文章编号:1005-1228 (2010) 05-0001-05

基于 SR- IOV 的 IO 虚拟化技术

李 超,董 青,戴华东

(国防科学技术大学计算机学院,湖南 长沙 410073)

摘 要:虚拟技术经过多年的发展,CPU 虚拟化与内存虚拟化均已成熟,而 I/O 虚拟化方面却未出现大的变化,成为当前虚拟技术性能提高的瓶颈。近期 Intel 公司提出的 SR-IOV 技术通过在硬件层增加虚拟支持,与原有 I/O 虚拟化中采用的 Passthrough 技术相结合,极大的提高了物理设备的使用效率和客户域的 I/O 性能。文章在总结虚拟技术中采用过的 I/O 模型基础上,分析了 SR-IOV 技术的实现和特点。

关键词:SR-IOV ;虚拟技术; I/O 模型

中图分类号:TP316 文献标识码:A

IO Virtualization Technology Based on SR-IOV

LI Chao, DONG Qing, DAI Hua-dong

(Compute College, National University of Defence Technology, Changsha, 410073, China)

Abstract: Through years of development in virtualization, CPU and memory virtualization have seen a great progress and improvement, while I/O virtualization has not shown much breakthrough, which posts a bottleneck to current virtualization performance. Recently, Intel Corp. brought up SR-IOV, a technology cooperated with Passthrough adopted in former I/O virtualization and strengthened by hardware improvement, which has greatly improved I/O efficiency of physical equipments and client domain. This paper summarizes I/O model used in virtualization technology and analyzes the implementation and characteristics of SR-IOV.

Key words: SR-IOV; Virtual machine; I/O model

虚拟化技术发展的初期主要致力于 CPU 与内存的虚拟化,注重于 CPU 及内存性能的提高。随着硬件技术的进步,虚拟化技术在部件、系统及应用级都取得全面发展,Intel VT 技术和 AMD VMX 技术的引入,虚拟技术在 CPU 虚拟化方面和内存虚拟化方面都取得较大的进展,但在在外围设备虚拟化方面,由于实现模型未发生太大变化,所以 I/O 设备的利用率未有显著提高,相对而言,其性能还远远滞后于无虚拟机环境下的 I/O 性能。

针对虚拟 I/O 设备问题,Xen[1-2]和 VMWare 分别采用 Split I/O[3]模型,Direct I/O[4]模型来实现,上述实现模型均在软件层实现,对设备的利用效率和对请求的访问速度均不如意。最近 Intel 提出了支持设备虚拟化技术的 SR- IOV[5]技术,通过在硬件层上支持 I/O 虚拟化显著的提高了虚拟域的 I/O 性能。本文通过分析 SR- IOV(Singe Root I/O Virtualization)技术的产生背景和技术规范,剖析其实现原理和实现模型。

本文首先分析了当前基于软件和硬件的几种高效虚拟化 I/O 访问模型,然后对 Intel 最新基于虚拟化 IO 的研究成果 SR- IOV 技术进行了介绍,并对其原理进行了分析,最后做出了评价。

1 相关技术研究

论文对当前虚拟技术中采用的高效 I/O 设备访问模型进行了研究,通过对比分析基于软件的 I/O 设备模拟技术 (包括 Split I/O、Direct I/O Passthrough I/O) ,以及硬件辅助虚拟化技术[6] (VT- x、VT- d 等) ,总结了虚拟化 I/O 技术的发展趋势。

1.1 基于软件的 I/O 模型

基于软件的虚拟化 I/O 模型是将 I/O 硬件的逻辑部分移入到虚拟机中,模拟层位于客户机与底层硬件之间。根据模拟层的具体位置,软件模拟 I/O 技术大致分为特权的宿主机模拟和虚拟机管理器 (以下简称 VMM) 模拟。宿主机模拟通常采用 Split I/O 技术,又称为前端 / 后端模拟;VMM 模拟一般采用 Direct I/O 技术[7]。

