(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 112486632 B
(45)授权公告日 2024.06.18

(21)申请号 202011413906.8

(22)申请日 2020.12.07

(65)同一申请的已公布的文献号
申请公布号 CN 112486632 A

(43)申请公布日 2021.03.12

(73)专利权人 中国船舶集团有限公司第七一六研究所
地址 222001 江苏省连云港市圣湖路18号

(72)发明人 殷进勇 杨建 杨鸿斌 李轶 方新茂 路朗 徐振朋 曾玮妮 张鹏 徐国强

(74)专利代理机构 南京理工大学专利中心 32203
专利代理师 朱炳斐

(51)Int.Cl.
G06F 9/455(2006.01)

(56)对比文件
Spyros Chiotakis, .vFPGAmanager: A Hardware-Software Framework for Optimal FPGA Resources Exploitation in Network Function Virtualization.《IEEE Xplore》.2018,第48页左栏的1行-第50页,右栏第25行。附图2-4.

审查员 朱晓岗

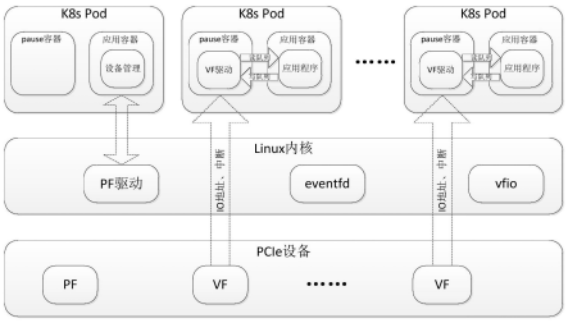
权利要求书1页 说明书3页 附图2页

(54)发明名称

一种面向k8s的用户态虚拟设备驱动框架

(57)摘要

本发明公开了一种面向k8s的用户态虚拟设备驱动框架,该框架采用SRIOV技术,将IO设备虚拟成多个共享硬件资源的虚拟设备,并将虚拟设备的IO地址、中断映射到pod的pause容器中,在pause容器运行虚拟设备的驱动程序。pod中的其他应用容器通过共享内存的方式访问pause容器中的驱动程序。该框架提供了虚拟设备管理、应用程序与驱动程序通信接口等内容,用户只需在框架下编写具体设备的驱动代码,降低了编写设备驱动程序的复杂性。该框架采用虚拟化和透传的方式,将设备驱动的大部分功能移植到pause容器内实现,可与应用程序一起交付,在保证设备效率的同时,提升了驱动程序的隔离性、可交付性和系统的可靠性。



CN 112486632 B

1.一种面向k8s的用户态虚拟设备驱动框架,其特征在于,所述驱动框架包括IO设备、Linux内核即内核空间,以及k8s pod即用户空间;

所述IO设备支持SRIOV,可以虚拟成多个虚拟设备VF,多个VF共享硬件资源;

所述Linux内核通过VFIO和eventfd将VF的IO地址、中断映射到k8s pod的pause容器中;VF设备驱动程序运行在pod 的pause容器内,pod中的其他应用容器通过共享内存的方式访问pause容器中的VF设备驱动程序,此外,其中一个pod 的应用容器中部署虚拟设备管理程序;

当pod请求一个虚拟设备时,虚拟设备管理程序分配虚拟设备的步骤如下:

步骤1,查询是否存在支持虚拟化的设备,如果存在,则转到步骤2;否则,转到步骤6;

步骤2,设备是否已经虚拟化,如果已经虚拟化,转到步骤4;否则,转到步骤3;

步骤3,根据设备虚属性和虚拟化方法,虚拟化出给定数量的虚拟设备;所述虚属性包括支持的虚拟设备数量;

步骤4,是否存在未分配的虚拟设备,如果存在,转到步骤5;否则,转到步骤6;

步骤5,通过VFIO和eventfd将虚拟设备的IO地址和中断映射到pod的pause容器中;

步骤6,设备分配失败,退出;

应用程序与VF设备驱动程序通过两个队列进行通信,实现应用程序读、写设备,具体实现方式包括:

提供了驱动接口和应用接口,其中驱动接口部署在pod的pause容器里,衔接VF设备驱动程序;应用接口部署在应用容器里,衔接应用程序;

同时,利用一个数组desc描述两个队列的状态。

2.根据权利要求1所述的面向k8s的用户态虚拟设备驱动框架,其特征在于,所述虚拟设备管理程序的功能包括虚拟设备分配,虚拟设备IO地址和中断映射。

3.根据权利要求1所述的面向k8s的用户态虚拟设备驱动框架,其特征在于,所述应用程序写设备,具体过程包括:

步骤1,应用程序将存有数据的buffer添加到写队列中;

步骤2,更改数组desc的状态,表示存在有效数据;

步骤3,通知VF设备驱动程序数据已更新;

步骤4,VF设备驱动程序查看数组desc的状态,读出有效数据。

一种面向k8s的用户态虚拟设备驱动框架

技术领域

[0001] 本发明属于用户态设备驱动框架领域,特别涉及一种面向k8s的用户态虚拟设备驱动框架。

背景技术

[0002] 在Linux系统中,传统的驱动程序运行在内核态,与内核其他部分共享同一地址空间。由于驱动程序本身的复杂性以及开发人员并不完全熟悉内核结构,其开发的驱动程序难免出现错误。一旦驱动程序产生错误,将影响所有使用该驱动程序的应用和其他引用该驱动程序的内核代码,甚至会造成整个操作系统的崩溃,影响整个系统的可靠性。

[0003] 为提升系统的可靠性,将驱动程序移到用户态是一个有效途径。驱动程序运行在用户态的主要优点包括:(1)驱动开发人员可以使用丰富的用户态应用程序开发工具、软件库,提高开发效率,降低开发难度;(2)用户态驱动程序错误不会影响整个系统,故障域减小,提高系统可靠性。

[0004] 目前开发用户态驱动程序主要采用的框架包括UIO和VFIO,UIO将驱动的很少一部分运行在内核空间,而在用户空间实现驱动的绝大多数功能,但UIO不能实现DMA功能,因此只适用于数据传输了较少的设备。VFIO是UIO的升级版,其利用IOMMU等硬件的支持,实现了设备地址空间的隔离,能够将设备地址空间、中断安全地映射到用户空间。基于VFIO框架,白子秋、胡怀湘等人提出了一种NVMe的用户态驱动程序设计方案,该论文发表在《计算机应用与软件》期刊上。

[0005] 作为容器编排软件的k8s已俨然成为云计算平台的标配,各地共有云平台均提供k8s服务;但目前的用户态设备驱动框架主要是针对传统应用模式提出的,不适用于容器化应用。

发明内容

[0006] 本发明的目的在于针对上述现有技术存在的问题,提供一种面向k8s的虚拟设备驱动框架,方便在k8s平台下开发用户态设备驱动程序。

[0007] 实现本发明目的的技术解决方案为:一种面向k8s的用户态虚拟设备驱动框架,所述驱动框架包括IO设备、Linux内核即内核空间,以及k8s pod即用户空间;

[0008] 所述IO设备支持SRIOV,可以虚拟成多个虚拟设备VF,多个VF共享硬件资源;

[0009] 所述Linux内核通过VFIO和eventfd将VF的IO地址、中断映射到k8s pod的pause容器中;VF设备驱动程序运行在pod的pause容器内,pod中的其他应用容器通过共享内存的方式访问pause容器中的VF设备驱动程序,此外,其中一个pod的应用容器中部署虚拟设备管理程序。

[0010] 进一步地,所述虚拟设备管理程序的功能包括虚拟设备分配,虚拟设备IO地址和中断映射。

[0011] 进一步地,当pod请求一个虚拟设备时,虚拟设备管理程序分配虚拟设备的步骤如

下：

[0012] 步骤1,查询是否存在支持虚拟化的设备,如果存在,则转到步骤2;否则,转到步骤6;

[0013] 步骤2,设备是否已经虚拟化,如果已经虚拟化,转到步骤4;否则,转到步骤3;

[0014] 步骤3,根据设备虚属性和虚拟化方法,虚拟化出给定数量的虚拟设备;所述虚属性包括支持的虚拟设备数量;

