[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 15/173 (2006.01)
G06F 9/54 (2006.01)
G06F 9/455 (2006.01)
H04L 29/06 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200810239899.7

[43] 公开日 2009 年 5 月 13 日

[11] 公开号 CN 101430674A

[22] 申请日 2008.12.23

[21] 申请号 200810239899.7

[71] 申请人 北京航空航天大学

地址 100191 北京市海淀区学院路 37 号北京航空航天大学计算机科学与工程学院

[72] 发明人 宋忠雷 肖利民 陈思名 彭近兵 祝明发 马 博

[74] 专利代理机构 北京慧泉知识产权代理有限公司

代理人 王顺荣 唐爱华

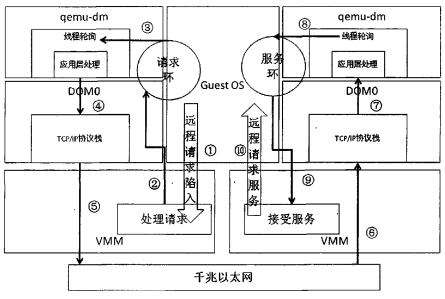
权利要求书 2 页 说明书 10 页 附图 3 页

[54] 发明名称

一种分布式虚拟机监控器内连通信方法

[57] 摘要

本发明提供一种分布式虚拟监控器通信技术的实现方法，它主要采用扩展系统设备模拟部分，结合描述符环机制，来利用现有的可靠传输协议实现 VMM 间可靠高效的通信，提供资源整合所必需的基础，为上层客户操作系统提供一个单一物理节点，实现客户操作系统对多节点资源的管理和使用。 该实现方法具体步骤如下：步骤一、准备阶段；步骤二、建立连接阶段；步骤三、数据传输过程；步骤四、连接关闭阶段。 本发明在现有的成熟技术基础上进行创新，实施简便，具有良好的使用和发展前景。



1、一种分布式虚拟机监控器通信技术的实现方法，其特征在于：该实现方法步骤如下：

步骤一、准备阶段：

各结点 VMM 创建虚拟机并为虚拟机分配内存时为双环预留空间，告知虚拟机页面不可用，并初始化请求环和服务环结构；

各节点 VMM 分别在启动虚拟机时启动设备模拟模块 dm，并将双环所在的页面映射到 domain0 地址空间内，以供 domain0 和工作在 domain0 之上的 dm 模块中应用程序使用；

在各节点 VMM 分别事件通道的初始化，事件通道由 VMM 管理，并呈现给 domain0 使用；

步骤二、建立连接阶段：

当客户操作系统启动时：

1. 由管理模块启动客户操作系统，首先陷入到 VMM 中，VMM 对客户操作系统进行分配核心资源后启动为客户操作系统提供设备模拟的 qemu-dm 模块；

启动 qemu-dm 模块中我们所实现的通信模块部分：

为提高数据传输率，我们使用三个连接进行通信，分别是不同物理节点上的 vcpu 间，远程 I/O 设备访问，远程访存；

qemu-dm 模块运行在 domain0 的应用层，各节点分别从客户操作系统的配置文件读取参数，将参数传递过来，比如节点 IP 地址；启动节点创建 socket，开始对连接进行监听，而非启动节点创建 socket 后发起连接请求；这个过程为阻塞方式，也就是说只有建立好连接之后才可往下进行；

通过系统调用方式进入到 domain0 的内核态使用 TCP/IP 协议栈完成连接的建立；分别将建立好的连接的描述符保存到连接数组，以供数据发送时使用；

步骤三、数据传输过程：

当客户操作系统发生缺页或者 I/O 设备访问时会首先陷入到 VMM 中；

VMM 对陷入原因进行分析，根据不同请求创建不同的请求，将需要发送的数据的地址映射到 domain0，然后通过写指针加入到请求环中，最后设置事件通道；

qemu-dm 模块中通信线程被 VMM 以事件通道的方式唤醒，开始对请求环进行轮询，当有数据需要发送会通过读指针将请求取出来，并根据请求组建数据包，为数据包加应用层头部；

数据传入 domain0 中的 TCP/IP 协议栈，通过网卡将数据发送出去；

目的节点接收到数据后由网卡中断交由 TCP/IP 协议栈，协议栈在去掉传输层以下的头部后将数据保存，并将地址保存到一个环描述符中，通过服务环写指针将它放入到服务环中去，然

后通过事件通道告知 VMM 来处理；

VMM 读取服务环，得到描述符中的地址后将数据映射到客户操作系统中去，这样接收数据完成；

步骤四、连接关闭阶段：

客户操作系统关闭；

释放启动节点资源并通过通信模块释放远程资源；

检查环中数据是否已经处理完毕；

将连接关闭；

释放环以及与通信相关资源；

通信线程服务退出。

一种分布式虚拟机监控器内连通信方法

（一）技术领域

本发明涉及一种分布式虚拟机监控器内连通信方法，属于计算机系统虚拟化技术，尤其是涉及一种以实现服务器机群虚拟化目标的分布式虚拟机监视器系统的内连通信技术。属于计算机技术领域。

（二）背景技术

虚拟化技术

单机虚拟化逐步成熟

虚拟化技术自从一出现便受到人们广泛的关注，现在，越来越多的厂商开始介入，从处理器层面的 AMD 和 Intel 到操作系统层面的微软的加入，从数量众多的第三方软件厂商的涌现到服务器系统厂商的高调响应。可见虚拟化技术发展迅猛。