收稿日期:2010- 04- 06

基金项目:国家 863 项目:支持多计算域的虚拟个人计算机系统 (2007AA01Z177)

作者简介:李 超 (1980-) ,男,湖北武汉市人,硕士,主研领域:虚拟化技术; 戴华东,男,湖北随州人,硕士生导师,主研领域:操作系统。

Split I/O 技术是指将传统的 I/O 驱动模型分成两部分:一部分在无 I/O 访问特权的虚拟机(简称 Dom N)中,称为前端驱动,不直接对 I/O 设备进行访问;另一部分则在有 I/O 访问特权的虚拟机(Dom0)中,称为后端驱动,可以调用物理 I/O 设备驱动,访问硬件。前端驱动接收 Dom N 上层应用的 I/O 请求,通过事件通道机制(多个虚拟机之间相互通讯的机制)传递给后端驱动。后端驱动处理前端驱动发送的请求,根据请求调用相应设备驱动程序对 I/O 设备进行访问,如图 1 所示:

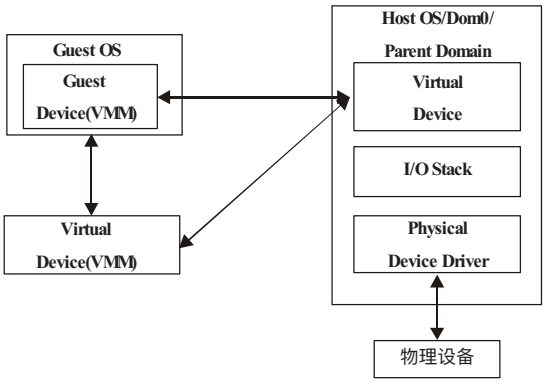

图 1 软件模拟技术:Split I/O 模型

Split I/O 通常可以采用高效的通信机制,这在很大程度上减少客户机之间上下文切换开销。但 Split I/O 也存在以下一些不足:

⑴ Dom N 进行 I/O 操作都需要通过 VMM 转发请求,这会带来虚拟机与 VMM 之间的场景切换开销。因此,该方法会使整个 I/O 架构中的 Dom0 成为瓶颈;

⑵ 在有大量的 I/O 请求时,更会带来巨大的客户机之间场景切换的开销,消耗大量 CPU 资源,影响虚拟 I/O 系统的效率;

⑶ 为了实现 Split I/O,需要修改客户端操作系统,以达到 VMM 和客户机之间的协同工作,因而无法支持非开源的操作系统。

Direct I/O 模型主要包括客户端驱动程序(相对于传统驱动不需修改)、设备虚拟层、I/O 数据传输的虚拟栈、VMM 中直接和底层设备交互的驱动程序以及物理设备。如图 2 所示,其中,设备虚拟层虚拟出各种 I/O 设备;I/O 虚拟栈将客户机的 I/O 地址映射到 VMM 的地址空间,处理 VMM 内部的通信,支持客户机与物理设备之间的 I/O 多路转发。当客户机发起 I/O 请求时,直接自陷到 VMM 中,以达到对物理设备的直接访问操作。但是由于 I/O 设备需要在多个客户机之间共享,因此需要通过 VMM 的介入以保证各个虚拟机对设备访问的合法性和一致性,这就导致了虚拟机的每次 I/O 操作都需要 VMM 的介入。

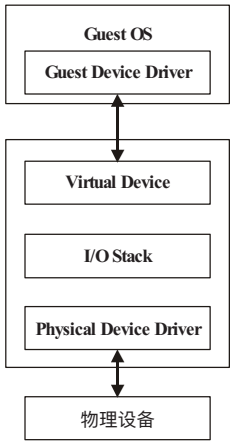

图 2 软件模拟技术:Direct I/O 模型

与 Split I/O 相比,Direct I/O 的实现方式更加有效和易于升级;兼容性方面 Direct I/O 模型更易于设备驱动程序的复用;故障隔离方面,Split I/O 通过将驱动部署在一个特定的客户机中来达到隔离性,而 Direct I/O 更容易发挥沙箱技术和其他技术的优势。但 Direct I/O 存在以下一些不足:

⑴ 由于每次 I/O 访问都需要 VMM 的介入,对于 I/O 密集型访问或者多虚拟机同时进行 I/O 操作时,VMM 将迅速成为瓶颈,导致 I/O 延迟的增大,I/O 效率的大幅度降低;

⑵ 驱动程序部署于 VMM 内部增加了 VMM 设计的复杂程度,难以移植设备驱动;

⑶ 完成一次 I/O 操作需要涉及多个寄存器的操作,这要求 VMM 截获每个寄存器访问并进行相应的模拟,导致多次上下文切换,使得性能下降。

Passthrough I/O 模型是指在客户机内部能够直接对硬件进行操作,如图 3 所示[8]。客户机与硬件的交互只需要经过少量、或者不需要经过 VMM 的管理。Passthrough I/O 模型将设备独占式地分配给指定的客户域,使该域具有最高的 I/O 访问性能。这样做的优点是:由于不需要模拟设备进行请求转换,所以访问速度高;客户机能根据最新硬件,加载对应驱动,可充分发挥新硬件的功能。由于客户机可以内部直接的操纵硬件设备,这大大的提高了 I/O 性能。

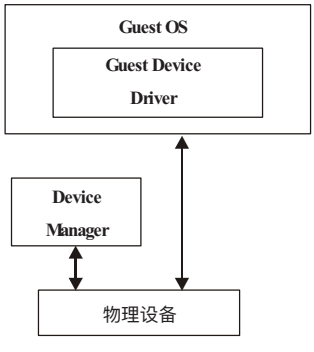

图 3 Passthrough I/O 模型

与基于软件的虚拟化 I/O 模型相比,Passthrough 技术可以直接对 I/O 设备进行操作，大大降低了 CPU 的开销;Passthrough 的 I/O 操作不需要经过一个有特权的客户机，不存在 I/O 瓶颈问题;Passthrough I/O 模型不需要修改客户机驱动;Passthrough 具有相对比较薄的 VMM,VMM 的设计相对简单，可移植性相对较好。但 Passthrough I/O 也存在不足:

⑴ Passthrough 最大程度上提高的 I/O 性能是以牺牲设备共享能力为代价的;

⑵ Passthrough I/O 中的设备只能被某个客户机所独占,难以充分发挥 I/O 设备的利用率。

1.2　硬件辅助的模型

由于目前基于软件的虚拟化 I/O 模型增加了 CPU 的负担,希望借助硬件辅助技术完成一部分虚拟化 I/O 的功能 。

Intel 和 AMD 都在处理器架构中提供对 Passthrough I/O 的支持。Intel 将这种支持称为 VT- d (Virtualization Technology for Directed I/O) ,AMD 称之为 IOMMU (I/O Memory Managemnt Unit) 。这种技术的 CPU 能够将 PCI 物理地址映射到客户机中。当这种映射发生时,硬件将负责访问和保护,客户机像宿主系统一样可以直接使用该设备。除了将客户机映射到物理内存外，还提供隔离机制，以便预先阻止其他客户机(或管理程序) 访问该区域[9]。

传统的 IOMMU 提供了一种集中的方式管理所有的DMA。除了传统的内部 DMA,还包括 AGP、GART- ner、TPT、TCP/IP 等这些特别的 DMA,它通过内存地址范围来区别设备，却不容易实现 DMA 隔离，因此 VT- d 通过更新设计的 IOMMU 架构,实现了多个 DMA 保护区域的存在，最终实现了 DMA 虚拟化，也叫做 DMA 重映射 (DMA Remapping) 。

