RMRR EXCLUSION

Technical Whitepaper

Alex Williamson alex.williamson@redhat.com
Myron Stowe myron.stowe@redhat.com
Laura Novich lnovich@redhat.com

Version 1.0

March 2015

100 East Davie Street
Raleigh NC 27601USA
Phone: +1 919 754 4950
Phone: 888 733 4281
Fax: +1 919 800-3804

TABLE OF CONTENTS

1. Executive Summary

Red Hat Enterprise Linux 7.1 blocks the use of certain devices from PCI device assignment configurations. This whitepaper explains the rationale for the changes that were made, gives you information so you know if your systems have been affected, and points out questions or requests that you should ask platform vendors in order to prevent these issues.

2. Introduction

PCI device assignment is a hypervisor feature which allows a physical PCI device to be directly assigned to a guest virtual machine. This gives the guest direct access to the physical hardware registers and programming of these devices and mostly eliminates the hypervisor in the I/O path through these devices for the guest. The benefits for PCI device assignment include reducing of I/O latency and increasing I/O bandwidth, while also allowing virtual machines to connect to devices which have no equivalent emulation model, such as fiber channel or infiniband HBAs (Host Bus Adaptors).

In order to enable PCI device assignment and allow the guest operating system to directly program the device, a mechanism is needed to provide both isolation and translation of DMA to and from the device. Isolation is required to ensure that the device cannot access either the host memory or the memory of any other virtual machines running on the host. Translation is required to allow the guest operating system to program the device for DMA (Direct Memory Access) using guest physical memory addresses. This avoids the need for any additional DMA drivers in the guest and as such, the guest is able to make use of the device just as it would on a bare metal system with no special drivers required beyond what is needed for the assigned device itself. This type of device assignment is referred to as transparent PCI device assignment and it is enabled by the I/O Memory Management Unit or the IOMMU. Intel® based systems provide an IOMMU with these capabilities on platforms supporting Intel® Virtualization Technology for Directed I/O, or VT-d. AMD® implements similar functionality on systems supporting AMD-Vi.

3. The Address Space Obstacle

When a PCI device is assigned to a virtual machine, the hypervisor must pin all of the guest's memory and map it through the IOMMU. The pinning process locks the guest's memory pages and binds their translation to physical memory, where a page is the unit of memory allocation for the system. These pinned pages are then mapped through the IOMMU, creating a translation between every physical memory address in the virtual machine to the host physical memory backing those addresses. This allows for transparent device assignment; the guest can program the assigned device to perform DMA on any address within the guest physical memory and the device will transparently access the corresponding host physical memory. This is fundamental to how PCI device assignment operates today.

The Intel® VT-d specification[1] includes a feature known as the Reserved Memory Region Reporting structure, or RMRR. RMRR definitions associate a physical memory region with one or more devices, so that when the IOMMU is initialized the device will continue to have access to the physical memory at the original address. In other words, the IOMMU must provide a direct one-to-one translation, or an identity mapping, for every referenced device in order to satisfy the RMRR.

RMRRs are designed to allow specific memory regions to be used by a device across the host OS initialization of the IOMMU hardware. These mappings are used for legacy emulation and/or basic usage of the device. For instance, USB controllers may provide PS/2 keyboard emulation for pre-boot BIOS and early OS boot, until a native USB driver in the host takes over the device. Similarly, an onboard graphics controller may require access to various structures in memory to continue to operate in VGA mode (or through UEFI runtime services) until the native host driver is loaded and the memory structures can be mapped normally through the IOMMU. In these specific known cases, the operating system assumes that the memory association terminates when device specific drivers are initialized. However, the VT-d specification makes no such exemption, requiring that the RMRR defined memory associations are retained for the duration of the host boot. Therefore only these few known cases are excluded, as all unknown cases must honor the RMRR requirement indefinitely.