Xen 是由剑桥大学计算机实验室开发的一个开源虚拟机监控器（VMM），在其上可以同时创建多个虚拟机，每个虚拟机运行一个操作系统。Xen 采用的虚拟化技术被称为 para-virtualization（半虚拟化），即 VMM 对客户操作系统的一般指令不予理睬，对操作系统的敏感指令需要使用超级调用（hypercall）来代替。这样极大的提高了系统的隔离性和性能，但是有一个缺点就是它需要用户对操作系统进行修改。随着 AMD 和 Intel 推出硬件辅助虚拟化支持技术，Xen 也支持全虚拟化和硬件辅助虚拟化。

VMware 公司是目前市场上的领军人物，它采用全虚拟化（Full-Virtualization）技术。全虚拟化技术就是客户操作系统不需要作任何改动可以直接运行在虚拟机上，它能允许用户使用常规的，无需修订的操作系统来作为客户端。

多机虚拟化走上舞台

随着应用对计算机资源的增长，单机资源已经满足不了用户的需求，如何突破单宿主机限制成为虚拟化技术的另一重要发展方向。

日本东京大学的 Virtual Multiprocessor 项目是基于 IA-32 机群的分布式 VMM。这个项目中，在 VMM 和硬件层之间，运行了一个简化的操作系统。它可以由这层 host OS 提供支持。Virtual Multiprocessor 采用了半虚拟化的技术，通过对客户操作系统的修改使其与 Virtual Multiprocessor 协作完成任务。Virtual Multiprocessor 运行于用户态，其主要功能依赖对宿主操作系统的系统调用，因而效率低下。其分布式 VMM 之间的通信是通过调用下层多操作系统

的 TCP 协议完成，节点间的通信由操作系统负责。

vNUMA 是澳大利亚新南威尔士大学开发的基于 IA-64 机群的分布式 VMM,其运行于最底层。客户操作系统(Linux)通过准虚拟化的方式与 vNUMA 协作。vNUMA 的主要目标在于提供透明的分布式共享存储用于科学计算。vNUMA 系统采用了一种被称为预虚拟化的方法，该方法由德国 Karlsruhe 大学、澳大利亚新南威尔士大学和 IBM 共同提出。这是一种提供工具支持的半自动 Guest 构造方法，利用汇编器的支持，对 Guest 系统的代码进行扫描，将其中的部分特权指令进行静态替换，对无法静态处理的指令采用 profile 方法动态地寻找并手工替换。这个方法的指导思想是采用编译工具支持，尽可能增加可以直接运行的指令，减少需要模拟的指令。

跨节点的通信系统

VMM 间的通信

Virtual Multiprocessor 通信

日本东京大学的 Virtual Multiprocessor 项目也是实现分布式虚拟机。这个项目中，在 VMM 和硬件层之间，运行了一个简化的操作系统，称为 Host OS。VMM 间的通信使用自己的应用协议，它使用 Host OS 中基于以太网的 TCP/IP 协议栈来完成数据的发送与接收。它的优点是通信协议使用现有的 TCP/IP 协议栈，给实现带来极大的方便。而缺点也是很显然的，它需要有 host OS 的支持，通信属于应用级的，效率上要低。

vNUMA 通信

vNUMA 是澳大利亚新南威尔士大学开发的一种基于 IA-64 机群的分布式 VMM。它是基于安腾体系结构的 VMM 直接运行在硬件上，底层通信直接在硬件上实现。但是它是非开源的。它的优点是由硬件实现效率上高，缺点就是专用硬件支持，给实现带来了困难。

其它跨结点间高效通信机制

VMMC 通信机制

Virtual Memory-Mapped Communication(VMMC) 是一种基于虚拟内存映射的通信机制，它支持从发送方虚拟内存到接收方虚拟内存的数据直接传送。支持数据零拷贝，但是它需要专用的硬件支持。

Active Messages

它是第一个被广泛使用的 ULN(User Level Networking)，最初是为并行的微机开发的，在不同的网络接口硬件上应用，接着发展成为一种用于通讯的汇编语言。它是不需要任何协议支持的，更加接近于硬件，效率高，但是在死锁处理和同步上较差。

Fast Sockets

它用的是 GAM(Globally Addressable Memory) 接口，并不将底部的队列结构展示出来，排除了处理程序的隐含执行，而是由用户进程为即将到来的消息提供缓存管理并管理流量以避

免死锁。它被广泛应用于机群互连，并推动了为解决由大批传送引起的节点争夺问题等相关协议的发展。虽然它只需要少量的数据拷贝，但是需要专用的平台接口才能实现。

综上所述，目前的 VMM 间通信主要有两种，一种是在基于 host OS 的 VMM 中，通信靠 host OS 中的协议栈来完成；另外一种是基于硬件的 VMM，它在专有通信硬件的基础上完成通信。本发明阐述的通信机制基于千兆以太网，为直接运行于硬件之上的 VMM 提供通信。

（三）发明内容

本发明的目的在于提供一种分布式虚拟机监控器内连通信方法，它主要采用环机制与可靠传输协议结合，并辅助以有效的通告机制，利用高速互连局域网网络，为分布式虚拟机监控器提供可靠高效的通信，完成集群资源的整合。