I/O 设备会产生非常多的中断请求,虚拟化技术必须正确地隔离这些请求,并路由到不同的虚拟机上。传统设备的中断请求可以具有两种方式：一种将 I/O 中断控制器路由，一种是通过 DMA 写请求直接发送出去的 MSI (Message Singled Interupt)，由于需要在 DMA 请求内嵌入目标内存地址，因此需要访问所有的内存地址,并不能实现中断隔离。VT- d 实现的中断重映射架构通过重新定义 MSI 的格式来解决这个问题,新的 MSI 仍然是一个 DMA 写请求的形式,不过并不嵌入目标内存地址,取而代之的是一个消息 ID,通过维护一个表结构,硬件可以通过不同的消息 ID 辨认不同的虚拟机区域。VT- d 实现的中断重映射可以支持所有的 I/O 源和中断类型,MSI 以及 MSI- X。

VT- d 技术可以隔离和保护硬件资源只给指定的虚拟机使用，硬件同时还需要具备多个 I/O 分区来同时为多个虚拟机服务,更好地支持 Passthrough 技术。

2　SR- IOV 介绍与原理

传统的基于软件和硬件辅助的虚拟化 I/O 方法虽然能够从不同角度提高虚拟化 I/O 的能力，但无法同时获得 I/O 设备的高性能和共享性。SR- IOV 技术规范正是针对这一问题提出了相应的解决方法。

2.1　SR- IOV 介绍

从软件的角度来看,I/O 设备通过三种方式与 CPU 进行通信,分别是:中断、寄存器读写和内存共享。软件通过寄存器读写对设备进行操作，而 I/O 设备通过中断的方式通知 CPU 处理的情况。内存共享则通过 DMA 使 I/O 设备与 CPU 之间进行大规模数据通信[10]。

SR- IOV 是 PCI- SIG 组织公布的一个新规范,旨在消除 VMM 对虚拟化 I/O 操作的干预,以提高数据传输的性能。SR- IOV 继承了 Passthrough I/O 技术,通过 IOMMU 减少存储保护和地址转换的开销。

具有 SR- IOV 功能的 I/O 设备是基于 PCIe 规范的，可以用来管理并创建多个 VF (virtual function) 。PCIe PF((Physical function)在 PCIe 总线上是主要实体,具有唯一的申请标示 RID,一个 PCIe 设备具有一个或多个 PF。SR- IOV 设备可以有一个或多个 PF，如图 4 所示:

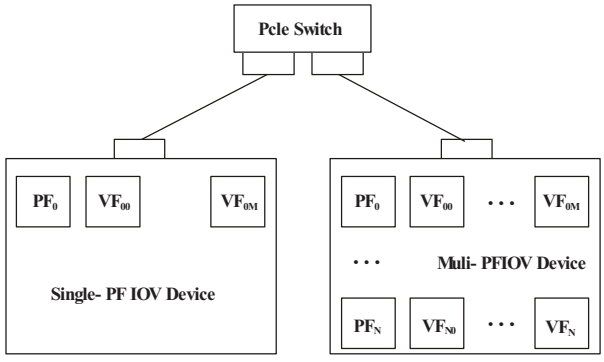

图 4　有 SR- IOV 能力的 I/O 设备

每个 PF 都是标准的 PCIe 功能，并且关联多个 VF。每个 VF 都有与性能相关的资源,专门用于软件实体在运行时的性能数据运转，同时这些 VF 共享物理设备资源,如图 5 所示。因此,VF 可以视为由 PF 进行配置和管理的“轻量级”PCIe 功能[11]。

每个 VF 对应唯一的 RID,RID 则确定了唯一的 PCIe 交换源。RID 也能够用于索引 IOMMU 页表,因此不同的 VM 可以使用不同的页表。IOMMU 页表是在

