Removing Guest Memory from the Host Kernel's Direct Map

Patrick Roy
roypat@amazon.co.uk

Amazon Web Services

Linux Plumbers Conference, September 2024

Goal

Goal

Defense in-depth measures to protect guest state from large class of transient execution issues (Spectre, ...)

- Guest Memory[1]
- vCPU State[2]

[1] https://lore.kernel.org/kvm/20240910163038.1298452-1-roypat@amazon.co.uk/

[2] https://lore.kernel.org/all/20240911143421.85612-1-faresx@amazon.de/

Goal

Goal

Defense in-depth measures to protect guest state from large class of transient execution issues (Spectre, ...)

- Guest Memory[1] → this session!
- vCPU State[2]

[1] https://lore.kernel.org/kvm/20240910163038.1298452-1-roypat@amazon.co.uk/

[2] https://lore.kernel.org/all/20240911143421.85612-1-faresx@amazon.de/

Goal

Goal

Defense in-depth measures to protect guest state from large class of transient execution issues (Spectre, ...)

- Guest Memory[1] → this session!
- vCPU State[2]

Constraint

Do not want to change VM model

→ KVM/host userspace should be able to access guest memory as for traditional VMs

[1] https://lore.kernel.org/kvm/20240910163038.1298452-1-roypat@amazon.co.uk/

[2] https://lore.kernel.org/all/20240911143421.85612-1-faresx@amazon.de/

Sharing

KVM access to guest memory

- kvm-clock
- Guest page table walks
- MMIO instruction fetch (x86)
- nested page table walks (?)
- more?

Sharing

KVM access to guest memory

- kvm-clock
- Guest page table walks
- MMIO instruction fetch (x86)
- nested page table walks (?)
- more?

Short Term

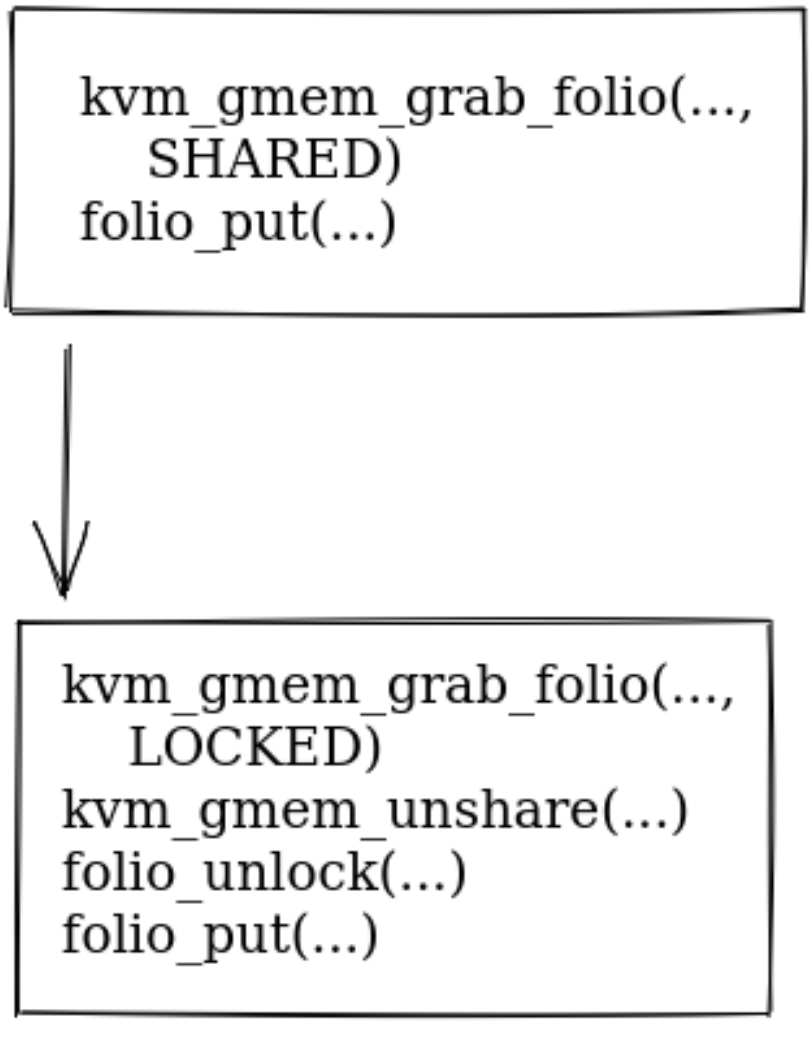
```
kvm_gmem_grab_folio(...,
    LOCKED | SHARED)
/* access memory */
kvm_gmem_unshare(...)
folio_unlock(...)
folio_put(...)
```

