

虚拟化底层之道--Qemu-KVM学习

—徐安 (QQ&微信: 2484769112)

目录

KVM技术学习

OpenStack, Docker的应对之策

广告--汉柏OPV-Suite

什么是虚拟化，什么是KVM

❖ **网络上最靠谱的定义：** 虚拟化是将计算机物理资源如服务器、网络、内存及存储等予以抽象、转换后呈现出来，使用户可以比原本的组态更好的方式来应用这些资源。这些资源的新虚拟部份是不受现有资源的架设方式，地域或物理组态所限制。

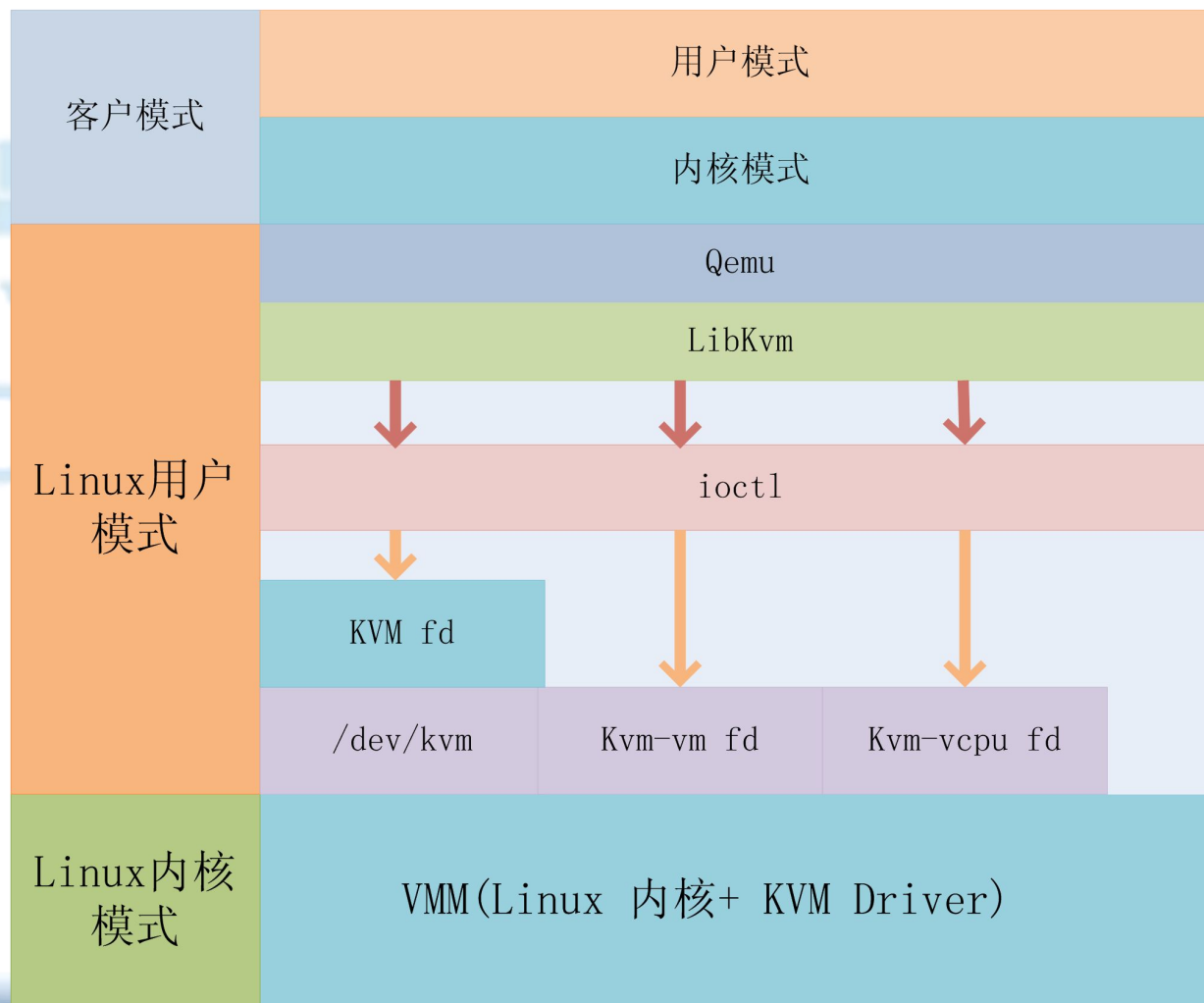
❖ **Virtualization = Abstract + Pool + Automate**