本发明所述的方法基于集群系统--一种通过外部互联网络连接的多计算机系统，其特点是各物理资源分布于多个节点之上，通过对各节点资源的虚拟化及模块协作，完成对服务器资源的整合，集群中的计算机需要通过网络传递消息的方式进行合作。本发明专利的目标是为在机群系统之上利用虚拟化技术提供具有对称式多处理器（Symmetric Multi-Processors，SMP）特性的虚拟机监控器提供可靠高效的通信。分布式虚拟机监控器的特点是直接工作于物理硬件之上，而无需操作系统的帮助。

本发明专利通过在机群节点上部署 VMM，在机群系统的物理结构上提供具有 SMP 特性的虚拟机监控器。并在 VMM 运行修改过的 linux 操作系统，称之为设备 domain(Dom0)，并由它提供可靠传输协议栈。通过在 VMM 中实现与设备 domain 高效的交互机制，来完成对设备 domain 协议栈的调用，从而达到 VMM 间可靠高效的数据传输，使支持 SMP 结构的商业化的操作系统无需修改即可运行在该虚拟机中。

VMM 的通信模块负责为布署于各物理结点之上的 VMM 提供可靠高效的数据传输，使 VMM 完成对多机资源的整合，为客户操作系统呈现单一的镜像。

本发明一种分布式虚拟机监控器内连通信方法，具体实现步骤如下：

步骤一、准备阶段：

各结点 VMM 创建虚拟机并为虚拟机分配内存时为双环预留空间，告知虚拟机页面不可用，并初始化请求环和服务环结构；

各节点 VMM 分别在启动虚拟机时启动设备模拟模块 dm，并将双环所在的页面映射到 domain0 地址空间内，以供 domain0 和工作在 domain0 之上的 dm 模块中应用程序使用；

在各节点 VMM 分别事件通道的初始化，事件通道由 VMM 管理，并呈现给 domain0 使用；

步骤二、建立连接阶段：

当客户操作系统启动时：

1. 由管理模块启动客户操作系统，首先陷入到 VMM 中，VMM 对客户操作系统进行分配核心资源后启动为客户操作系统提供设备模拟的 qemu-dm 模块；

启动 qemu-dm 模块中我们所实现的通信模块部分：

为提高数据传输率，我们使用三个连接进行通信，分别是不同物理节点上的 vcpu 间，远程 I/O 设备访问，远程访存；

qemu-dm 模块运行在 domain0 的应用层,各节点分别从客户操作系统的配置文件读取参数，将参数传递过来，比如节点 IP 地址。启动节点创建 socket,开始对连接进行监听，而非启动节点创建 socket 后发起连接请求；这个过程为阻塞方式，也就是说只有建立好连接之后才可往下进行；

通过系统调用方式进入到 domain0 的内核态使用 TCP/IP 协议栈完成连接的建立；分别将建立好的连接的描述符保存到连接数组，以供数据发送时使用；

步骤三、数据传输过程：

当客户操作系统发生缺页或者 I/O 设备访问时会首先陷入到 VMM 中；

VMM 对陷入原因进行分析，根据不同请求创建不同的请求，将需要发送的数据的地址映射到 domain0,然后通过写指针加入到请求环中，最后设置事件通道；

qemu-dm 模块中通信线程被 VMM 以事件通道的方式唤醒，开始对请求环进行轮询，当有数据需要发送会通过读指针将请求取出来，并根据请求组建数据包，为数据包加应用层头部；

数据传入 domain0 中的 TCP/IP 协议栈，通过网卡将数据发送出去；

目的节点接收到数据后由网卡中断交由 TCP/IP 协议栈，协议栈在去掉传输层以下的头部后将数据保存，并将地址保存到一个环描述符中，通过服务环写指针将它放入到服务环中去，然后通过事件通道告知 VMM 来处理；

VMM 读取服务环，得到描述符中的地址后将数据映射到客户操作系统中去，这样接收数据完成；

步骤四、连接关闭阶段：

客户操作系统关闭；

释放启动节点资源并通过通信模块释放远程资源；

检查环中数据是否已经处理完毕；

将连接关闭；

释放环以及与通信相关资源；

通信线程服务退出。

本发明一种分布式虚拟机监控器内连通信方法，其优点及功效在于：通过利用现有的可靠

传输协议栈，并通过双环和事件通道等机制的结合来完成分布式 VMM 的通信，本发明提高了分布式 VMM 系统中通信的高效性和可扩展性，并且本发明在现有的成熟技术基础上进行创新，实施更加容易，具有良好的使用和发展前景。

（四）附图说明

图 1 DVMM 系统整体结构示意图

图 2 两结点系统模块示意图

图 3 通信整体架构示意图

图 4 两结点通信模型示意图

图 5 描述符环结构示意图

图 6 通信过程详细示意图

（五）具体实施方式

请参阅图 1 至 5 所示，本发明一种分布式虚拟监控器内连通信方法，

1. 方法概述

本发明专利基于集群系统--一种通过外部互联网络连接的多计算机系统，其特点是各物理资源分布于多个节点之上，通过对各节点资源的虚拟化及模块协作，完成对服务器资源的整合，集群中的计算机需要通过网络传递消息的方式进行合作。本发明专利的目标是为在机群系统之上利用虚拟化技术提供具有对称式多处理器（Symmetric Multi-Processors，SMP）特性的虚拟机监控器提供可靠高效的通信。分布式虚拟机监控器的特点是直接工作于物理硬件之上，而无需操作系统的帮助。

