Virtio原理简介

2018-08-07 | 2.虚拟化， virtio | 5566 | 3
3k | 3 分钟

前言

> 实现IO虚拟化主要有三种方式：全虚拟化、半虚拟化和透传。全虚拟化Guest OS不会感知到自己是虚拟机，也无需修改Guest OS，但是它的效率比较低。半虚拟化Guest OS知道自己是虚拟机，通过Frontend/Backend驱动模拟实现IO虚拟化。透传就是直接分配物理设备给VM用。Virtio是一种半虚拟化的设备抽象接口规范，在Qemu和KVM中得到了广泛使用，本文将简单介绍Virtio的基本原理。

代码版本

- Qemu
  - qemu-2.10.1
- Kernel
  - kernel-3.10.0

Virtio SPEC

- Virtual I/O Device (VIRTIO) Version 1.0
  - http://docs.oasis-open.org/virtio/virtio/v1.0/cs04/virtio-v1.0-cs04.html
- Virtio PCI Card Specification Version 0.9.5
  - http://ozlabs.org/~rusty/virtio-spec/virtio-0.9.5.pdf

正文

Virtio在Guest中实现了前端驱动，在Host中实现了后端驱动，前后端之间通过Virtqueue(虚拟队列)交换数据，Host中会使用后端驱动程序模拟一个PCI设备，因此也称前端驱动为Driver，后端驱动为Device。Guest在Host OS上表示为一个Qemu的进程，Guest OS的

pa实际上也属于Host OS的地址空间，因此Virtio采用的Virtqueue的方式来避免了Guest和Host间数据的复制。下面先介绍一下Virtio，更为详细的描述请阅读前言中提供的Virtio SPEC文档，那里是最准确而详细的描述，在此仅简单介绍一下后文需要用到的概念。

Virtio规范简介

Basic Facilities of a Virtio Device

每个Virtio设备包括以下部分

- Device status field **设备状态字段**
- Feature bits **特征位**
- Device Configuration space **设备配置空间**
- One or more virtqueues **一个或多个virtqueues**

virtqueues

在virtio设备上进行批量数据传输的机制被称为virtqueue，每个设备可以拥有零个或多个virtqueue，每个virtqueue由三部分组成：

- Descriptor Table
- Available Ring
- Used Ring

每部分在客户机内存中是物理连续的，并且有不同的对齐要求，virtqueue的每个部分的内存对齐和大小要求如下表，其中Queue Size对应virtqueue中的最大buffer数，始终为2的n次幂，以特定于总线的方式指定：

Virtqueue Part	Alignment	Size
Descriptor Table	16	16*(Queue Size)
Available Ring	2	6 + 2*(Queue Size)
Used Ring	4	6 + 8*(Queue Size)

Descriptor Table指的是Driver用于Device的缓冲区，由Queue Size个Descriptor组成。Descriptor中存有GPA的字段addr，长度字段len，可以链接next Descriptor的next指针等(形成描述符链)。如果协商了VIRTIO_F_INDIRECT_DESC feature则可以使用Indirect Descriptors来