Sharing

KVM access to guest memory

- kvm-clock
- Guest page table walks
- MMIO instruction fetch (x86)
- nested page table walks (?)
- more?

"Long" Term

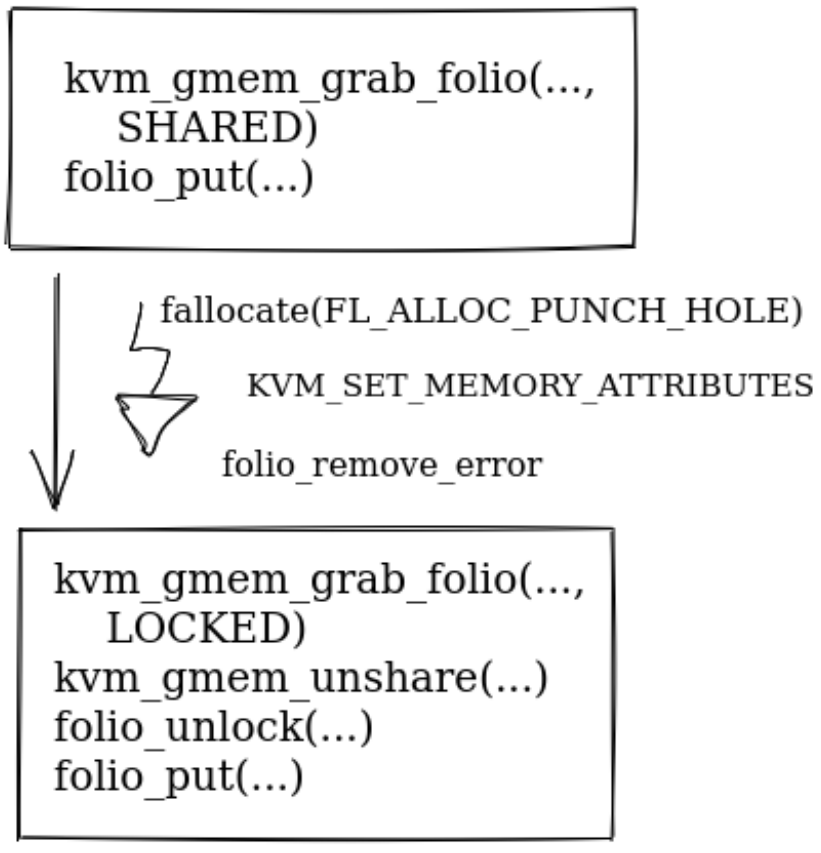


Sharing

KVM access to guest memory

- kvm-clock
- Guest page table walks
- MMIO instruction fetch (x86)
- nested page table walks (?)
- more?

"Long" Term



How to deal with "long term sharing"

"pfncache approach"?

Respond to KVM invalidations/MMU notifiers (RFC v2)

→ can't do direct map modification in notifier, nasty races

Destructive ahead-of-time Guest Sharing?

→ Guest might not always know what to share (nested page table walks? guest swap/migration?)

Disallow Them

Treat each access as separate short-term share (translate gpa, manipulate direct map, etc.)

→ Performance, no more pfncaches, ...

Backup Slides