DMA 交换中用于存储保护和地址转换。设备初始化和配置资源没有应用在 VF 上，因此与传统的多功能 PCIe 设备相比，在有限的芯片设计预算里 SR- IOV 设备可以包含更多的 VF, 具有更好的可扩展性[12]。

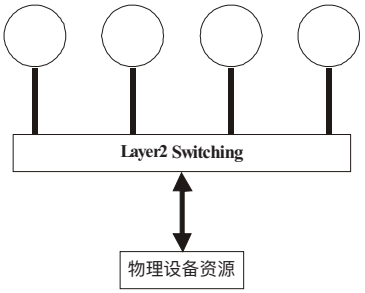

图 5　SR- IOV 设备的资源共享

SR- IOV 提出了地址传输服务 ATS，用以提高性能。ATS 需要每个 I/O 设备都使用地址转换机制, 通过本地 I/O TLB (I/O 转换旁路缓冲) 作为地址转换缓冲。这使得设备在传输之前就能够转换 DMA 地址，从而避免了 I/O TLB 在 IOMMU 地址转换过程中失效。

2.2　SR- IOV 的实现模型

SR- IOV 的实现模型包括 VF 驱动、PF 驱动、IOVM (SR- IOV 管理器) 。VF 驱动是运行在客户机上的普通设备驱动; PF 驱动则部署在宿主机上对 VF 进行管理; 在宿主机上的 IOVM 用于管理 PCIe 拓扑的控制点以及表示每个 VF 的配置空间；整体结构如图 6 所示:

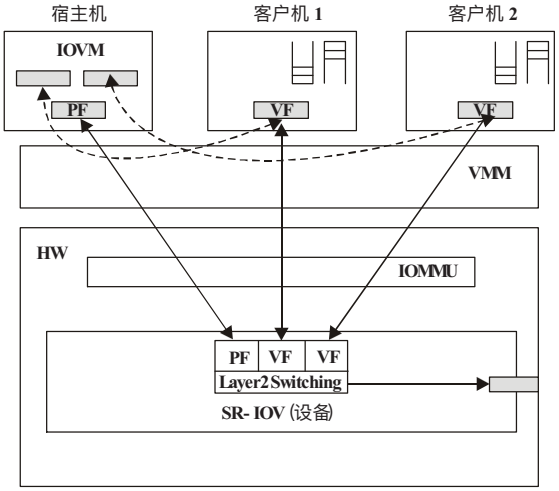

图 6　SR- IOV 的实现模型

为了使该结构独立于底层的 VMM, 每部分都不能使用特定的 VMM 接口。例如, PF 驱动和 VF 驱动的通信可以直接使用 SR- IOV 设备，其接口不会依赖于特定的 VMM 接口。

(1) PF 驱动　PF 驱动可以直接访问 PF 的所有资源, 并负责配置和管理所有 VF。它可以设置 VF 的数量, 全局的启动或停止 VF, 还可以进行设备相关的配置。 PF 驱动同样负责配置 2 层分发, 以确保从 PF 或 VF 进入的数据可以正确地路由。

(2) VF 驱动　VF 驱动像普通的 PCIe 设备驱动一样在客户机上运行，直接访问特定的 VF 设备完成数据的转移, 期间不需要 VMM 的干预。从硬件成本角度来说, VF 只需要模拟临界资源 (如 DMA 等) , 而对性能要求不高的资源则可以通过 IOVM 和 PF 驱动进行模拟。

(3) IOVM　IOVM 为每个 VF 分配了完整的虚拟配置空间，因此客户机能够像普通设备一样模仿和配置 VF。当宿主机初始化 SR- IOV 设备时, 它无法通过简单的扫描 PCIe 功能 (通过 PCIM 实现) 的供应商 ID 和设备 ID 列举出所有的 VF。因为 VF 只是修改过的"轻量级"功能, 不包含完整的 PCIEe 配置空间, 无法响应普通的 PCI 扫描。该模型可以使用 Linux PCI 热插拔 API 动态地为宿主机增加 VF, 然后将 VF 分配给客户机。

