OpenAnolis
龙 蜥 社 区

设备透传虚拟机的快速启动优化

浪潮电子信息产业股份有限公司

/ 目录 /

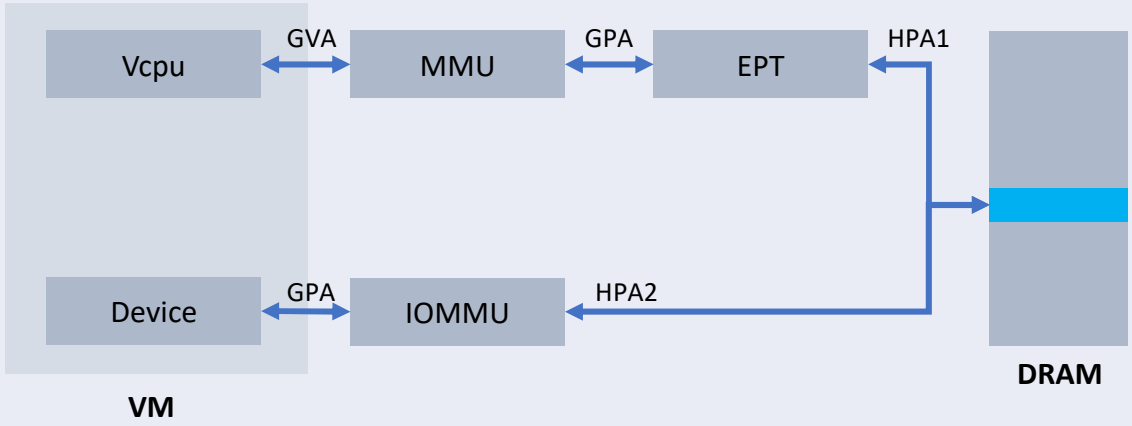
Contents

OpenAnolis 龙蜥社区

虚拟机设备透传时存在的问题

将网卡/GPU等设备通过vfio透传到虚拟机，并且为虚拟机并分配了比较大的内存时，虚拟机的启动时间会明显变慢，可能由十几秒延长到数分钟，在某些场景会严重影响用户使用体验。

设备透传实现



网卡/GPU等设备通过vfio透传到虚拟机之后：

- 设备驱动程序访问的虚拟地址经过过EPT映射为HPA1
- 设备DMA访问的内存经过IOMMU映射为HPA2
- HPA1和HPA2必须相等，vfio通过VFIO_IOMMU_MAP_DMA命令实现

Free page reporting机制

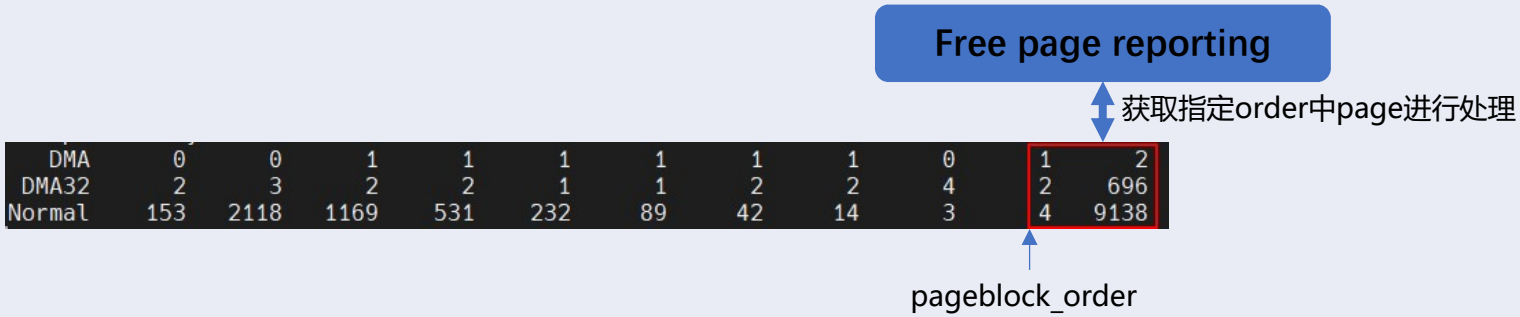
mm / virtio: Provide support for free page reporting | collapse
[v17,0/9] mm / virtio: Provide support for free page reporting
[v17,1/9] mm: Adjust shuffle code to allow for future coalescing
[v17,2/9] mm: Use zone and order instead of free area in free_list manipulators
[v17,3/9] mm: Add function __putback_isolated_page
[v17,4/9] mm: Introduce Reported pages
[v17,5/9] virtio-balloon: Pull page poisoning config out of free page hinting
[v17,6/9] virtio-balloon: Add support for providing free page reports to host
[v17,7/9] mm/page_reporting: Rotate reported pages to the tail of the list
[v17,8/9] mm/page_reporting: Add budget limit on how many pages can be reported per pass
[v17,9/9] mm/page_reporting: Add free page reporting documentation

