author	Sean Christopherson <seanjc@google.com>	2025-04-04 12:38:17 -0700
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2025-05-02 07:44:31 +0200
commit	116c7d35b8f72eac383b9fd371d7c1a8ffc2968b (patch)	
tree	5f69ef41e1d9a4f93cca095c99949bd2168aa1d5	
parent	26ccc791de508e609cd238e1f4bc234d24af1bfb (diff)	
download	linux-116c7d35b8f72eac383b9fd371d7c1a8ffc2968b.tar.gz	

diff options

context: 3

space: include

mode: unified

KVM: x86: Reset IRTE to host control if *new* route isn't postable

```
commit 9bcac97dc42d2f4da8229d18feb0fe2b1ce523a2 upstream.

Restore an IRTE back to host control (remapped or posted MSI mode) if the
*new* GSI route prevents posting the IRQ directly to a vCPU, regardless of
the GSI routing type.  Updating the IRTE if and only if the new GSI is an
MSI results in KVM leaving an IRTE posting to a vCPU.

The dangling IRTE can result in interrupts being incorrectly delivered to
the guest, and in the worst case scenario can result in use-after-free,
e.g. if the VM is torn down, but the underlying host IRQ isn't freed.

Fixes: efc644048ecd ("KVM: x86: Update IRTE for posted-interrupts")
Fixes: 411b44ba80ab ("svm: Implements update_pi_irte hook to setup posted interrupt")
Cc: stable@vger.kernel.org
Signed-off-by: Sean Christopherson <seanjc@google.com>
Message-ID: <20250404193923.1413163-3-seanjc@google.com>
Signed-off-by: Paolo Bonzini <pbonzini@redhat.com>
Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

Diffstat

-rw-r--r--	arch/x86/kvm/svm/avic.c	58
-rw-r--r--	arch/x86/kvm/vmx/posted_intr.c	28

2 files changed, 41 insertions, 45 deletions

```
diff --git a/arch/x86/kvm/svm/avic.c b/arch/x86/kvm/svm/avic.c
index 2274981487d891..4bb2fbe6676a1f 100644
--- a/arch/x86/kvm/svm/avic.c
+++ b/arch/x86/kvm/svm/avic.c
@@ -801,6 +801,7 @@ int svm_update_pi_irte(struct kvm *kvm, unsigned int host_irq,
 {
        struct kvm_kernel_irq_routing_entry *e;
        struct kvm_irq_routing_table *irq_rt;
+       bool enable_remapped_mode = true;
        int idx, ret = 0;

        if (!kvm_arch_has_assigned_device(kvm) ||
@@ -838,6 +839,8 @@ int svm_update_pi_irte(struct kvm *kvm, unsigned int host_irq,
                    kvm_vcpu_apicv_active(&svm->vcpu)) {
                        struct amd_iommu_pi_data pi;

+                       enable_remapped_mode = false;
+
                        /* Try to enable guest_mode in IRTE */
```

```
                        pi.base = __sme_set(page_to_phys(svm->avic_backing_page) &
                                            AVIC_HPA_MASK);
@@ -856,33 +859,6 @@ int svm_update_pi_irte(struct kvm *kvm, unsigned int host_irq,
                         */
                        if (!ret && pi.is_guest_mode)
                                svm_ir_list_add(svm, &pi);
-               } else {
-                       /* Use legacy mode in IRTE */
-                       struct amd_iommu_pi_data pi;
-
-                       /**
-                        * Here, pi is used to:
-                        * - Tell IOMMU to use legacy mode for this interrupt.
-                        * - Retrieve ga_tag of prior interrupt remapping data.
-                        */
-                       pi.prev_ga_tag = 0;
-                       pi.is_guest_mode = false;
-                       ret = irq_set_vcpu_affinity(host_irq, &pi);
-
-                       /**
-                        * Check if the posted interrupt was previously
-                        * setup with the guest_mode by checking if the ga_tag
-                        * was cached. If so, we need to clean up the per-vcpu
-                        * ir_list.
-                        */
-                       if (!ret && pi.prev_ga_tag) {
-                               int id = AVIC_GATAG_TO_VCPUID(pi.prev_ga_tag);
-                               struct kvm_vcpu *vcpu;
-
-                               vcpu = kvm_get_vcpu_by_id(kvm, id);
-                               if (vcpu)
-                                       svm_ir_list_del(to_svm(vcpu), &pi);
-                       }
                }