❖ **特点：**

- 脱离物理设备
- 可弹性控制
- 按需使用
- 可度量

什么是KVM

- ❖ **KVM (全称是 Kernel-based Virtual Machine) 是开源的 Linux 下 x86 硬件平台上的全功能虚拟化解决方案，自 Linux 2.6.20 之后集成在Linux的各个主要发行版本中。**

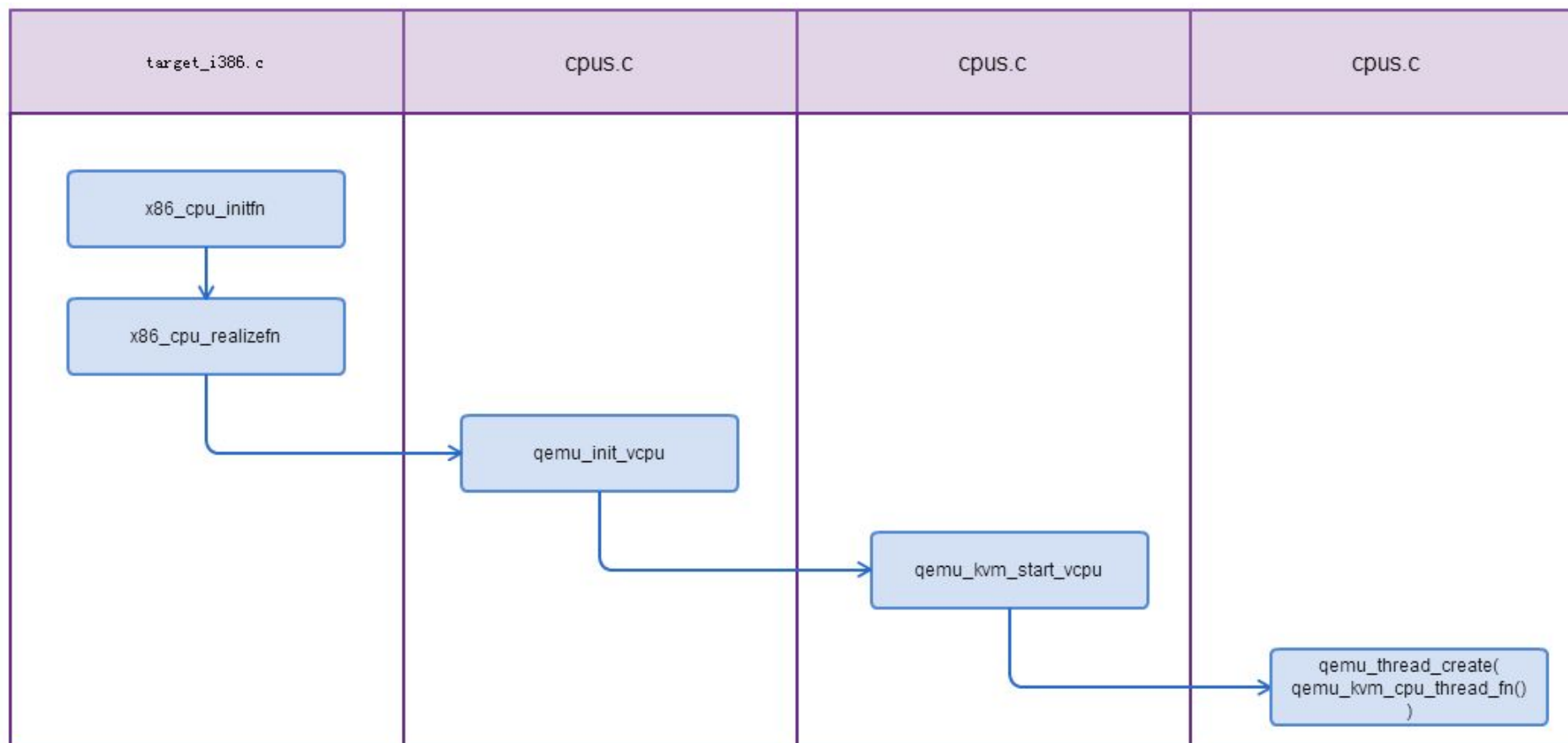


Qemu-kvm线程模型

- ❖ 主（父）线程。主线程执行循环，主要做三件事情
 - 执行**select**操作；执行定时器回调函数；执行下半部（**BH**）回调函数 -- **FT**实现在这里
- ❖ 执行客户机代码的线程
 - 如果有多个**vcpu**，就意味着存在多个线程。每个**vcpu**是一个单独的**thread**；
 - **vcpu thread**在**VM_ENTER**以后执行客户机代码，在**VM_EXIT**以后根据退出原因执行**PIO**或者**MMIO**。
- ❖ 异步io文件操作线程，**aio_context_thread_pool**
 - 提交**i/o**操作请求到队列中，该线程从队列取请求，并进行处理。
 - 如果**aio**类型是**threads pool**，启动一个**thread**去执行类似**bdrv_aio_readv/raw_aio_writev**操作
 - 如果**aio**类型是**native io**，同步执行完**io_submit**后立马返回，启动专门的**work**线程去执行**io_getevents** -- 异步得到返回结果
- ❖ **VNC**线程