本发明专利通过在机群节点上部署 VMM，在机群系统的物理结构上提供具有 SMP 特性的虚拟机监控器。并在 VMM 运行修改过的 linux 操作系统，称之为设备 domain(Dom0)，并由它提供可靠传输协议栈。通过在 VMM 中实现与设备 domain 高效的交互机制，来完成对设备 domain 协议栈的调用，从而达到 VMM 间可靠高效的数据传输，使支持 SMP 结构的商业化的操作系统无需修改即可运行在该虚拟机中。

VMM 的通信模块负责为布署于各物理结点之上的 VMM 提供可靠高效的数据传输，使 VMM 完成对多机资源的整合，为客户操作系统呈现单一的镜像。

2. 分布式 VMM 通信的特点

分布式 VMM 系统是直接运行在物理硬件之上的，负责管理和整合硬件资源，为运行在上层的操作系统服务。通信作为分布 VMM 系统的一部分，为实现整个分布式 VMM 系统的其他功能服务。

由于VMM本身并不具有可靠传输层协议，并且不负责外部设备的管理（以太网网卡），而这些是设备domain所有的，为利用以太网完成VMM间的可靠通信，需要VMM本身实现与设备domain的高效交互机制，完成对设备domain中协议栈的使用。这样充分利用了设备domain，而不必在VMM实现繁杂的传输层协议和网卡驱动，使VMM更具有可扩展性。

3. 系统结构

VMM通信按功能流程分为以下几个模块：

模块一、VMM通信的预处理。

当Guest OS引发远程请求时会陷入到VMM中，这时VMM对Guest OS的陷入原因进行分析，对不同的通信进行区分，并加入不同的头部，构成了应用层的头部。在DVMM系统中，通信的种类主要有：

- IPI，每次传输一个寄存器的内容。
- 远程设备访问，每次传输IOREQ和控制信息，不超过100字节。
- DSM，每次通信的数据量是一页。
- 远程I/O操作，每次传输一个指令。

模块二、VMM与dom0上DM模块中通信线程的交互。

VMM为通信的两端，处于内核态，即硬件之上，而qemu-dm模块运行于domain0之上，处于用户态，两者必须有数据交互。为了使VMM与Dom0交换数据，引入双环机制，分别称为请求环和服务环。由于发送或接收的数据通常比较大，VMM和Dom0并不交换实体数据，而以引用的方式来通告。环是存放发送数据控制结构的，称为描述符环。VMM将要发送的数据或者Dom0将接收的数据的地址保存在环中一项中，并通过VMM中的授权表将数据所在页面授权给Dom0由它映射过来完成发送，这样既减少了数据的持贝，提高了效率，又增大了环的容量。

授权表是为实现不同domain之间数据共享的，在DVMM中，客户操作系统与Dom0可以看作是对等的两个domain，VMM应保证两者的独立性、安全性。然而通信需要让dom0发送GOS中的数据，或者接收到数据提供给GOS，为减少数据的拷贝，需要实现两者之间的数据共享。

显然，对于请求环和服务环来说，它们应该能被VMM和Dom0共同访问，而且最重要一点是，它们所处的位置必须安全的，不能被其它模块所修改。基于以上考虑，发送环和接收环从Guest OS的内核页面中预留出来，并限制Guest OS不能使用此页面。然后通过映射的方式以供Dom0来使用。Guest OS的页面是由VMM来管理的，所以它也可以使用。

如图5所示，请求环和服务环结构相同，每一个环有两个指针，分别为读和写，由VMM和dom0共同使用，一端读一端写，VMM在请求通信时会通过写指针将请求放入到环中，而在另一端的dom0通过读指针读取请求进行处理将数据发送出去，对于服务环，Dom0收到数据后，新建环一项结构通过写指针将它放入到服务环，然后VMM通过读指针将服务读取并处理。

事件通道机制是为解决 VMM 与 Dom0 之间的协作。在 DVMM 系统中，为提高通信的效率，通信采用异步方式， VMM 通过事件通道机制来通知 dom0 内核服务器来处理请求。这样 VMM 在不影响其性能的情况下也保障了通信的高效。

事件通道本质上设置掩码位，每一位代表一个通道，由 VMM 和 Dom0 两者共同使用，当需要触发 Dom0 事件时，比如 VMM 发起通信请求，将数据准备好后，通过对事件通道的置位，并由 Dom0 查询得知有数据请求需要处理。Dom0 也可以对它进行修改，不过由于它是由 VMM 来维护的，所以需要通过超级调用来完成。

模块三、qemu-dm 模块中实现通信线程服务器

在 DVMM 系统中，qemu-dm 模块本身负责的是 I/O 模拟，我们对 qemu-dm 进行的扩展，使其在负责 I/O 模拟的同时还负责节点间可靠消息的传递。通信线程主要负责连接的建立，请求和服务环的轮询，数据的发送与接收。

在 Guest OS 创建时，qemu-dm 负责建立到其他不同节点对应 qemu-dm 的 TCP 链接。通信线程会分别对 TCP 套接字以及请求环进行轮询。当通信线程发现此次消息的目的是 qemu-dm 时，它会将消息传递给 qemu-dm；当通信线程发现此次消息的目的是 VMM 时，其会将消息直接放入服务环并利用事件通道通知 VMM 消息的到来。

每当 VMM 需要传递信息时，其会将通信的目的以及消息放入请求发送环并利用事件通道的方式通知通信线程。通信线程从请求发送环中取走对应的消息后会选择与目的节点对应的 TCP 链接，并将这个消息通过 socket 发送至目的节点。