1 Intel® Virtualization Technology for Directed I/O Architecture Specification, September 2013, Rev. 2.2

The primary intention of these regions is simply to allow devices to continue to operate across IOMMU initialization until native drivers are available. The VT-d specification also includes an advisory statement to this effect:

> "The RMRR regions are expected to be used for legacy usages (such as USB, UMA Graphics, etc.) requiring reserved memory. Platform designers should avoid or limit use of reserved memory regions since these require system software to create holes in the DMA virtual address range available to system software and its drivers."[2]

Unfortunately, some platform vendors have not adhered to this advisory and continue to make extensive use of RMRRs within their systems. This is often done for the purpose of allowing a device to continue to communicate with platform management hardware through a shared physical memory address, while an operating system is running.

From a PCI device assignment perspective, RMRRs introduce a number of issues. The first, is that a device with a corresponding RMRR region has a requirement that the device must continue to have identity mapped access to the specified physical memory. This interferes with our VM usage of the device which requires that the VM address space is programmed through the IOMMU to allow transparent access for DMA by the guest. When an RMRR region for a device is present, the hypervisor no longer has full control of the IOMMU address space presented to the guest. If the identity mapping for the RMRR is retained, the address space consumed by it cannot be used within the guest, introducing reserved memory requirements in the guest address space. If the RMRR is terminated and the address space specified by it remapped to the guest physical memory, the device may continue to DMA to the RMRR defined memory. This not only fails to honor the RMRR, but potentially results in integrity issues within the guest.

Second, the RMRR requirements create an isolation issue for the associated devices. In the scenario where an RMRR creates a shared physical memory region between a device and a platform controller, new security vulnerabilities are potentially created where the guest with an assigned device can attempt to exploit the host platform via the RMRR memory area. For instance, if the memory area is used to report temperature data from the device for the purposes of thermal regulation, reporting incorrect data may compromise the platform's stability.

2ibid, Section 8.4

Furthermore, as the hypervisor needs to account for the sharing of RMRR memory regions between devices in evaluating device isolation, a shared physical memory region between two devices may indicate that they are not isolated from one another. If one of those devices share another memory region with a third device, it can be considered that the set of all three devices are not fully isolated from one another. For scenarios where each device shares an RMRR memory region with a platform management controller, this quickly escalates to a large set of non-isolated devices, making assignment of any individual device within the set impractical.

For all these reasons, updates to RHEL 7.1 automatically exclude devices associated with RMRRs from the kernel interfaces required for PCI device assignment.

4. Determining The Extent Of Impact

For the majority of platform vendors, this change is not expected to interfere with current operation nor is it expected to significantly reduce the set of devices available for PCI device assignment. Most vendors adhere to the VT-d specification advisory, only including RMRRs for USB devices and UMA graphics devices. Furthermore, since the usage of RMRRs by USB devices is understood, an exemption is provided in the kernel to allow assignment of these devices to continue. UMA graphics devices are not currently supported for PCI device assignment, resulting in effectively no change for these devices. Virtual functions adhering to the PCIe SR-IOV specifications typically are not associated with RMRRs and are therefore likely to be unaffected by this change.

Under the changes made to Red Hat Enterprise Linux 7.1, when an attempt is made to assign a device that is entangled by an RMRR association, the kernel reports the following error in dmesg:

```
vfio-pci 0000:01:00.1: Device is ineligible for IOMMU domain attach due to platform
RMRR requirement. Contact your platform vendor.
```

libvirt managed guests including such devices will fail to start with a permission denied error in the domain error log (as well as the above dmesg error):

```
error: Failed to start domain vm0
error: internal error: process exited while connecting to monitor:
2014-11-05T21:19:49.631332Z qemu-kvm: -device vfio-
pci,host=01:00.1,id=hostdev0,bus=pci.0,addr=0x5: vfio: failed to set iommu for
container: Operation not permitted
2014-11-05T21:19:49.631390Z qemu-kvm: -device vfio-
pci,host=01:00.1,id=hostdev0,bus=pci.0,addr=0x5: vfio: failed to setup container for
group 15
2014-11-05T21:19:49.631414Z qemu-kvm: -device vfio-
pci,host=01:00.1,id=hostdev0,bus=pci.0,addr=0x5: vfio: failed to get group 15
2014-11-05T21:19:49.631440Z qemu-kvm: -device vfio-
pci,host=01:00.1,id=hostdev0,bus=pci.0,addr=0x5: Device initialization failed.
2014-11-05T21:19:49.631470Z qemu-kvm: -device vfio-
pci,host=01:00.1,id=hostdev0,bus=pci.0,addr=0x5: Device 'vfio-pci' could not be
initialized
```