CPU虚拟化-1

❖ VCPU线程被Main线程创建



CPU虚拟化-2

❖ VCPU什么时候被调度？

不需要被调度，一直都在运行--

主要在右边的while中运行。

调用KVM的KVM_RUN就是把

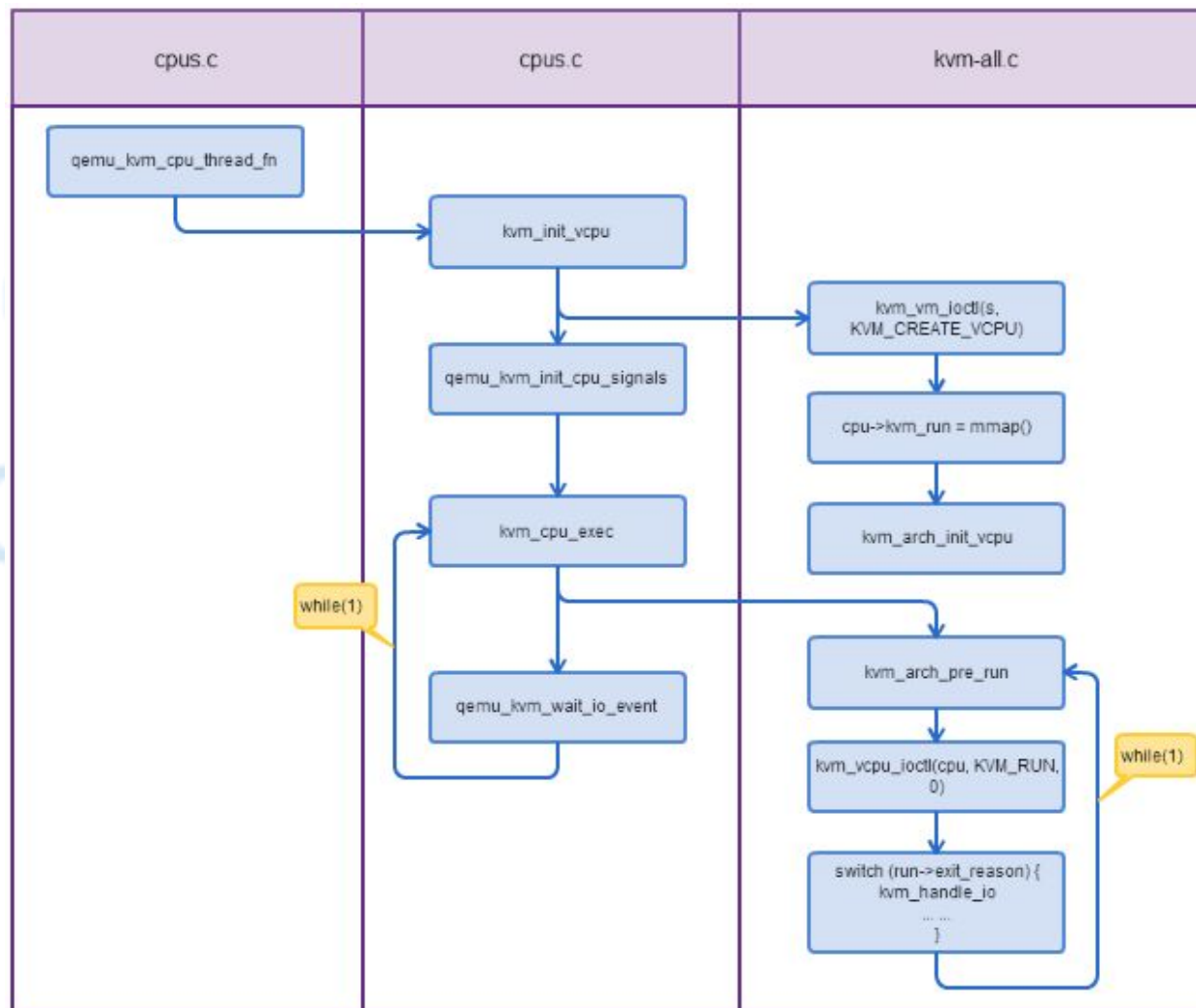
CPU交给虚拟机及KVM。

虚拟机有IO请求需求QEMU-KVM

处理时，会退出，QEMU-KVM

根据run->exit_reason的原因

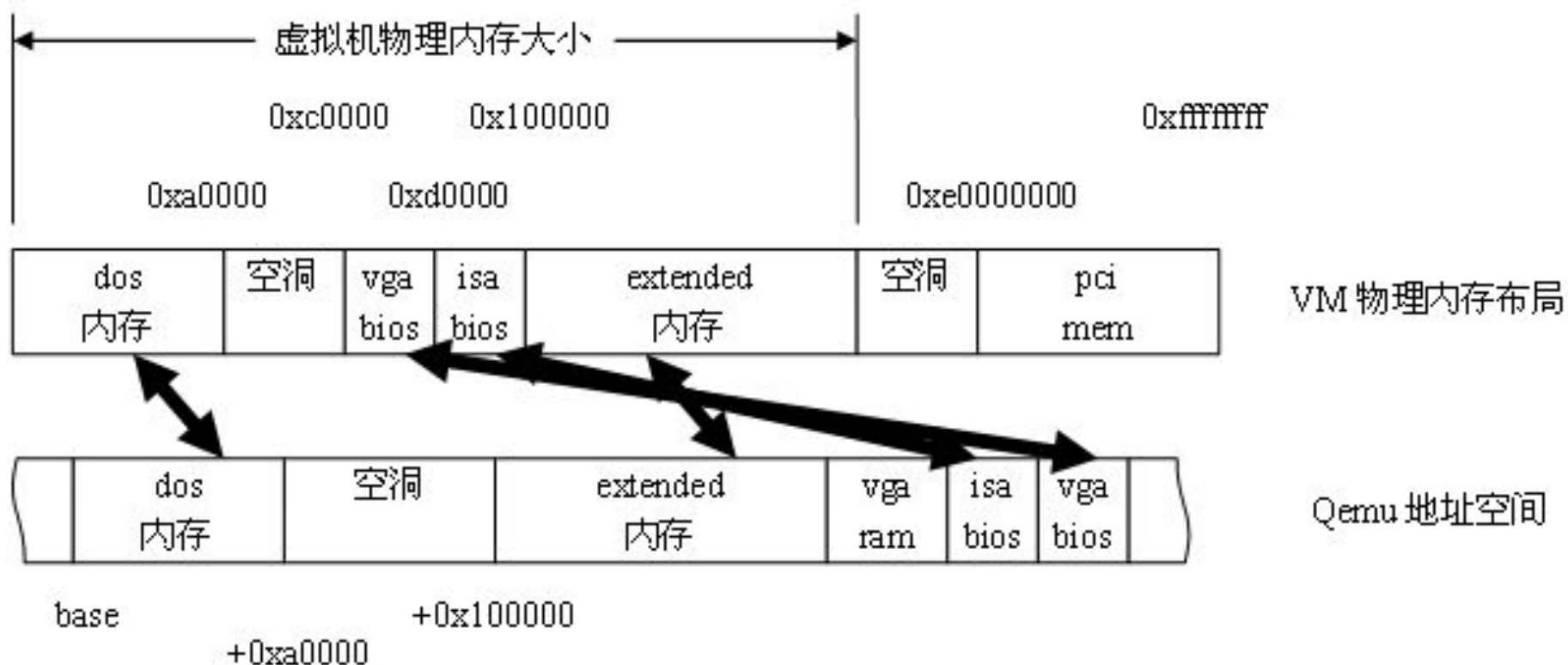
进行下一步的动作。



内存虚拟化-1

❖ 虚拟机地址访问过程

GVA (VM页表) --> GPA (kvm_mem_slot结构体) --> HVA (Host页表) --> HPA



内存虚拟化-2

❖ 有两个方法加快以上翻译的过程

- 影子页表：
 - **VMM**可以使用该**HPA**来构建影子页表，即建立**GVA**到**HPA**的映射。
- 使用**EPT**：
 - 则**VMM**在“**ept violation vm exit**”发生时利用以上素材建立**GPA**与**HPA**的映射关系--**EPT**表。

❖ 我们到底用的是哪个呢？

- `$cat /proc/cpuinfo | grep ept`检查硬件是否支持**ept**机制。
- 如果支持那么**kvm**会自动的利用**EPT**。