PCIM 负责对 PCI 设备进行扫描、识别, 可以将 VF 转换成完整的功能, 并对 SR- IOV 资源进行分配。由于 IOVM 为每个 VF 分配了虚拟的完整的配置空间, 一旦发现 VF 并分配给客户机，该客户机能够像使用普通 PCIe 功能那样初始化和配置该 VF。这部分功能可以在应用层面上完成, 如 Xen 中 HVM 的设备模式, 内核中的后端驱动。

影响 SR- IOV 性能关键是用于处理 I/O 设备的中断。在 SR- IOV 中, VMM 不再干预 I/O 操作, 从而提高了 I/O 设备的性能。2 层分发根据接收方地址对数据进行了分类，通过 DMA 直接将数据存储到接收方的缓冲区上, 并产生 MSI 或者 MSI- x。IOMMU 重映射了接收方 DMA 缓冲器地址, 将 VF 驱动的客户机物理地址转换为物理地址。VMM 可以捕捉该中断并根据向量识别到具体的客户机。然后, VMM 将虚拟 MSI 中断通知给客户机，并读出在当地缓冲器里的数据。客户机 VF 驱动处理虚拟中断，从本地缓冲区中读取接收的数据。从而是客户机在一次中断中能够处理多个接收的数据。

PF 和 VF 驱动之间需要通道进行配置、管理信息和事件通知的通信。例如, VF 驱动需要将客户机的服务请求发送给 PF 驱动。PF 驱动也需要将一些 I/O 事件转发给每个 VF 驱动, 通知资源状态的变化。这些事件包括等待全局设备重置、链接状态改变、驱动移除等。在该结构模型中, VF 和 PF 驱动之间的通信取决于底层硬件平台。例如, Intel SR- IOV 的 82576 Gbit 网卡通过简单的邮箱和门铃机制进行通信：发送方将消息写入邮箱, 然后按响"门铃", 这将会产生一个中断并通

知接收方消息已经发送。接收方接收信息后,将共享寄存器中的一位进行设置,表明信息已接收。

SR- IOV 提供一个安全的运行环境,允许 PF 驱动监控和实施 VF 设备的带宽分配、中断屏蔽、拥塞控制、广播和多播等,从而增加了 VM 之间的性能和安全隔离。PF 驱动监督 VF 驱动的请求,并对 VF 驱动行为和其使用的资源进行监控。PF 如果发现异常可以立即采取正确的行为。例如,PF 在发现异常时能够停止分配给某个虚拟机的 VF。

2.3　SR- IOV 的优势

SR－IOV 平台提供了一系列的技术优势,包括提高 IO 性能,提高系统的性价比,增加可扩展性,数据保护和安全性。

⑴ 增加系统性能

Passthrough 的优势:

a. VF 设备可以直接访问寄存器,IOMMU 技术使得 GPA (客户机物理地址) 转换为宿主机物理地址,这些使得客户机几乎可以达到本机的性能。与软件模拟的 I/O 设备相比,每个虚拟机能够通过较低的 CPU 开销获得很高吞吐量。

b. 传统的陷入和模拟 I/O 寄存器的读写和任务切换占用了大量的 CPU 利用率。VF 的另一个优势是可以直接进行 I/O 寄存器的读写,而不需要陷入和模拟,CPU 页表机制可以直接将 VF 设备的 MMIO 空间映射到客户机上面。

中断重映射的优势　IOMMU 技术可以改善中断重映射技术,减少客户机从硬件中断到虚拟中断的处理延迟。由于中断延迟是虚拟环境的主要瓶颈之一,采用 IOMMU 的 MSI- x 技术将大大减少中断延迟,降低了由 VMM 处理 I/O 导致的系统开销,提高系统性能。

