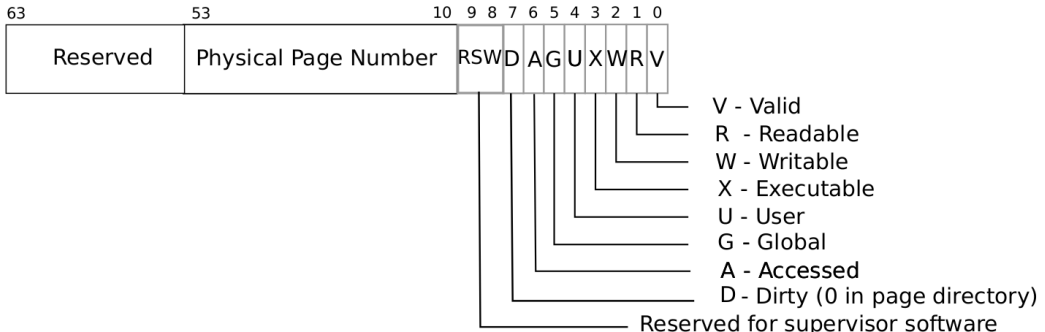
- Virtual memory
  - 在现代处理器结构上，为了满足进程间编址的独立性和地址空间的隔离性，以及物理内存对应用程序的透明性，引入虚拟内存系统。
  - 程序中使用的地址均为虚拟地址，有CPU硬件单元MMU根据软件设置的页表结构自动翻译为物理地址。
  - 软件需要维护的仅仅只有页表结构。
- seL4 不提供除简单的硬件页表机制外的其他虚拟内存管理，用户态必须提供创建中间分页结构、映射、和取消映射的服务。
- seL4 的应用程序可以自由定义它们自己的虚拟地址空间布局，只有一个约束：seL4 的内核占用了高段虚拟地址范围。在大多数的32位平台，高段指的是：0xe0000000 之上。这个变量是根据硬件平台来设置，可以在源代码中搜索 kernelBase 变量来查找。

RISCV-SV39

- SV39 多级页表的硬件机制 - rCore-Tutorial-Book-v3 3.6.0-alpha.1 文档 (rcore-os.cn)
- Q: 为什么seL4 kernel对

内核页表的设置没有三层？





API

- seL4_RISCV_PageTable_Map
  - 映射二级页表
- seL4_RISCV_Page_Map
  - 映射页表项
- seL4_RISCV_PageTable_Unmap
  - 解映射二级项
- seL4_RISCV_Page_Unmap
  - 解映射页表项

Thanks for Listening.