通信线程会对每一条消息做简单的解释，当其发现 VMM 需要发送页面时，其首先会将该页面映射至自己的地址空间中，让后直接从该页面获取数据并发送至对应的 socket。与发送的过程类似，当通信线程发现远程的页面数据到达时，它首先会将目的页面映射至自己的线性地址空间中，并直接将 socket 中的页面数据读入目的客户页面中。

由于在多节点系统中，每个通信线程可能维护多个 TCP 连接，所以通信线程还还需要负责在请求发送环与 TCP 套接字之间转发或路由对应的包。VMM 仅仅负责按照节点编号进行通信，节点编号所对应的 IP 地址对 VMM 透明。所以通信线程必须负责将 VMM 所需要传递的包按照其目的节点对应的 IP 地址，从对应的 TCP 连接中发送出去。

模块四、domain0 中协议栈对数据的传输

Dom0 中提供的协议栈以及网络设备驱动程序：由于在 VMM 中，外部设备由 dom0 负责管理，所以我们只能复用 dom0 提供的网络设备驱动程序与可靠传输协议（TCP），TCP 协议为面向连接的通信，而且本身提供了重传机制等，因而可以实现 VMM 间的可靠通信。

通过多种机制的结合，通信系统为 DVMM 系统提供简单、可靠但高效的数据传输。

如图6中所示，当Guest OS需要远程页面或I/O访问时，会①陷入到VMM中，VMM根据Guest OS陷入原因处理请求②将需要发送的数据请求放入到发送环中，此时处于③qemu-dm模块中的应用级进程轮询此请求，当发现有数据需要发送时进行④应用层的处理并调用dom0中的协议栈将数据通过⑤网卡从以太网上发送出去。接收方⑥网卡收到数据后中断由协议栈接收，⑦交由应用层处理，⑧放入接收环，并⑨告知VMM，⑩VMM服务于Guest OS.这样整个发送接收过程就完成了。

4. 系统工作流程

初始化阶段：

通信模块的初始化

由于通信协议是面向连接的C/S模式，因此初始化时各节点并不是对称的系统在初始化阶段区分两类节点：系统选取一个节点作为启动节点，其余节点作为非启动节点。启动节点以server端建立socket，并开始对端口监听连接，而非启动节点建立socket后呼叫连接，将连接保存到数组中。

环的初始化：

双描述符环实现在客户操作系统的保留页面中，在客户操作系统启动进进行页面的初始化清零，然后设置读写指针，读写指针均指向环起始位置，根据两者差值判断是否有数据。

服务的建立：

DVMM系统通过对qemu-dm模块进行扩展，加入两个通信线程，其中一个为负责发送数据的线程，它对请求发送环进行轮询对进行处理。另一个是负责接收数据的线程，它由设备domain0中协议栈引发，当协议栈接收到数据后放入服务环中，以供VMM处理。线程在qemu-dm模块启动过程建立。

系统正常工作阶段：

分布式VMM实现多节点的资源整合，为提供不同节点上VMM间的模块通信。

连接建立：多节点间利用socket建立连接，启动节点为server端，其它节点为client端。

数据发送：VMM有数据要发送将请求置入到请求环中，通知qemu-dm模块处理，然后利用TCP/IP协议栈发送。

数据接收：TCP/IP协议栈接收后放入到服务环中，由VMM处理。

下面结合附图，详述具体实施步骤如下：

步骤一、准备阶段：

各结点VMM创建虚拟机并为虚拟机分配内存时为双环预留空间，告知虚拟机页面不可用，并初始化请求环和服务环结构；

各节点VMM分别在启动虚拟机时启动设备模拟模块dm,并将双环所在的页面映射到domain0

地址空间内，以供 domain0 和工作在 domain0 之上的 dm 模块中应用程序使用；

在各节点 VMM 分别事件通道的初始化，事件通道由 VMM 管理，并呈现给 domain0 使用。

步骤二、建立连接阶段：

当客户操作系统启动时：

1. 由管理模块启动客户操作系统，首先陷入到 VMM 中，VMM 对客户操作系统进行分配核心资源后启动为客户操作系统提供设备模拟的 qemu-dm 模块；

启动 qemu-dm 模块中我们所实现的通信模块部分：

为提高数据传输率，我们使用三个连接进行通信，分别是不同物理节点上的 vcpu 间，远程 I/O 设备访问，远程访存；

qemu-dm 模块运行在 domain0 的应用层,各节点分别从客户操作系统的配置文件读取参数，将参数传递过来，比如节点 IP 地址。启动节点创建 socket,开始对连接进行监听，而非启动节点创建 socket 后发起连接请求。这个过程为阻塞方式，也就是说只有建立好连接之后才可往下进行；

通过系统调用方式进入到 domain0 的内核态使用 TCP/IP 协议栈完成连接的建立,分别将建立好的连接的描述符保存到连接数组，以供数据发送时使用。

步骤三、数据传输过程：

当客户操作系统发生缺页或者 I/O 设备访问时会首先陷入到 VMM 中；

VMM 对陷入原因进行分析，根据不同请求创建不同的请求，将需要发送的数据的地址映射到 domain0,然后通过写指针加入到请求环中，最后设置事件通道；

qemu-dm 模块中通信线程被 VMM 以事件通道的方式唤醒，开始对请求环进行轮询，当有数据需要发送会通过读指针将请求取出来，并根据请求组建数据包，为数据包加应用层头部；