kernel 5.7合入free page reporting机制，并且支持virtio-balloon注册

virtio-balloon: add support for free page reporting | collapse
[v21,QEMU,0/5] virtio-balloon: add support for free page reporting
[v21,QEMU,1/5] linux-headers: Update to allow renaming of free_page_report_cmd_id
[v21,QEMU,2/5] linux-headers: update to contain virito-balloon free page reporting
[v21,QEMU,3/5] virtio-balloon: Replace free page hinting references to 'report' with 'hint'
[v21,QEMU,4/5] virtio-balloon: Implement support for page poison tracking feature
[v21,QEMU,5/5] virtio-balloon: Provide an interface for free page reporting

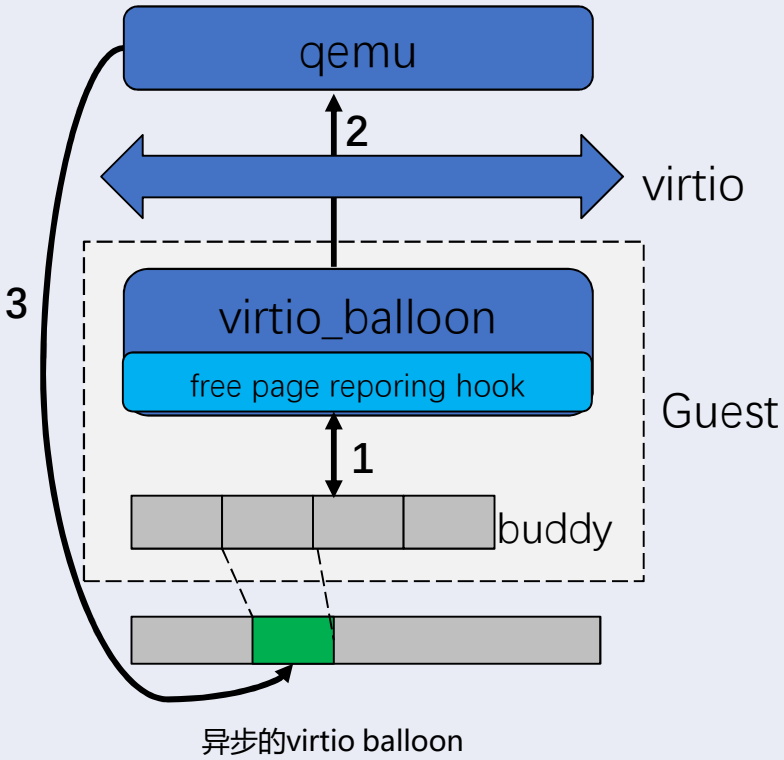
qemu端的5.1.0支持

Free page reporting机制



- Free page reporting机制提供了回调函数注册，每隔2秒遍历每个zone中的空闲页并调用回调函数进行处理
- 只处理pageblock_order以上的buddy内存
- 每个zone水线至少要比low水线多32*2M（X86）

