Nested Kernel

- I1：外核执行时，受保护数据的有效虚拟到物理映射配置为只读。
  - 通过三步执行 I1
    - 要求显示初始化 PTP
    - 创建更新 PTE 的显式接口
    - 将所有 PTP 对应的 PTE 设置为只读
  - I3：确保在外核执行前没有未验证的映射。
  - I4：只有声明的 PTP 可用于映射。 — nk_declare_PTP 用于声明 PTP，nk_write_PTE 中检查是否声明。
  - I5：所有到 PTP 的映射设置为只读。 — nk_write_PTE 确保所有 PTP 都是受写保护的
  - I6：CR3 仅加载预先声明的顶级 PTP。
    - nk_load_CR3 确保加载到 CR3 的都是声明过的顶级 PTP。
    - 在外核执行时将该指令页取消映射，仅在需要时映射。

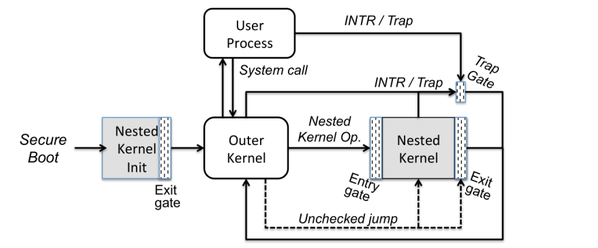

Figure 1. PerspicuOS State Transition Diagram. Only shaded blocks can execute PerspicuOS privileged operations (Table 2). All transitions out of the nested kernel must go through the Exit Gate.

图中 Nested Kernel Init 模块，初始化分页系统并将受保护页注册到嵌套内核数据结构，然后在执行前开启 PG 和 WP。

- I2：外核执行时，强制执行有效虚拟到物理映射的写保护。
  - 利用 x86 硬件支持执行写保护 — 外核可能通过禁用 WP、禁用 PG 和劫持控制流等方式关闭保护。
  - I7：外核执行前，CR0.WP&PG 要置位。 ← 执行I7
  - I8：CR0.WP 不会被外核代码禁用。
  - I9：禁用 PG 后控制流会转到嵌套内核。
  - I10：嵌套内核控制 SMM 中断处理和执行。 — 未实现
  - I11：调用外核的中断/陷阱处理程序前，要启用 WP。
  - I12：IDT 必须是被写保护的，IDTE 只能由嵌套内核更新。
  - I13：嵌套内核堆栈对外核的修改是写保护的。

- 虚拟特权级切换
  - 嵌套内核与外核在同一地址空间，嵌套内核操作由禁用和启用 WP 的入口和出口门包装

```
entry:
  pushfq                          Save current flags
  cli                             Disable interrupts
  mov %rax, -8(%rsp)              Spill regs for temps
  mov %rcx, -16(%rsp)
  mov %rsp, %rcx                  Save stack ptr in rcx
  mov %cr0, %rax                  Get current CR0 value
  and ~CR0_WP,%rax                Clear WP bit in copy
  mov %rax, %cr0                  Write back to CR0
  cli                             Disable interrupts
  mov PerCPUSecureStack,%rsp      Switch to secure stack
  push %rcx                       Save orig stack ptr
  mov -0x8(%rcx), %rax            Restore spilled regs
  mov -0x10(%rcx), %rcx
```

Figure 2. Nested Kernel Entry.

入口门，关中断->关 WP->管中断->切换到安全的嵌套内核堆栈

```
exit:
  mov 0(%rsp), %rsp               Restore orig stack ptr
  push %rax                       Spill scratch reg
  mov %cr0, %rax                  Get current CR0 value
1:
  or CR0_WP, %rax                 Set WP in CR0 copy
  mov %rax, %cr0                  Write back to CR0
  test CR0_WP, %eax               Ensure WP set
  je 1b                           If not, loop back
  pop %rax                        Restore clobbered reg
  popfq                           Restore flags
                                  (incl interrupt status)
```

Figure 3. Nested Kernel Exit

出口门，相反操作

  - 在嵌套内核执行时禁用中断 — 功能较少，不会影响性能
  - 外核的中断处理程序可能在禁用 WP 时运行
    - 隔离 x86 IDT，配置所有中断和陷入都先经过嵌套内核中的中断门，设置 WP 后才将控制流转移到外核处理程序
    - 中断门使用类似于退出门的循环检查确保启用 WP
  - 嵌套内核使用单独的堆栈，进入和退出时保存和恢复外核栈指针

执行I12
执行I11
执行I13
执行I8
执行I9

- 内核代码完整性
  - 外核代码加载时验证，确保不含受保护指令
    - 修改 PTE、PTP
    - 修改 CR0、CR3、CR4、MSR
  - 动态生命周期外核代码完整性，配置处理器和 pMMU
    - 验证后的代码页映射为只读
    - 默认设置所有内核页是不可执行的，EFER.NX
    - 用户代码和数据映射为特权模式不可执行，CR4.SMEP

Operation	x86 Instruction	Description	Constraints
nk_declare_PTP	None	Initialize physical page descriptor as usable in page tables	Asserting invariant I4
nk_write_PTE	mov VAL, PTEADDR	Update *p*MMU mapping	Asserting invariants I4 and I5
nk_remove_PTP	mov VAL, PTEADDR	Remove physical page from being used as PTP	Supporting invariants I4, I5, and I6.
nk_load_*CR0*	mov %REG, *%CR0*	Controls enforcement of read-only mappings	*WP-bit* must be set: invariant I8
nk_load_*CR3*	mov %REG, *%CR3*	Controls MMU mapping base PML4 page	Value must be a declared PML4-PTP
nk_load_*CR4*	mov %REG, *%CR4*	Controls user mode execution with *SMEP* flag	*CR4 SMEP* flag must be 1
nk_load_MSR	wrmsr Value, MSR	Control enforcement of no-execute permissions	*EFER* NX-Bit must be set to 1

Table 2. Nested Kernel Operations, *Protected Instructions*, Description, and Constraints

将外核中的相关指令替换为调用嵌套内核操作，隔离对 pMMU 的访问

- 特权寄存器完整性
  - 外核执行时取消受保护指令的映射，仅按需映射
  - 但对于 CR0，仍存在于嵌套内核的入口和退出门
    - 不需要验证入口门中的 CR0，因为其目的就是禁用 WP，以执行嵌套内核
    - 对于退出和中断门，使用循环检查确保启用 WP
  - 将入口门虚拟地址与一段包含陷入到嵌套内核的代码的物理地址相匹配 — 关闭分页后控制流转移到嵌套内核

已确保外核代码无法修改特权寄存器，但要考虑外核跳转到嵌套内核指令执行