数据传入 domain0 中的 TCP/IP 协议栈，通过网卡将数据发送出去；

目的节点接收到数据后由网卡中断交由 TCP/IP 协议栈，协议栈在去掉传输层以下的头部后将数据保存，并将地址保存到一个环描述符中，通过服务环写指针将它放入到服务环中去，然后通过事件通道告知 VMM 来处理；

VMM 读取服务环，得到描述符中的地址后将数据映射到客户操作系统中去，这样接收数据完成。

步骤四、连接关闭阶段：

客户操作系统关闭；

释放启动节点资源并通过通信模块释放远程资源；

检查环中数据是否已经处理完毕；

将连接关闭；

释放环以及与通信相关资源；

通信线程服务退出。

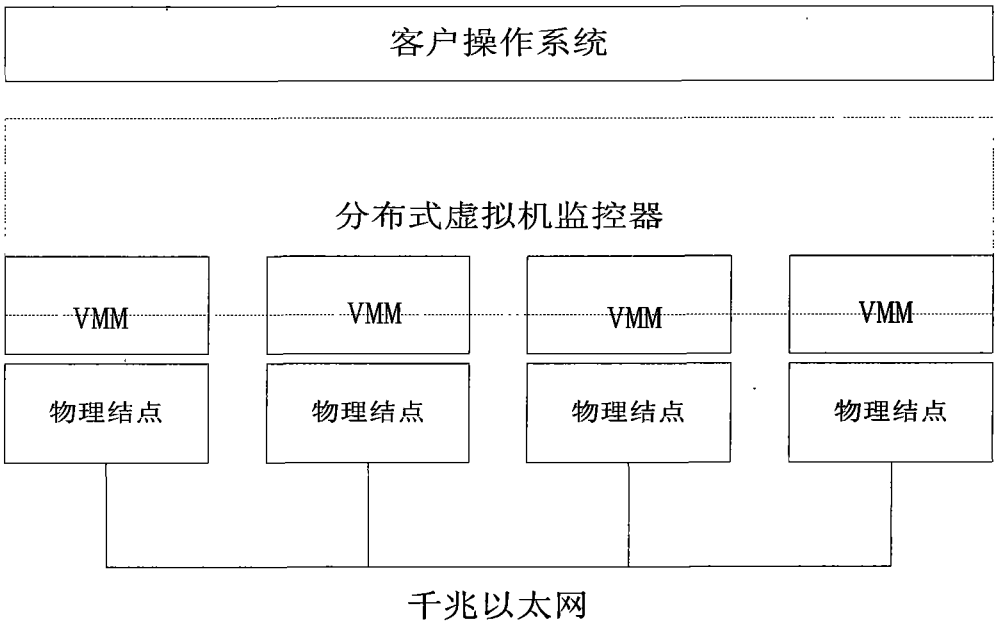

客户操作系统
分布式虚拟机监控器
VMM
VMM
VMM
VMM
物理结点
物理结点
物理结点
物理结点
千兆以太网

图 1

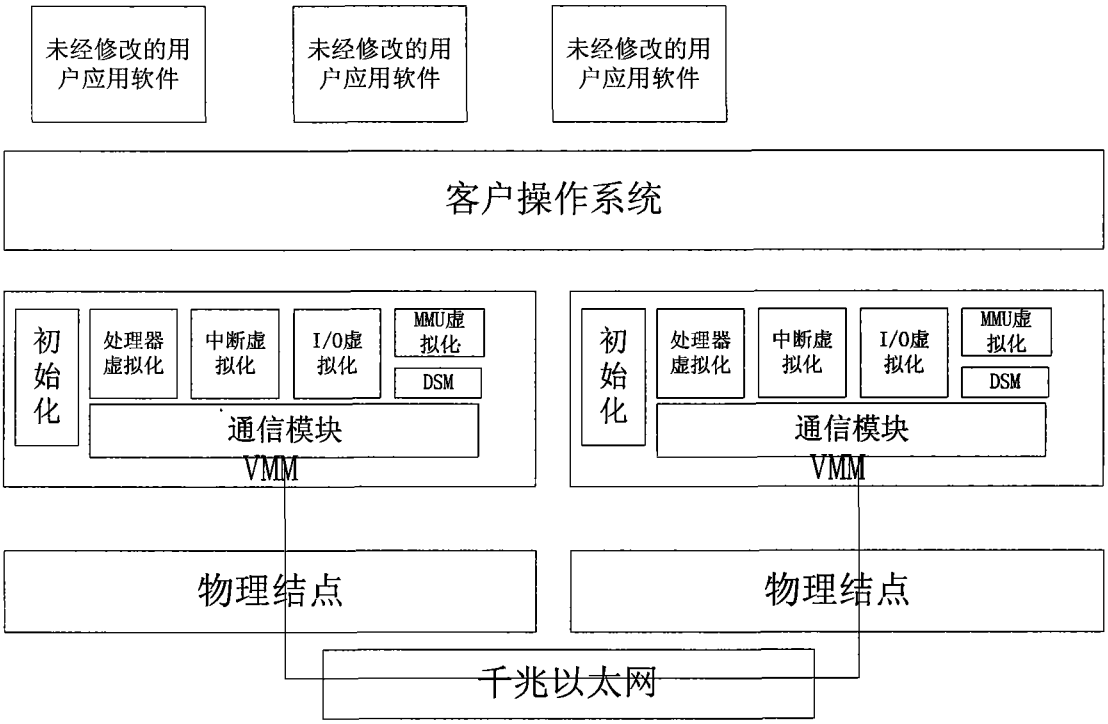

未经修改的用户应用软件
未经修改的用户应用软件
未经修改的用户应用软件
客户操作系统
初始化
处理器虚拟化
中断虚拟化
I/O虚拟化
MMU虚拟化
DSM
通信模块
VMM
初始化
处理器虚拟化
中断虚拟化
I/O虚拟化
MMU虚拟化
DSM
通信模块
VMM
物理结点
物理结点
千兆以太网

