Memory Failure Handling for guest_memfd+HugeTLB

For 2025-07-17 guest_memfd bi-weekly upstream call

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

How is it handled for 4K pages (pre-HugeTLB support)?

- Starting point in the kernel is the #MCE handler, leading up to `memory_failure()`
- `memory_failure()` marks folios with HWpoison
- Without mmap support in guest_memfd, 4K pages are only used for private pages (“not mapped” from host perspective)
  - `memory_failure()` hands the folio to `kvm_gmem_error_folio()`, which will unmap the pages from stage-2 page tables
  - On the next private fault, `__kvm_gmem_get_pfn()` checks for HWpoison and returns `-EHWPOISON`, which goes all the way to userspace.

How is it handled for 4K pages (post-mmap support)?

- Same if 4K pages are used for private pages
- If 4K pages are used as shared pages,
  - If pages are mapped, the userspace VMM will be killed with SIGKILL
    - SIGKILL will probably lead to freeing of the folio on guest_memfd inode release, so folio will be freed with HWpoison flag
    - `memory_failure()` continues to `kvm_gmem_error_folio()`, which will unmap the pages from stage-2 page tables, which will unmap shared pages too, for a non-CoCo VM.
      - David: Only get a SIGKILL if try to unmap, but failed to unmap, so it's okay, aligned with everything else (should double-check)
      - Ackerley (after meeting): My bad. If pages are mapped
        - But not faulted in, collect_procs() won't pick it up so no signal will be sent to those processes
        - And faulted in, collect_procs() will pick up the process, but in kill_procs(), SIGKILL is only if for folios not faulted in (not even on the process list). SIGBUS will be sent for processes on the list
        - So for all shared and private pages, mapped or not, userspace VMM will get a SIGBUS
          - If faulted, the SIGBUS will be from memory_failure()
          - If not faulted, the SIGBUS will be from __do_fault() (later)
  - If 4K pages are not mapped, no SIGKILL.
    - Continue to `kvm_gmem_error_folio()`, which will unmap the pages from stage-2 page tables
    - On next fault (if there is a fault), `__do_fault()` (outside of guest_memfd) will discover HWpoison and SIGBUS the userspace VMM

Why handle specially for gmem+HugeTLB?

- HugeTLB support implies conversion support
- Conversion support implies runtime folio restructuring (split/merge)
- Memory failure can happen at any time during guest_memfd lifetime
  - Race-related inconsistencies when marking folios poisoned
  - E.g. if `PGTY_hugetlb` is removed but folio is not yet split, `memory_failure()` would think it's a THP page when it is not

Requirements

1. Guest(s) should continue to run for as long as possible
2. Userspace VMM must know exactly which PFN failed
3. Don’t let memory failure handling slow down regular operations (like conversion)

Proposal

1. In `memory_failure()`, identify guest_memfd folios, handle them separately from any other type of folio.
2. Handle memory failure:
    a. If bad memory was detected but not yet consumed, do nothing, defer all handling till consumption
    b. If `memory_failure()` is entered because bad memory was consumed, SIGBUS userspace VMM
    c. Dan: detected but not consumed => unrecoverable, should handle
        i. Vishal: implement in future
        ii. David: mm alignment session: memory failure handling for HugeTLB
            1. Record error on page, keep it mapped, inject/virtualize MCE
            2. Opt-in system to different memory failure policies
    d. Jiaqi: userspace memory failure is mostly about unmap or not. Want to SIGBUS
        i. To kill early, KVM needs new mechanism to look up relevant process
    e. Callout: No unmapping in `kvm_gmem_error_folio()`

Discussion parts

1. How to identify a guest_memfd folio
2. guest_memfd memory failure handling

[Handling] Why SIGBUS?

- SIGBUS, unlike SIGKILL, can be handled (can install signal handler)
  - Memory error matches definition of SIGBUS
- Using a signal allows reporting precisely the (virtual address of the) PFN of bad memory
  - If VM supports #MCE injection, userspace VMM can virtualize memory failure
- Signal vs KVM exit: handle without unmapping the page

[Handling] Why not unmap?

- Handling when bad-memory-not-yet-consumed: don’t unmap to allow guest to continue using memory until the specific part of the page with bad memory is consumed
  - Big difference if the folio is a HugeTLB folio, smaller difference for split folios
- Handling on consumption: No point unmapping
  - Successful #MCE virtualization (non-CoCo and SNP VMs): guest will avoid bad memory anyway
  - Cannot virtualize #MCE: (e.g. TDX)
    - Bad private memory: TD is already torn down
    - Bad shared memory: Replacing memory won’t replace contents, so TD won’t function anyway.

[Identification] How to identify a guest_memfd PFN?

- ~~page type~~
  - Can’t use this when mapcount > 0
- ~~folio->mapping~~
  - May race with truncation
- Global data structure containing all guest_memfd PFNs

Tracking of failed PFNs when returning to HugeTLB?

- Defer till next #MCE (aka don’t track)
  - Less work
  - IIUC HugeTLB doesn’t dissolve poisoned folios when freeing (only on demotion), so deferring gives system a chance to dissolve poisoned folios at next #MCE
- Track and return folio with HWpoison to HugeTLB on freeing from guest_memfd
  - HugeTLB folio may float around with HWpoison, entire folio can’t be used until demotion
- David: Set when returning to HugeTLB, let HugeTLB improve on that

How to track failed PFNs?

- Global set of failed PFNs
  - Doesn't complicate folio restructuring
- Use HWpoison flag in folio
  - Need different processes for HugeTLB vs split folios
  - Need to lock out restructuring
- Vishal: For lifetime of guest_memfd, still keep handing out errored folios (but not poisoned)
- David: mark hwpoison, summarize hwpoison and merge
  - But can't fault in hwpoisoned folio
- Jiaqi: hwpoison, but tell core-mm/kvm to continue faulting it in
  - David: Address space flag
- Dan: what if guest doesn't respect injected MCE, takes down all other guests e.g. MCE on VMENTER (or other instructions that make it unrecoverable)
  - Jiaqi: let userspace VMM defend this case, VMM can request unmapping (as a policy). VMM can decide based on VM type.
    - Leave various options
- David: HWpoison is used by other subsystems to guard really bad errors. Should keep it set on the folio.