影子页表

- ❖ CR3指向影子页表；任何修改页表的动作都被捕获，由KVM来修改影子页表

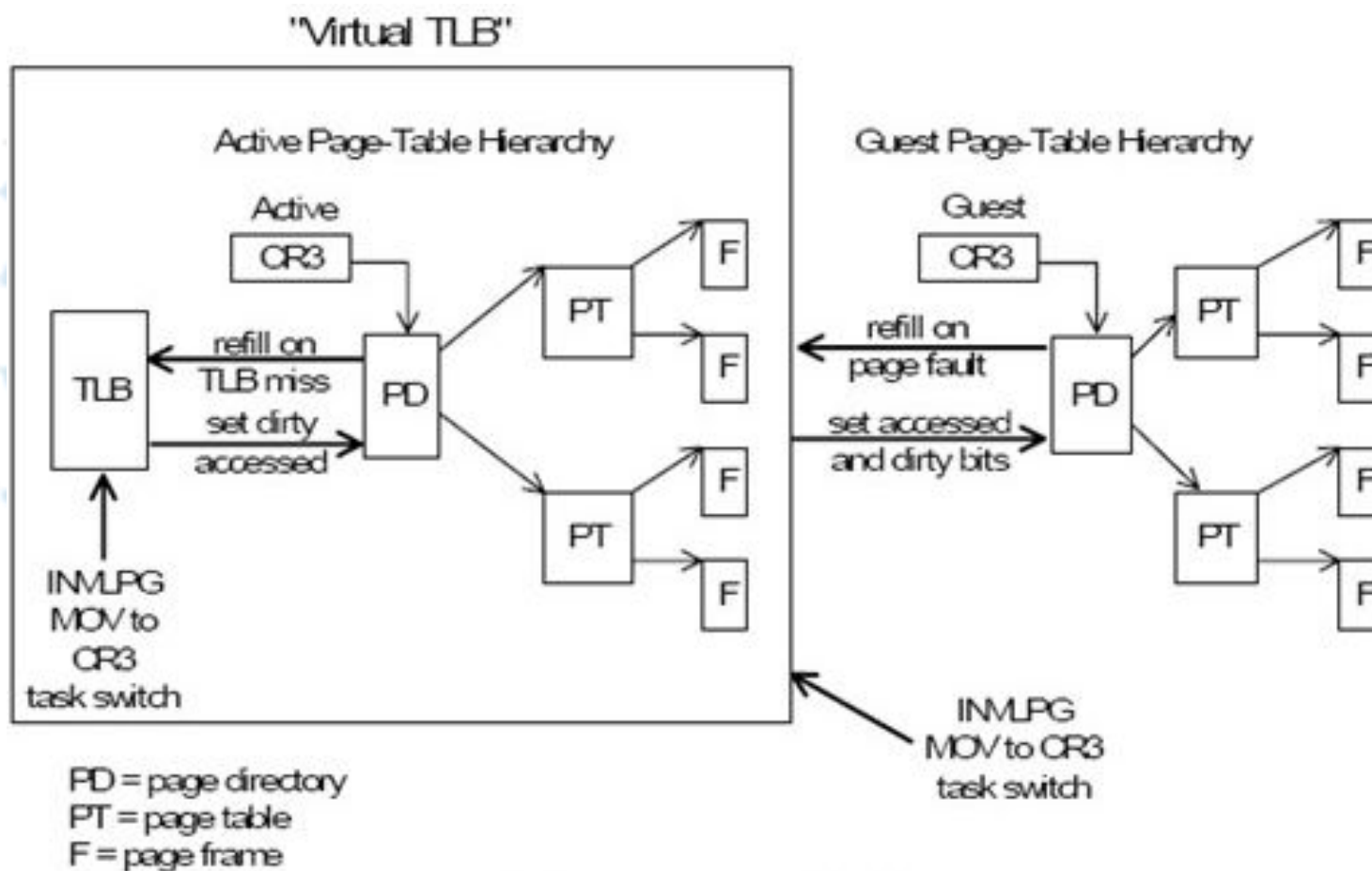