Free page reporting机制应用



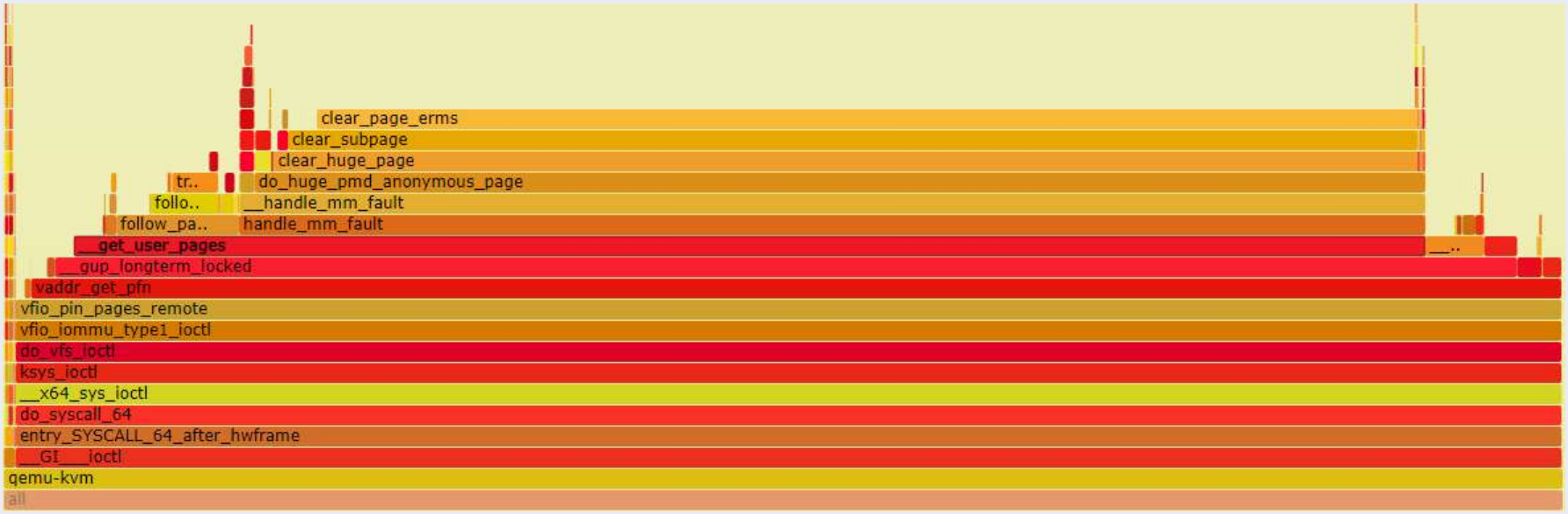
异步的virtio balloon

- Virtio_balloon注册free page reporting回调函数
- 每隔2秒周期性的调用回调函数将guest中的空闲页通过virtio发送给qemu进程
- Qemu进程调用madvise释放空闲页对应的内存