图 2

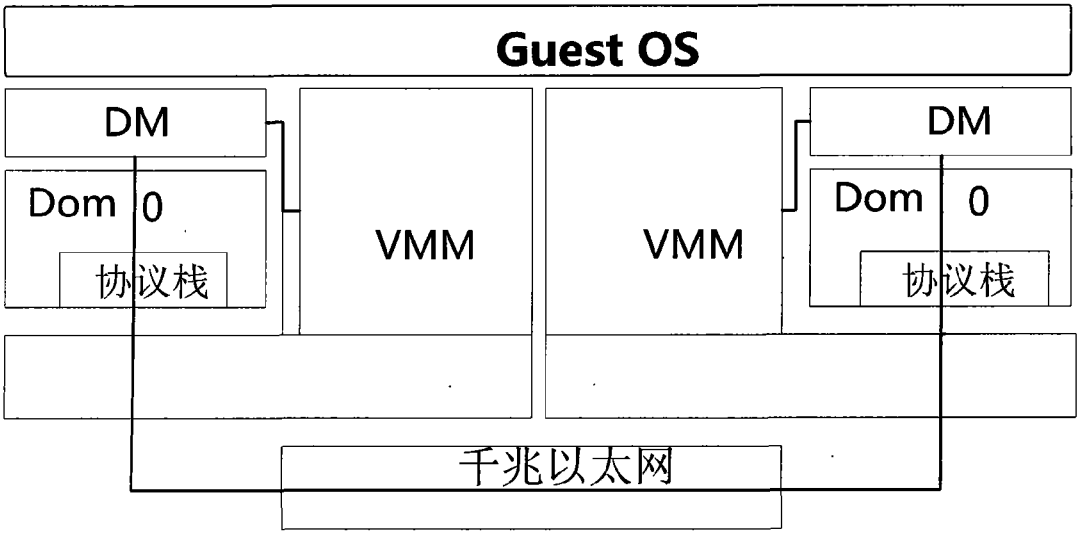

Guest OS
DM
Dom 0
协议栈
VMM
VMM
DM
Dom 0
协议栈
千兆以太网

图 3

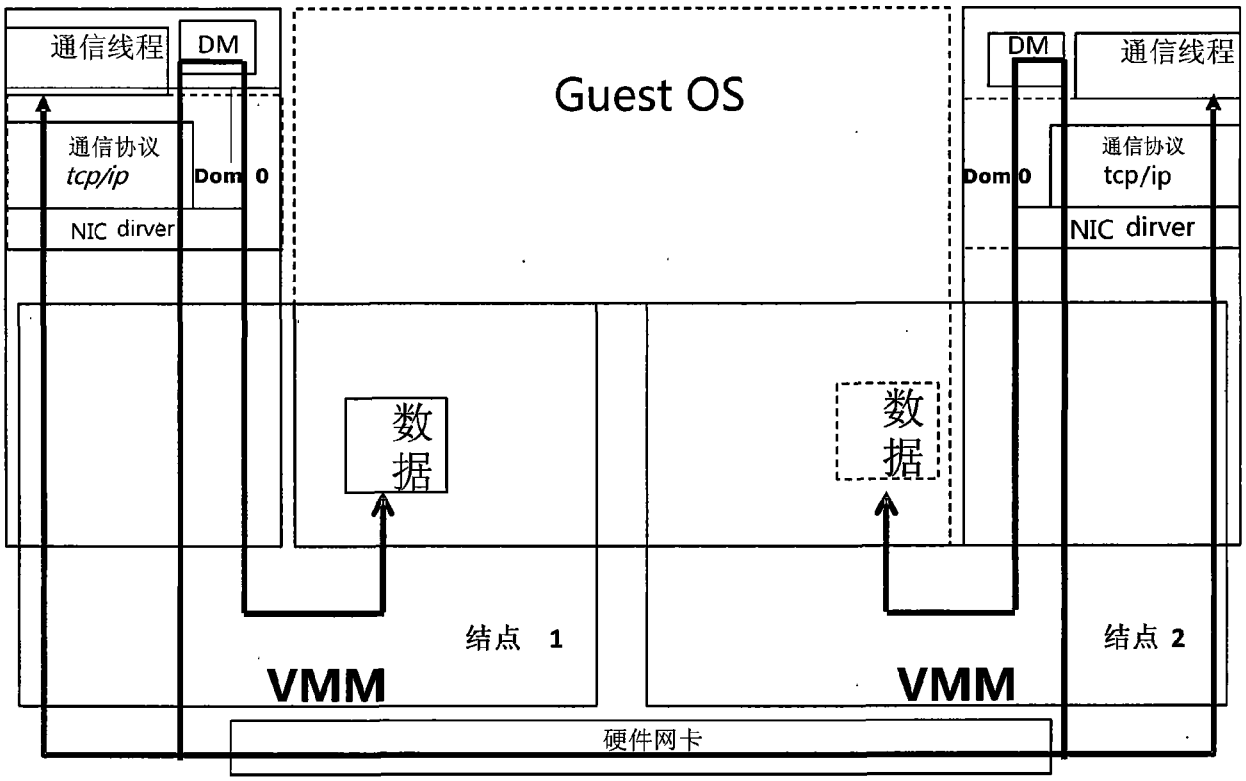

通信线程
DM
通信协议
tcp/ip
Dom 0
NIC dirver
Guest OS
数据
结点 1
VMM
DM
通信线程
通信协议
tcp/ip
Dom 0
NIC dirver
数据
结点 2
VMM
硬件网卡

