CPU Isolation & IPI interference

Valentin Schneider <vschneid@redhat.com>

LPC 2025

Red Hat

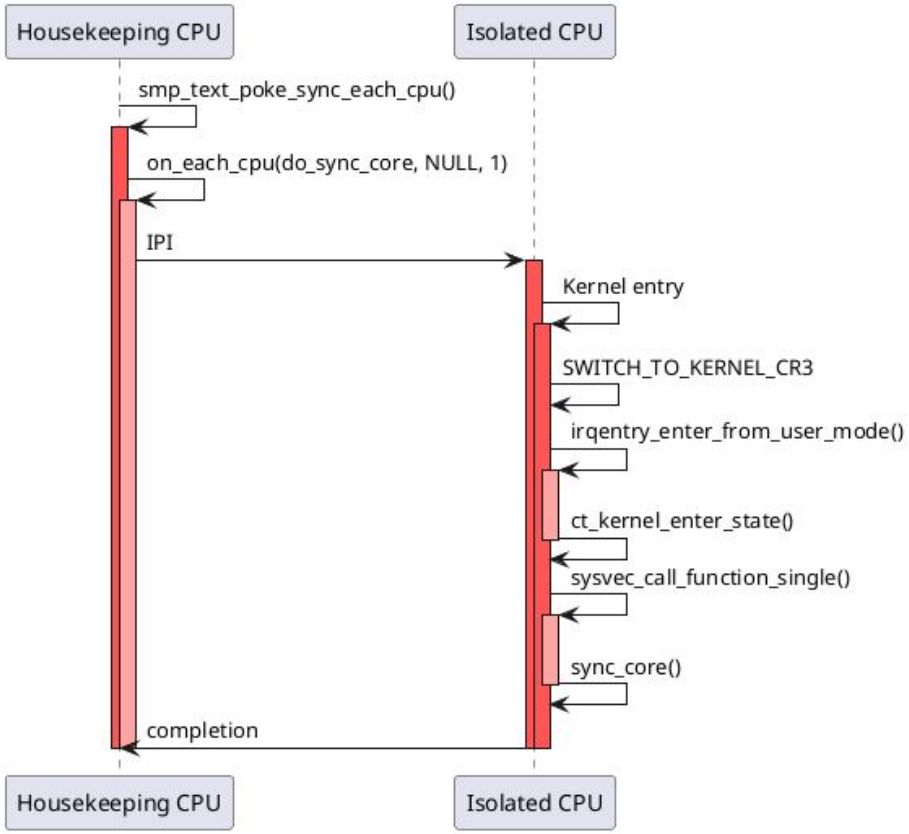
Context

- CPU Isolation, NOHZ_FULL, RCU_NOCB…
  - Single userspace task on **isolated** CPU
  - No (voluntary) kernel entry
- Some **IPIs** still end up hitting the **isolated** CPU
  - **smp_text_poke()** (**static keys** & friends)
  - **vunmap()**'s **flush_tlb_kernel_range()** (freeing / unmapping)
- Deferral concept: IPI **doesn't concern userspace**?
  - Don't send it
  - Execute related callback ASAP upon kernel entry

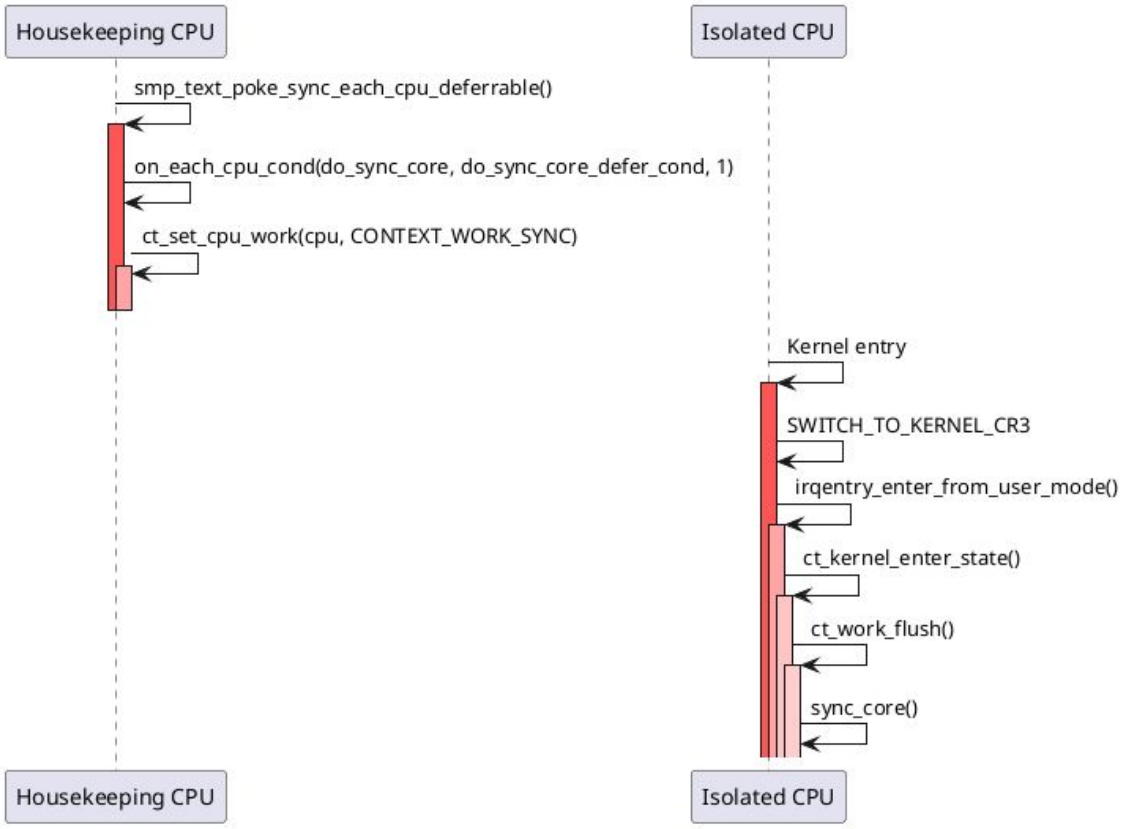
The danger zone

- Operation is /!\ not **immediately** executed upon kernel entry /!\
- Applies currently, but deferral has to deal with being completely asynchronous

Current approach



Deferral



Status

- V7: https://lore.kernel.org/lkml/20251114150133.1056710-1-vschneid@redhat.com/

- Deferral for **smp_text_poke()** looks about OK
- Deferral for **flush_tlb_kernel_range()** not so much...

Status

- Deferral for `flush_tlb_kernel_range()` not so much...
  - `ct_set_cpu_work()` approach
    - Danger zone: accessing unmapped kernel pages
    - AndyL says it may work
  - `SWITCH_TO_KERNEL_CR3` hackery
    - No danger zone per se
    - A big eyesore
- Newer hardware can do this
  - AMD INVLPGB; Zen3 and later; supported as of v6.15
  - Intel RAR; patches out there but not yet merged
- Not all architectures have this problem (e.g. arm64)

Thank you!


linkedin.com/company/red-hat


youtube.com/user/RedHatVideos


facebook.com/redhatinc


x.com/RedHat

Red Hat