[0015] 步骤4,是否存在未分配的虚拟设备,如果存在,转到步骤5;否则,转到步骤6;

[0016] 步骤5,通过VFIO和eventfd将虚拟设备的IO地址和中断映射到pod的pause容器中;

[0017] 步骤6,设备分配失败,退出。

[0018] 进一步地,所述应用程序与VF设备驱动程序通过两个队列进行通信,实现应用程序读、写设备,具体实现方式包括:

[0019] 提供了驱动接口和应用接口,其中驱动接口部署在pod的pause容器里,衔接VF设备驱动程序;应用接口部署在应用容器里,衔接应用程序;

[0020] 同时,利用一个数组desc描述两个队列的状态。

[0021] 进一步地,所述应用程序写设备,具体过程包括:

[0022] 步骤1,应用程序将存有数据的buffer添加到写队列中;

[0023] 步骤2,更改数组desc的状态,表示存在有效数据;

[0024] 步骤3,通知VF设备驱动程序数据已更新;

[0025] 步骤4,VF设备驱动程序查看数组desc的状态,读出有效数据。

[0026] 本发明与现有技术相比,其显著优点为:1)用户只需在框架下编写具体设备的驱动代码,降低了编写设备驱动程序的复杂性;2)采用虚拟化和透传的方式,在保证设备效率的同时,提升了驱动程序的隔离性、可交付性和系统的可靠性。

[0027] 下面结合附图对本发明作进一步详细描述。

附图说明

[0028] 图1为一个实施例中面向k8s的用户态虚拟设备驱动框架组成图。

[0029] 图2为一个实施例中虚拟设备分配流程图。

[0030] 图3为一个实施例中应用程序与设备驱动程序通信示意图。

具体实施方式

[0031] 为了使本申请的目的、技术方案及优点更加清楚明白,以下结合附图及实施例,对本申请进行进一步详细说明。应当理解,此处描述的具体实施例仅仅用以解释本申请,并不用于限定本申请。

[0032] 在一个实施例中,结合图1,提供了一种面向k8s的用户态虚拟设备驱动框架,所述驱动框架包括IO设备、Linux内核即内核空间,以及k8s pod即用户空间;

[0033] 所述IO设备支持SRIOV,可以虚拟成多个虚拟设备VF,多个VF共享硬件资源;

[0034] 所述Linux内核通过VFIO和eventfd将VF的IO地址、中断映射到k8s pod的pause容器中;VF设备驱动程序运行在pod的pause容器内,pod中的其他应用容器通过共享内存的方

式访问pause容器中的VF设备驱动程序,此外,其中一个pod的应用容器中部署虚拟设备管理程序,该程序的功能包括虚拟设备分配,虚拟设备IO地址和中断映射。

[0035] 该框架提供了虚拟设备管理程序、应用程序与驱动程序通信接口等内容,用户只需在框架下编写具体设备的驱动代码,降低了编写设备驱动程序的复杂性。该框架采用虚拟化和透传的方式,将设备驱动的大部分功能移植到pause容器内实现,可与应用程序一起交付,在保证设备效率的同时,提升了驱动程序的隔离性、可交付性和系统的可靠性。

[0036] 进一步地,在其中一个实施例中,结合图2,当pod请求一个虚拟设备时,虚拟设备管理程序分配虚拟设备的步骤如下:

[0037] 步骤1,查询是否存在支持虚拟化的设备,如果存在,则转到步骤2;否则,转到步骤6;

[0038] 步骤2,设备是否已经虚拟化,如果已经虚拟化,转到步骤4;否则,转到步骤3;

[0039] 步骤3,根据设备虚属性和虚拟化方法,虚拟化出给定数量的虚拟设备;所述虚属性包括支持的虚拟设备数量;

[0040] 步骤4,是否存在未分配的虚拟设备,如果存在,转到步骤5;否则,转到步骤6;

[0041] 步骤5,通过VFIO和eventfd将虚拟设备的IO地址和中断映射到pod的pause容器中;

[0042] 步骤6,设备分配失败,退出。

[0043] 进一步地,在其中一个实施例中,结合图3,所述应用程序与VF设备驱动程序通过两个队列进行通信,实现应用程序读、写设备,具体实现方式包括:

[0044] 提供了驱动接口和应用接口,其中驱动接口部署在pod的pause容器里,衔接VF设备驱动程序;应用接口部署在应用容器里,衔接应用程序;

[0045] 同时,利用一个数组desc描述两个队列的状态。

[0046] 进一步地,在其中一个实施例中,所述应用程序写设备,具体过程包括:

[0047] 步骤1,应用程序将存有数据的buffer添加到写队列中;

[0048] 步骤2,更改数组desc的状态,表示存在有效数据;

[0049] 步骤3,通知VF设备驱动程序数据已更新;

[0050] 步骤4,VF设备驱动程序查看数组desc的状态,读出有效数据。

[0051] 本发明基于VFIO框架,针对k8s的Pod架构提出了一种用户态设备驱动框架,支撑k8s架构下的用户态设备驱动程序开发。该框架将设备驱动程序封装到容器内,与应用一起交付、部署、运行,使应用之间的隔离性更强、降低不同应用之间的干扰程度,提高了系统的可靠性。

[0052] 以上显示和描述了本发明的基本原理、主要特征及优点。本行业的技术人员应该了解,本发明不受上述实施例的限制,上述实施例和说明书中描述的只是说明本发明的原理,在不脱离本发明精神和范围的前提下,本发明还会有各种变化和改进,这些变化和改进都落入要求保护的本发明范围内。本发明要求保护范围由所附的权利要求书及其等效物界定。

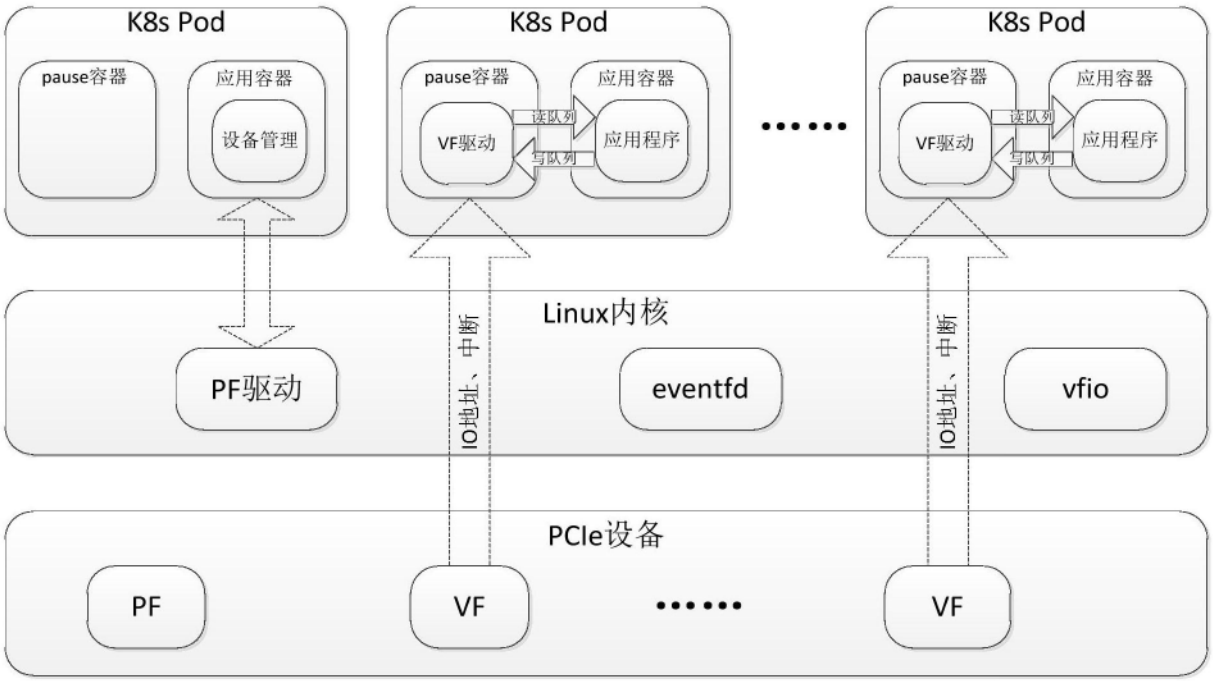

K8s Pod
pause容器
应用容器
设备管理
K8s Pod
pause容器
应用容器
VF驱动
读队列
写队列
应用程序
K8s Pod
pause容器
应用容器
VF驱动
读队列
写队列
应用程序
Linux内核
PF驱动
IO地址、中断
eventfd
IO地址、中断
vfio
PCIe设备
PF
VF
VF