The following Information, found in dmesg, can be used to determine whether a given system includes devices associated with RMRRs:

```
IOMMU: Setting RMRR:
IOMMU: Setting identity map for device 0000:02:00.0 [0xbdf7f000 - 0xbdf8efff]
IOMMU: Setting identity map for device 0000:01:00.0 [0xbdf7f000 - 0xbdf8efff]
IOMMU: Setting identity map for device 0000:01:00.2 [0xbdf7f000 - 0xbdf8efff]
IOMMU: Setting identity map for device 0000:02:00.0 [0xbdf8f000 - 0xbdf92fff]
IOMMU: Setting identity map for device 0000:01:00.0 [0xbdf8f000 - 0xbdf92fff]
IOMMU: Setting identity map for device 0000:01:00.2 [0xbdf8f000 - 0xbdf92fff]
IOMMU: Setting identity map for device 0000:02:00.0 [0xbdf93000 - 0xbdf94fff]
IOMMU: Setting identity map for device 0000:01:00.0 [0xbdf93000 - 0xbdf94fff]
IOMMU: Setting identity map for device 0000:01:00.2 [0xbdf93000 - 0xbdf94fff]
IOMMU: Setting identity map for device 0000:01:00.0 [0xbdff6000 - 0xbdffcfff]
IOMMU: Setting identity map for device 0000:01:00.2 [0xbdff6000 - 0xbdffcfff]
IOMMU: Setting identity map for device 0000:01:00.4 [0xbdff6000 - 0xbdffcfff]
```

In this example, PCI devices 0000:01:00.0, 0000:01:00.2, 0000:01:00.4, and 0000:02:00.0 have RMRR associations, with the specific RMRR memory region per device listed in brackets. Due to the changes in RHEL 7.1 mentioned earlier, these devices are not available for PCI device assignment to a guest.

In the past, even if RMRRs were present for a device and the device has been previously used for PCI device assignment, the user would likely not have experienced any issues. The reset mechanisms used by the hypervisor on the device may have caused the device to discontinue usage of the RMRR area or the RMRR address may have fallen into an unused region of memory in the guest address space. However, neither of these behaviors can be guaranteed due to the nature of these devices and the layout of RMRR memory within the host. The prudent approach is to prevent devices with associated RMRRs from being used in PCI device assignment configurations. The risk of VM integrity issues and platform stability are too severe and there are no guidelines by which a user can make an informed decision regarding the ongoing use of devices with RMRR requirements.

Furthermore, it should be noted that in Red Hat Enterprise Linux 7, other uses for the VFIO userspace driver interface beyond QEMU/KVM are also affected by this change, including, but not limited to, the VFIO version of the Data Plane Development Kit (DPDK) driver. The VFIO-DPDK driver has a similar issue to a virtual machine in terms of RMRRs; the userspace DPDK driver defines the address space used by the device and is also unable to account for RMRRs in the IOMMU address space.

5. Conclusion

Users encountering restricted device usage due to this change, are encouraged to contact their hardware platform vendors for system BIOS updates or configuration guidance.

6. Disclaimer

AMD-Vi based platforms have a similar feature to RMRRs known as I/O Virtualization Memory Definition, or IVMD. The changes described in this document currently apply only to Intel® VT-d based systems. While AMD-Vi systems may present similar issues, Red Hat is not currently aware of any platform vendors making use of IVMDs for devices commonly assigned to virtual machines.

7. References

http://www.intel.com/content/www/us/en/intelligent-systems/intel-technology/vt-directed-io-spec.html