增加ring的容量。

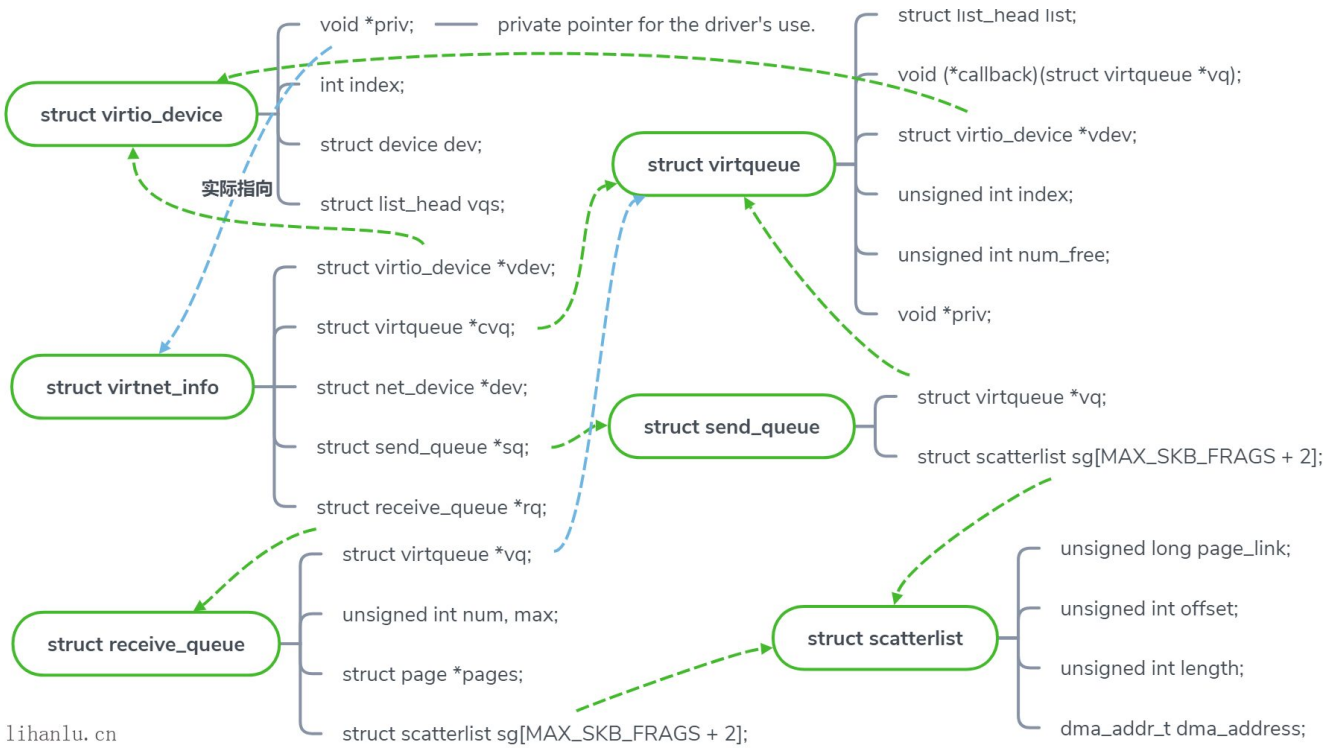
Available Ring中的每个条目是一个是描述符链的头部，它仅由Driver写入并由Device读取，Device获取Descriptor后，Descriptor对应的缓冲区可能是可读的也可能是可写的，可读的用于Driver发送数据，可写的用于接收数据。

Used Ring的介绍直接贴SPEC文档中的描述上来比翻译过来更容易理解。The used ring is where the device returns buffers once it is done with them: it is only written to by the device,and read by the driver.简单来说Used Ring的作用就是Device使用完Descriptor之后，将Descriptor放入这里，通知Driver回收。

相关数据结构

> 一些数据结构只列举了部分关键信息

如图所示，**virtnet_info**作为virtio网络设备的私有数据将**virtio_device**和**net_device**链接在一起。对于virtio网络设备来说，它至少有两个**virtqueue**(如果协商了**VIRTIO_NET_F_MQ**则可以创建多个队列,详见VirtIO SPEC中5.1.2)，一个用于TX(**send_queue**)，另一个用于RX(**receive_queue**)，TX和RX队列中都包含了**virtqueue**和**scatterlist[]**。**virtio_device**、**virtnet_info**、**receive_queue**、**send_queue**、**virtqueue**给定任一结构体均可得到其它结构体信息。



可以把**virtqueue**理解为一个接口类，而**vring_virtqueue**作为这个接口的一个实现，**vring_virtqueue**通过成员vq可以与上述其它struct建立联系。virtio的环形缓冲区机制是由vring