图1

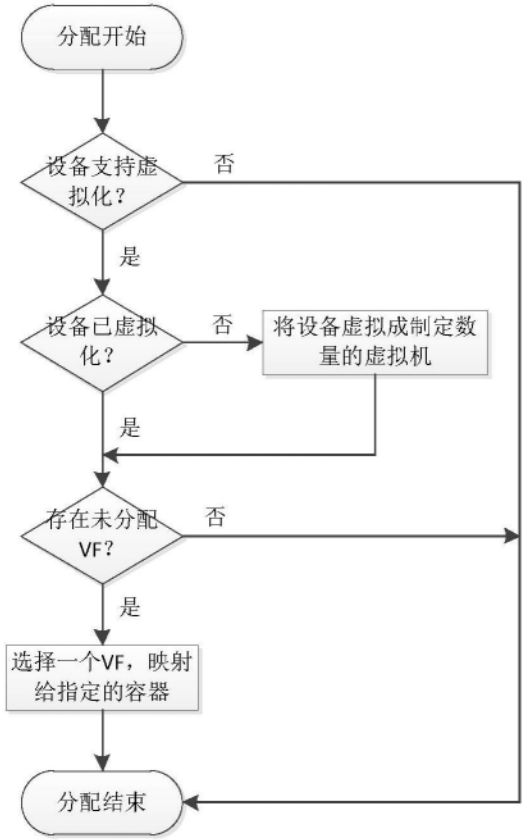

分配开始
设备支持虚拟化？
否
是
设备已虚拟化？
否
将设备虚拟成制定数量的虚拟机
是
存在未分配VF？
否
是
选择一个VF，映射给指定的容器
分配结束

图2

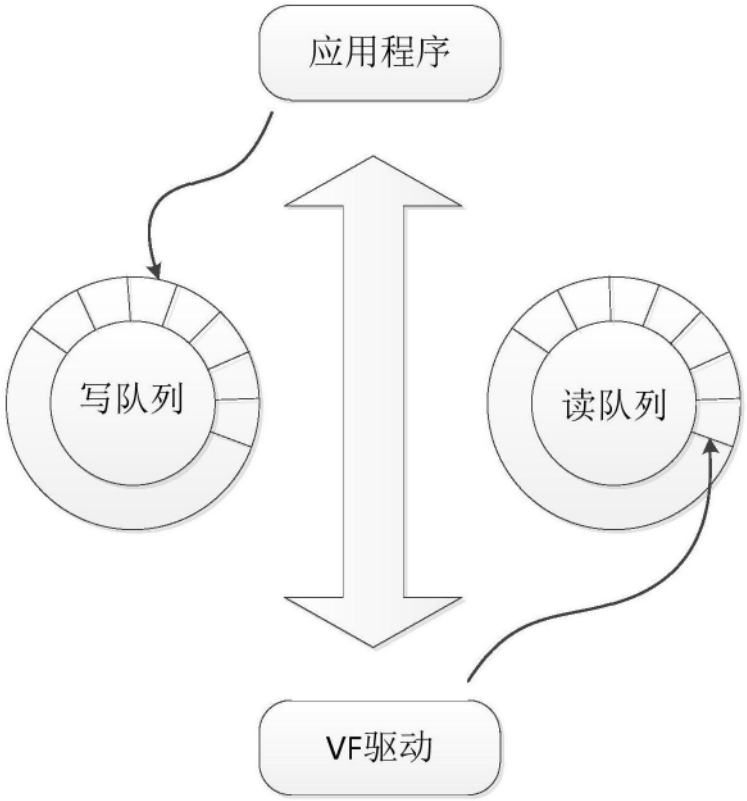


图3