图 4

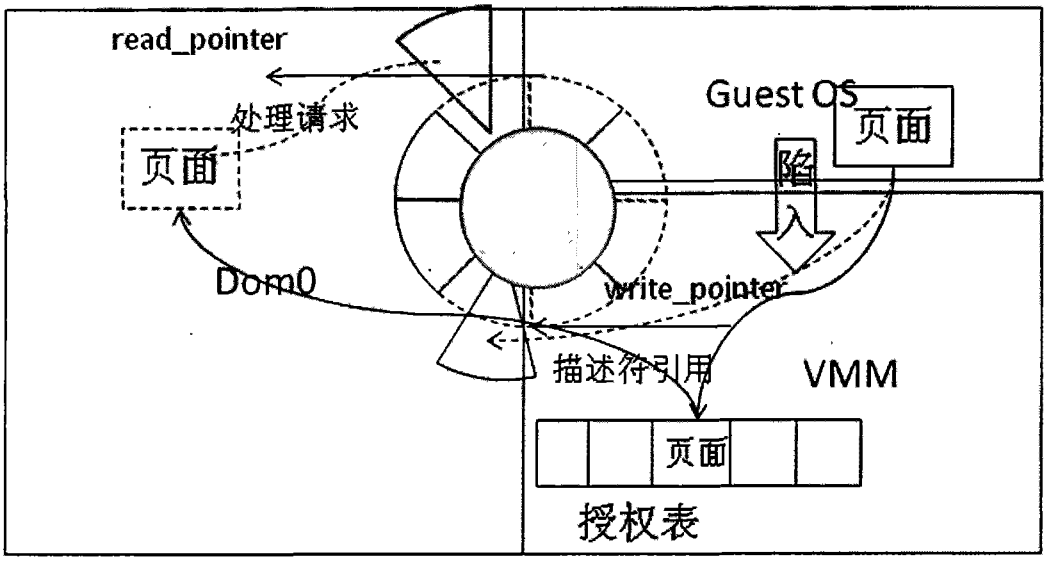

read_pointer
处理请求
页面
Guest OS
页面
陷入
Dom0
write_pointer
描述符引用
VMM
页面
授权表

图 5

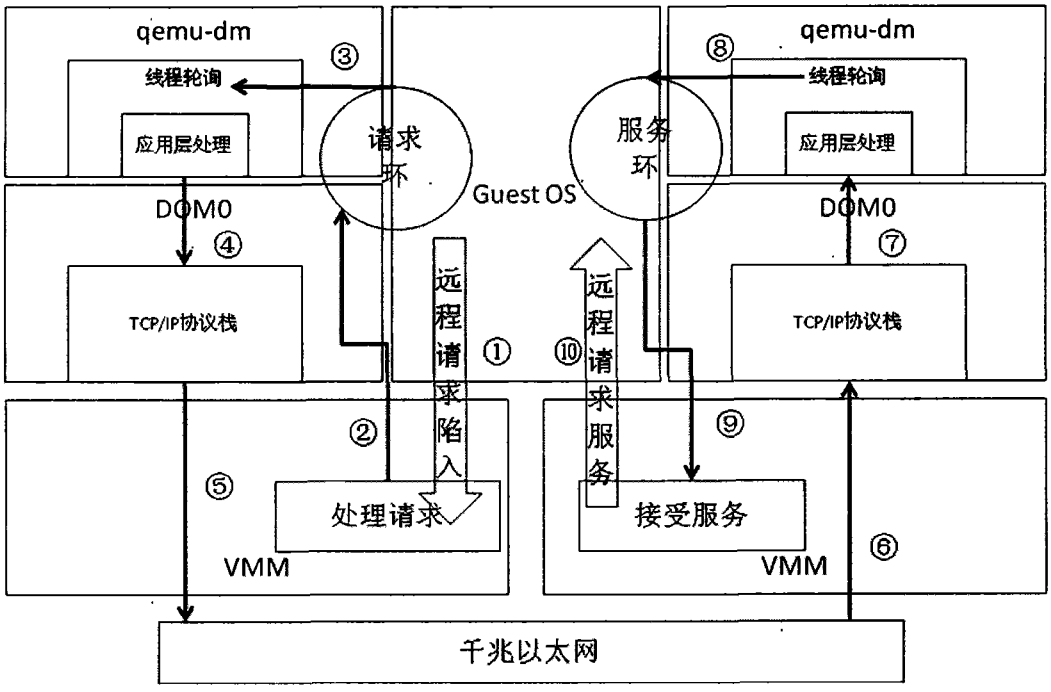

qemu-dm
③
线程轮询
应用层处理
请求环
DOM0
④
TCP/IP协议栈
Guest OS
远程请求陷入
①
②
⑤
处理请求
VMM
服务环
⑧
qemu-dm
线程轮询
应用层处理
DOM0
⑦
TCP/IP协议栈
⑩
远程请求服务
⑨
接受服务
VMM
⑥
千兆以太网

图 6