Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024

Post-Copy Live Migration with guest_memfd

James Houghton

jthoughton@google.com

LINUX PLUMBERS CONFERENCE Vienna, Austria / Sept. 18-20, 2024

Goals

- Review the scope of “KVM Userfault”
- Settle on the userspace API for KVM Userfault
- Discuss guest_memfd’s interaction with userfaultfd

Problem

- Need to support post-copy for memory-encrypted VMs
- Need to support post-copy for regular VMs that use guest_memfd
- Need post-copy to scale to 400+ vCPUs

Post-copy with memory-encrypted VMs today: procedure

- It can *technically* be done, with caveats

Procedure:

1. Do not set any memory to KVM_MEMORY_ATTRIBUTE_PRIVATE, register guest memory VMAs with userfaultfd
2. Guest-private faults will exit to userspace
   a. Page-in, set KVM_MEMORY_ATTRIBUTE_PRIVATE, resume vCPU
3. Guest-shared faults will go to userfaultfd, page-in normally

Post-copy with memory-encrypted VMs today: caveats

- Changing memory attributes is expensive
  - Setting attributes doesn't scale
    - We take mmu_lock and slots_lock
  - Requires dynamic memory allocations (xarray)
  - We need to have demand-fetch resolution in <50us
    - ~20,000 per vCPU per second, 100+ vCPUs
- For non-memory-encrypted VMs, all faults will go to userfaultfd initially
- Ideally leave memory attributes alone, so we need something else...

Intercepting accesses to guest_memfd memory

- guest_memfd will be mmap()-able, faults on non-private memory will succeed
- Non-guest accesses to non-private memory will use the userspace page tables
  - userfaultfd is usable for post-copy in this case
- Guest accesses to private *cannot* use GUP
  - **Need a method of intercepting these**
- Will guest accesses to non-private memory use GUP?
  - Most likely not, but the answer doesn't matter

KVM Userfault: concept

- KVM API to prevent **guest** accesses to any memory
- Does not intercept KVM's own accesses to guest memory
  - Only possible for guest-shared memory; we can just use userfaultfd
- KVM Userfault is not tied to guest_memfd
  - Instead: per-memslot bitmap

KVM Userfault: UAPI

- Enable with memslot flag KVM_MEM_USERFAULT
  - While enabled: faults will be at PAGE_SIZE only
- For each memslot, there is a bitmap describing if a guest-fault should exit
  - No reason to use memory attributes: “userfault” is transient
- Exit will be KVM_EXIT_MEMORY_FAULT
  - flags has KVM_MEMORY_EXIT_FLAG_USERFAULT
- On fault, userspace will write page contents and update the bitmap
- Collapse (or zap) page tables when KVM_MEM_USERFAULT is cleared

KVM Userfault: bitmap

- Stored entirely in userspace?
  - Each fault would need to copy_from_user() to check the bitmap
  - Userspace must be careful to order memory operations correctly when clearing bits in the bitmap
    - i.e. smp_wmb() between memory installation and bitmap update
    - But it probably already needs to be careful
  - Userspace will likely need a bitmap no matter what; saves a copy in KVM
  - Pass in using unused space in kvm_userspace_memory_region2?
  - I like this option
- Stored in KVM?
  - Need another syscall to update it
    - Bitmap needs an update for each demand-fetch, so an extra syscall per demand-fetch
    - When updating bitmap, KVM has the chance to collapse page tables

KVM Userfault and userfaultfd

- KVM Userfault and userfaultfd are completely different
- We still use userfaultfd for non-guest-private memory
  - KVM Userfault does not replace userfaultfd
- KVM Userfault replaces KVM_CAP_EXIT_ON_MISSING (never merged)
  - Enables post-copy scalability improvement when using userfaultfd

Userfaultfd-enlightenment of guest_memfd

- Guest-private faults will use KVM Userfault
- Non-private faults will go to vm_ops→fault
  - Needs to be userfaultfd-enlightened
- Userfaultfd enlightenment is anon-, shmem-, and hugetlbfs-specific
  - No other filesystems (including guest_memfd) participate
- Proposal: fs-generic minor fault mode
  - On fault: instead of calling vm_ops→fault, call handle_userfault().
  - On resolution: call vm_ops→fault and install a PTE/PMD
    - (Or could just call GUP with something like FOLL_NO_GENERIC_UFFD).

Questions