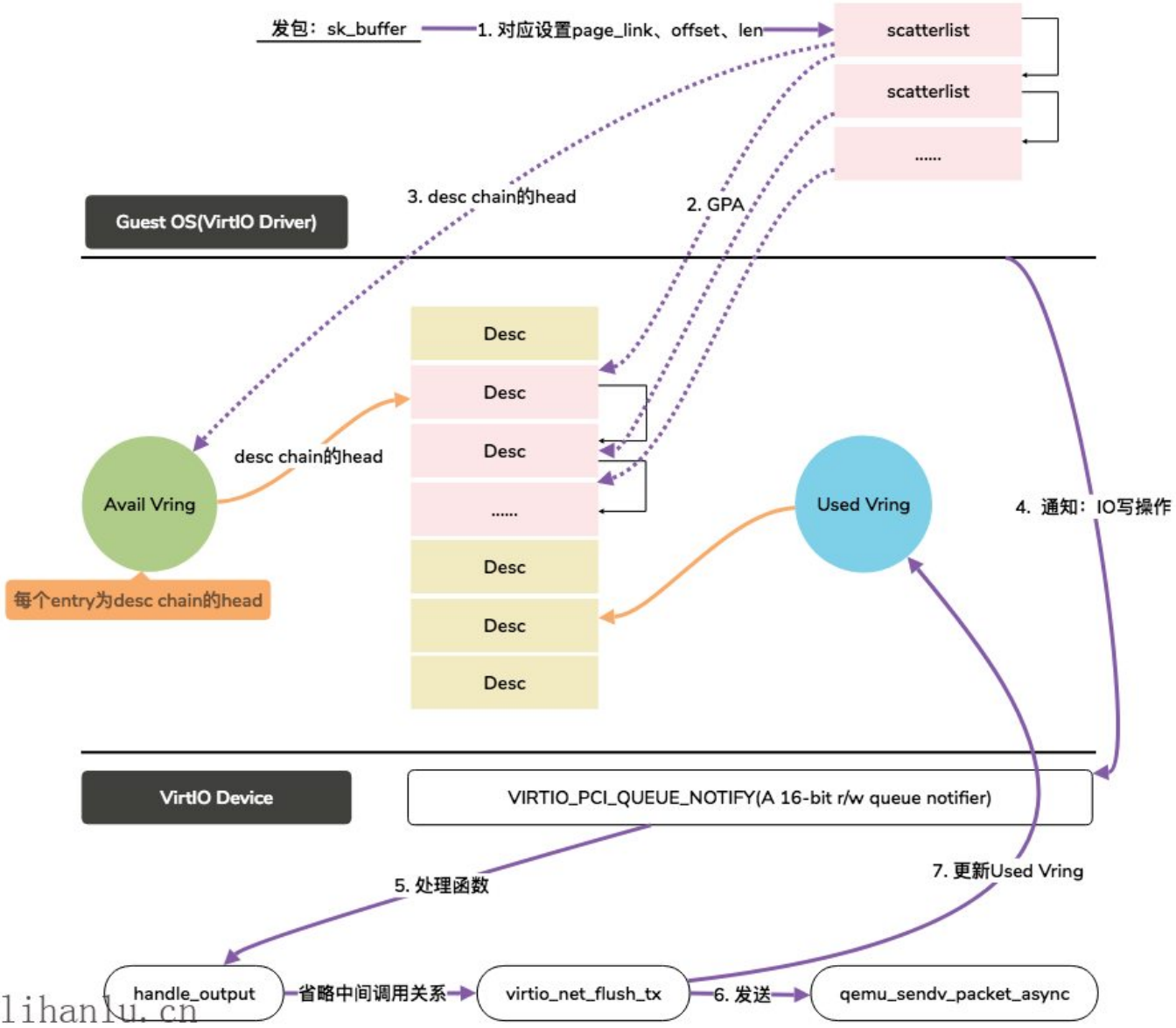
来承载的，vring由三部分组成：Descriptor表(vring_desc)，Available ring(vring_avail)和Used ring(vring_used)。

Vring机制简介

在virtio设备上进行批量数据传输的机制被称为**virtqueue**。每个设备可以拥有零个或多个**virtqueue**，当Driver想要向设备发送数据时，它会填充Descriptor Table中的一项（或将几项链接在一起），并将描述符索引写入**Available Ring**中，然后它通知Device，当Device完成后，它将描述符索引写入**Used Ring**中并发送中断。(详见SPEC 2.4)

下图Virtio网络设备发包过程为例讲解上述机制，Driver将sk_buffer填充进scatterlist table中(只是设置地址没有数据搬移)，然后通过计算得到GPA并将GPA写入Descriptor Table中，同时将Desc chain的head记录到**Available Ring**中，然后通过PIO的方式通知Device，Device发包并更新**Used Ring**。

> 后续会有一篇文章专门分析发包过程



联系我

你可以直接在下方留言，也可以E-Mail联系我。

打赏

virtio

68%

昵称 邮箱 网址(http://)

Just go go

提交

3 评论

charles Chrome 78.0.3904.108 Windows 10.0
6 天前 回复

你好，有个问题请教。 Desc里面填充的是GPA， 后端Qemu进行发包的时候，如何将GPA转换成HPA或者HVA？ 难道是利用EPT页表？

Anonymous Chrome 80.0.3987.132 Windows 10.0
2020-03-19 回复

请问一下,qemu_sendv_packet_async最后是跟host linux里面的net driver交互吗,有没有具体的过程?

Anonymous Chrome 79.0.3945.88 Windows 10.0
2020-01-03 回复

Vring机制简介分析发包过程有过程链接么.

lauren Chrome 79.0.3945.88 Windows 10.0
2020-01-08 回复

@Anonymous , http://lihanlu.cn/virtio-net-xmit/

68%

京ICP备19012335号-1