/ 目录 /

Contents

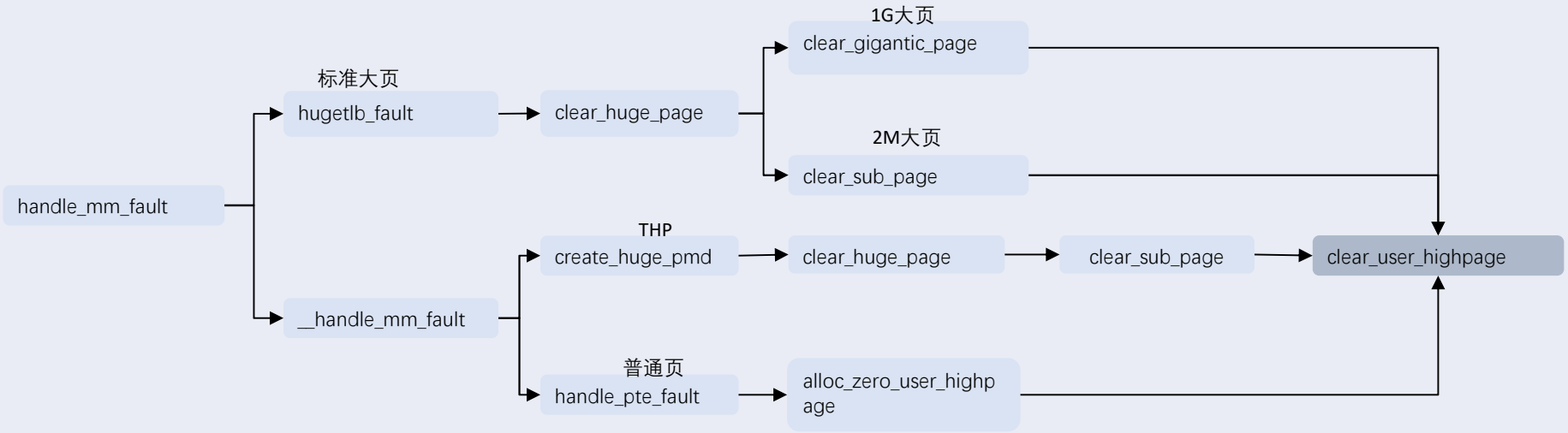
OpenAnolis 龙蜥社区



qemu-kvm通过ioctl(container, VFIO_IOMMU_MAP_DMA, &dma_map)进入内核空间

性能瓶颈点在clear_subpage

内存页清零




```
static inline void clear_user_highpage(struct page *page, unsigned long vaddr)
{
        void *addr = kmap_atomic(page);
        clear_user_page(addr, vaddr, page);
        kunmap_atomic(addr);
}
```

clear_user_page为架构相关代码

基于free page reporting实现内存页预清零

1、通过free page reporting接口注册hook函数

2、hook函数周期性的将buddy中2M/4M的空闲页内存进行清零并设置清零flag

3、应用程序申请的内存触发缺页需要对内存页进行清零时，首先判断该页的是否设置了清零flag，如果已设置则跳过清零操作。

- 由于内存页清零属于耗时且不紧急的操作，所以只有在CPU空闲时才执行

内存页预清零使用限制

- **不能作用于标准大页**

- Free page reporting机制原理是监控buddy的空闲内存，而标准大页的空闲内存不被buddy管理。
  除非page在加入标准大页内存池之前已经做过内存预清零操作。

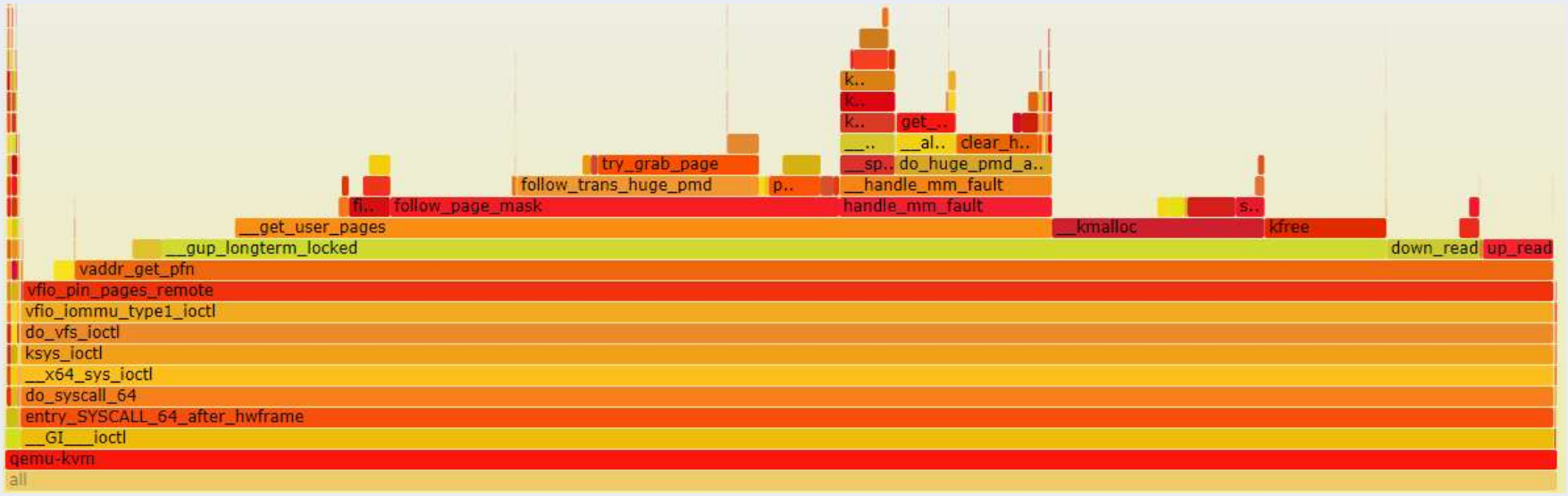
- **不要在虚拟机里开启该功能**

- 该功能内存清零时将会导致qemu申请的虚拟内存产生page fault，从而真实占用物理内存。
- 该功能开启将导致vritio-balloon的free page reporing特性不能生效（前提是qemu也支持该特性）。