                if (!ret && svm) {
@@ -898,6 +874,34 @@ int svm_update_pi_irte(struct kvm *kvm, unsigned int host_irq,
        }

        ret = 0;
+       if (enable_remapped_mode) {
+               /* Use legacy mode in IRTE */
+               struct amd_iommu_pi_data pi;
+
+               /**
+                * Here, pi is used to:
+                * - Tell IOMMU to use legacy mode for this interrupt.
+                * - Retrieve ga_tag of prior interrupt remapping data.
+                */
+               pi.prev_ga_tag = 0;
+               pi.is_guest_mode = false;
+               ret = irq_set_vcpu_affinity(host_irq, &pi);
+
+               /**
+                * Check if the posted interrupt was previously
+                * setup with the guest_mode by checking if the ga_tag
+                * was cached. If so, we need to clean up the per-vcpu
+                * ir_list.
+                */
+               if (!ret && pi.prev_ga_tag) {
+                       int id = AVIC_GATAG_TO_VCPUID(pi.prev_ga_tag);
+                       struct kvm_vcpu *vcpu;
+
+                       vcpu = kvm_get_vcpu_by_id(kvm, id);
```

```
+                       if (vcpu)
+                               svm_ir_list_del(to_svm(vcpu), &pi);
+               }
+       }
 out:
        srcu_read_unlock(&kvm->irq_srcu, idx);
        return ret;
```

```
diff --git a/arch/x86/kvm/vmx/posted_intr.c b/arch/x86/kvm/vmx/posted_intr.c
index 46fb83d6a286e7..4ca480ced35dc2 100644
--- a/arch/x86/kvm/vmx/posted_intr.c
+++ b/arch/x86/kvm/vmx/posted_intr.c
@@ -270,6 +270,7 @@ int pi_update_irte(struct kvm *kvm, unsigned int host_irq, uint32_t guest_irq,
 {
        struct kvm_kernel_irq_routing_entry *e;
        struct kvm_irq_routing_table *irq_rt;
+       bool enable_remapped_mode = true;
        struct kvm_lapic_irq irq;
        struct kvm_vcpu *vcpu;
        struct vcpu_data vcpu_info;
@@ -308,21 +309,8 @@ int pi_update_irte(struct kvm *kvm, unsigned int host_irq, uint32_t guest_irq,

                kvm_set_msi_irq(kvm, e, &irq);
                if (!kvm_intr_is_single_vcpu(kvm, &irq, &vcpu) ||
-                   !kvm_irq_is_postable(&irq)) {
-                       /*
-                        * Make sure the IRTE is in remapped mode if
-                        * we don't handle it in posted mode.
-                        */
-                       ret = irq_set_vcpu_affinity(host_irq, NULL);
-                       if (ret < 0) {
-                               printk(KERN_INFO
-                                  "failed to back to remapped mode, irq: %u\n",
-                                  host_irq);
-                               goto out;
-                       }
-
+                   !kvm_irq_is_postable(&irq))
                        continue;
-               }

                vcpu_info.pi_desc_addr = __pa(&to_vmx(vcpu)->pi_desc);
                vcpu_info.vector = irq.vector;
@@ -330,11 +318,12 @@ int pi_update_irte(struct kvm *kvm, unsigned int host_irq, uint32_t guest_irq,
                trace_kvm_pi_irte_update(host_irq, vcpu->vcpu_id, e->gsi,
                                vcpu_info.vector, vcpu_info.pi_desc_addr, set);

-               if (set)
-                       ret = irq_set_vcpu_affinity(host_irq, &vcpu_info);
-               else
-                       ret = irq_set_vcpu_affinity(host_irq, NULL);
+               if (!set)
+                       continue;
+
+               enable_remapped_mode = false;

+               ret = irq_set_vcpu_affinity(host_irq, &vcpu_info);
                if (ret < 0) {
                        printk(KERN_INFO "%s: failed to update PI IRTE\n",
                                        __func__);
@@ -342,6 +331,9 @@ int pi_update_irte(struct kvm *kvm, unsigned int host_irq, uint32_t guest_irq,
                }
        }

```

```
+       if (enable_remapped_mode)
+               ret = irq_set_vcpu_affinity(host_irq, NULL);
+
        ret = 0;
 out:
        srcu_read_unlock(&kvm->irq_srcu, idx);
```