共享性的优势　Passthrough 技术将设备分配给指定的虚拟机,可以达到几乎本机的性能,其缺点是整个 I/O 设备只能供一个虚拟机使用。这种方式违背了虚拟化的本意,即 I/O 资源的共享是为使得硬件利用的最大化。SR- IOV 技术能够使分配给每个虚拟机的 VF 都达到其最高性能,这使得所有虚拟机能够充分利用 I/O 设备资源,达到该设备的最高性能。

⑵ 减轻系统管理员负担

减少物理设备的优势　使用 VF 替代多个物理 I/O 设备,降低硬件开销,简化布线,降低功耗,减少转换器端口的使用数目,降低设备数目。

安全性优势　通过硬件辅助数据保护和安全得到了加强,使得数据和 I/O 流在虚拟机之间的创建和隔离得到了增强。

可扩展性优势　系统管理员可以使用单个更高带宽的 I/O 设备代替多个带宽较低的设备达到带宽的要求。利用 VF 将带宽进行隔离,使得单个物理设备好像是隔离的多物理设备。此外,这还可以为其他类型的设备节省插槽。

⑶ 简化虚拟机设计

通用性优势　SR- IOV 技术不需要在客户机中安插任何前端驱动,也不需在 VMM 中维护任何后端驱动,没有额外的维护开销。

减少对宿主机依赖　SR- IOV 技术不依赖宿主机进行 I/O 操作,所以当运行的客户机数量多时,不会增加宿主机的负荷。

参考文献:

[1] K. Fraser, S. Hand , R. Neugebauer, I. Pratt, A. Warfield, and M. Williamson. Safe hardware access with the Xen virtual machine monitor [C]. In 1st Workshop on Operating System and Architectural Support for the on demand IT InfraStructure(OASIS), 2004.10.

[2] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel. Diagnosing Performance Overheads in the Xen Virtual Machine Environment[C]. In First ACM/USENIX Conference on Virtual Execution Environments(VEE'05), 2005.6.

[3] Paul Barham, Boris Dragovic, Keir Fraser. Xen and the art of virtualization[J]. ACM Press, 2003:164- 177.

[4] J. Sugerman, G. Venkitachalam, and B.- H. Lim. Virtualizing I/O devices on VMware workstation's hosted virtual machine monitor[C]. In Proceedings of the USENIX Annual Technical Conference, 2001.6.

[5] SR- IOV Networking in Xen: Architecture, Design and Implementation Yaozu Dong, Zhao Yu, and Greg Rose. SR- IOVnetworking in Xen: Architecture, design andimplementation [C]. In WIOV '08: Proceedings of the 1stWorkshop on I/O Virtualization, December 2008.

[6] G. Neiger, A. Santoni, F. Leung, D. Rodgers, R. Uhlig. Intel virtulization technology: Hardware support for efficient processor virtualization[C]. Intel Technology Journal, 2006.

[7] Barham P, Dragovic B, Frase K, Hand S, Harris T, Ho A, Neugebauer R, Pratt I, Warfield A. Xen and the art of virtualization [C]. proceedings of 19th ACM Symposium on Operating Systems Principles, October, 2003.

[8] Liu Jx, Huang W, Abali B, K.Panda D. High performance vmm- bypass i/o in virtual machines [C]. proceedings of the USENIX Annual Technical Conference, May, 2006.

[9] Intel Corporation. Intel virtualization technology for direct I/O [J]. Intel technology Journal 10(03):205- 216.September, 2008.

[10] Y. Dong, J. Dai, et al. Towards high- quality I/O virtualization [C]. Proceeding of the Israeli Experimental Systems Conference (SYSTOR), Haifa, Israel 2009.

[11] PCI- SIG Single Root I/O Virtualization 1.0 Specification [EB/OL]. http://www.pcisig.com/specifications/iov/single_root

[12] PCI Special Interest Group[EB/OL]. http://www.pcisig.com/home.