/ 目录 /

Contents

OpenAnolis 龙蜥社区



内存页预清零之后，get_user_pages中执行时间比较长的函数follow_page_mask、handle_mm_fault、find_extern_vma

代码执行流程

```
vfio_pin_map_dma(...)
{
        while(size)
        {
                npage = vfio_pin_pages_remote(dma,start_vaddr,size,&pfn,limit)
                vfio_iommu_map(iommu,start_vaddr,pfn,npage,true)
        }将物理地址连续的内存执行iommu map
}
vfio_pin_pages_remote(...)
{
        for (...)
        {
                pin_user_pages_remote(NULL, mm, vaddr, 1, flags | FOLL_LONGTERM, page, NULL, NULL);
                pfn = page_to_pfn(page[0]);
                if(pfn != *pfn_base + pinned)
                        break;
        }每次只pin一个page，然后判断和上一个page是否物理连续
}
```

代码执行流程

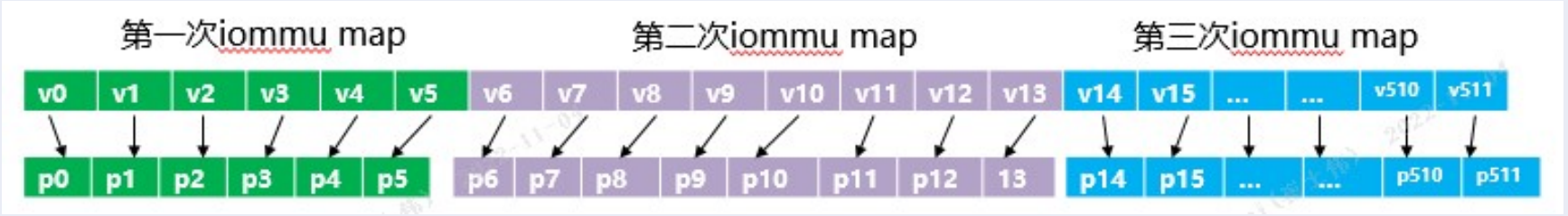
```
get_user_pages(...)
{
        while(npages)
        {
                if(!vma || start >= vma->vm_end)//相邻的两个虚拟内存页是否属于同一个vma
                {
                        vma = find_extern_vma(...)
                }
                page = follow_page_mask(vma,start,...);
                if(!page)
                        faultin_page(...)
        }
}
```

虚拟机的物理内存是qemu mmap的连续虚拟内存，属于同一个vma，所以每次pin多个page可以跳过频繁执行的find_extern_vma

可优化方案

- 修改gup代码， get_user_pages函数可以pin物理连续的多个page
- 修改vfio代码，每次pin多个page，然后分批将这些page中物理连续的部分执行iommu map

内核社区优化方案



- 不修改gup代码， pin_user_pages_remote每次获取512(4096/8)个page。
- 分批将这512个page中物理连续的部分进行iommu map。

```
4d83de6da265 vfio/type1: Batch page pinning
4b6c33b32296 vfio/type1: Prepare for batched pinning with struct vfio_batch
be16c1fd99f4 vfio/type1: Change success value of vaddr_get_pfn()
```

5.12内核合入

```
Pinning one 4K page at a time is inefficient, so do it in batches of 512
instead.  This is just an optimization with no functional change
intended, and in particular the driver still calls iommu_map() with the
largest physically contiguous range possible.

Add two fields in vfio_batch to remember where to start between calls to
vfio_pin_pages_remote(), and use vfio_batch_unpin() to handle remaining
pages in the batch in case of error.

qemu pins pages for guests around 8% faster on my test system, a
two-node Broadwell server with 128G memory per node.  The qemu process
was bound to one node with its allocations constrained there as well.
```

/ 目录 /

Contents

OpenAnolis 龙蜥社区

系统配置 开启特性	开启THP（默认）	2M大页	1G大页
基准值	79.5	37	36.8
大块内存pin	60	18	18
内存页预清零+大块内存pin	11.7	18.3	18.9

- 虚拟机分配512G内存。
- 时间计算方法：time vrish start testvm，时间为从qemu启动到出现grub界面为止。
- 内存预清零在最理想情况下，即当前系统上所有free page已经清零完毕。

OpenAnolis
龙 蜥 社 区

THANKS

SUBHEADING

龙蜥社区