東京 2025

LINUX PLUMBERS CONFERENCE

TOKYO, JAPAN / DECEMBER 11-13, 2025

CoCo MC: Optimizing guest_memfd conversions

Contact ackerleytng@google.com if you have any questions/suggestions!

東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

What we'd like from the community

- Your thoughts on how guest_memfd should manage mappings in IOMMU page tables during conversions
- Any other ideas for improving conversion performance

Introduction to guest_memfd

- guest-first memory provider built for KVM
- The only supported memory provider for private memory for CoCo VMs

- Core principle: private guest memory must not be accessible by the host
  - Remove or disallow mappings in host page tables

guest_memfd: huge pages and sharing

- guest_memfd gets huge pages and takes ownership of them
    - HugeTLB [1]
    - Best-effort huge pages (like THP) [2]

- Guest requests to share some of its memory
- guest_memfd splits folios backing that memory
    - Why: to track users using struct page refcounts

[1] https://lore.kernel.org/all/cover.1747264138.git.ackerleytng@google.com/T/

[2] https://lore.kernel.org/all/20241212063635.712877-1-michael.roth@amd.com/T/

Problem: folio splitting is expensive

- Time cost: work that needs to be done:
    - Split filemap entries
        - Allocations! (XArray entries)
    - Undo HugeTLB vmemmap optimization (HVO)
        - More allocations! (Pages to store struct pages)
    - Split folios (copy flags, etc)
- Memory cost in allocations

Don’t restructure folios

Just manage mappings and users of memory.

Proposal: components

- Manage mappings: guest_memfd calls into IOMMU code to perform unmapping
- Manage (actually, limit) users: guest_memfd exposes VM_PFNMAP VMAs, removes struct pages

Manage mappings: in guest_memfd

- guest_memfd manages mappings in Stage 2 page tables
  - Tell KVM what level to map at
  - Tell KVM to unmap pages during truncation/conversions
- On list: guest_memfd removes pages from kernel direct map [1]
- On list: guest_memfd manages mappings in host userspace page tables
  - Unmap pages during conversions [2]
  - Tell core-mm what level to map at
- Discussions: guest_memfd to unmap from IOMMU page tables
  - Support mapping private memory for Confidential IO [3]

[1] https://lore.kernel.org/all/20250924151101.2225820-1-patrick.roy@campus.lmu.de/T/
[2] https://lore.kernel.org/all/cover.1760731772.git.ackerleytng@google.com/T/
[3]
https://lore.kernel.org/all/CAGtprH_qh8sEY3s-JucW3n1Wvoq7jdVZDDokvG5HzPf0HV2=pg@mail.gmail.com/

Manage mappings: IOMMU page tables

- Why the requirement? Ensure devices don't write to guest private memory
  - guest_memfd forces IOMMU to unmap shared pages on conversion to private
  - and forces IOMMU to unmap private pages on conversion to shared
- How?
  - IOMMU gets pages directly from guest_memfd, gives guest_memfd a handle to IOMMU
  - Maximum flexibility: map memory using fd+offset as a reference: no requirement to mmap() guest_memfd

Manage mappings: Insufficient

- Mappings are not the only source of “users”, GUP indicates usage
- Scenario
  - GUP-ed page 10 in a huge page => refcount on huge page increased
  - guest wants to convert page 2
  - Should guest_memfd permit the conversion?

Manage users: VM_PFNMAP VMAs + drop struct pages

- Why VM_PFNMAP?
  - Because VM_PFNMAP VMAs don't allow GUP
  - Instead of having guest_memfd force users to stop using pages (like unmapping), don't let them use guest_memfd pages to start with.
- Why drop struct pages?
  - Seal the deal: nobody can hold refcounts if the struct page doesn't exist
  - (requires changes within guest_memfd, KVM and platform code to not rely on struct pages)
- How to drop struct pages? If requested with guest_memfd flag,
  - Hot-unplug allocated memory to remove page structs
  - Hot-plug them on removal from guest_memfd ownership

Manage users: Can virtualization avoid GUP?

- Virtualization increasingly offloaded, IO done through IOMMU
- If guest_memfd has a handle on IOMMU page tables, can the remaining users of GUP be moved away from GUP?
- How necessary is GUP for virtualization, going forward?

Revisit: do we actually need to drop struct pages?

- Can VM_PFNMAP be correctly used with memory that is described by struct pages?
  - What are the pitfalls of VM_PFNMAP while struct pages exist?
- What types of folios be VM_PFNMAP be applicable to? Would ZONE_DEVICE folios be okay?

Thank you!

- Please feel free to reach out to discuss anything about guest_memfd!
- I'm available in the hallways or at ackerleytng@google.com.

東京 2025

LINUX PLUMBERS CONFERENCE

TOKYO, JAPAN / DECEMBER 11-13, 2025