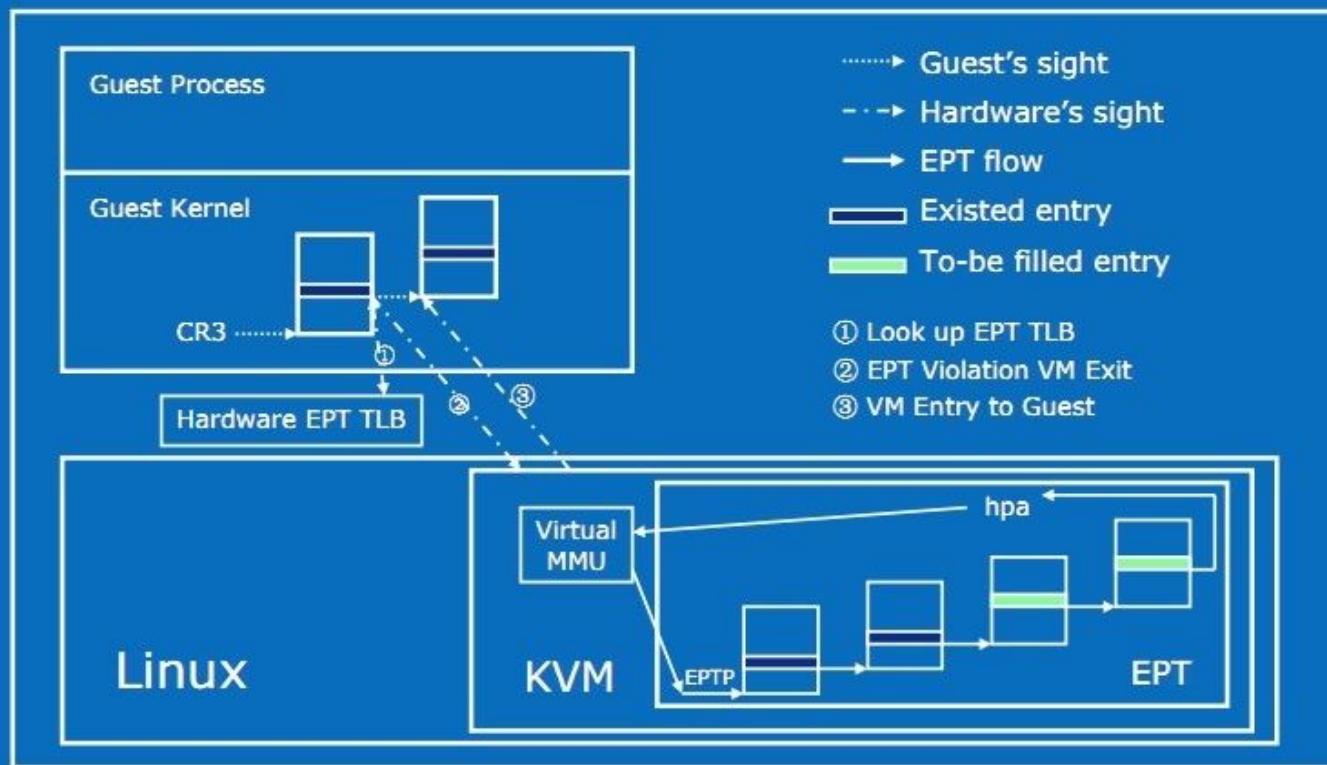


图 3.1 VTLB 工作机制

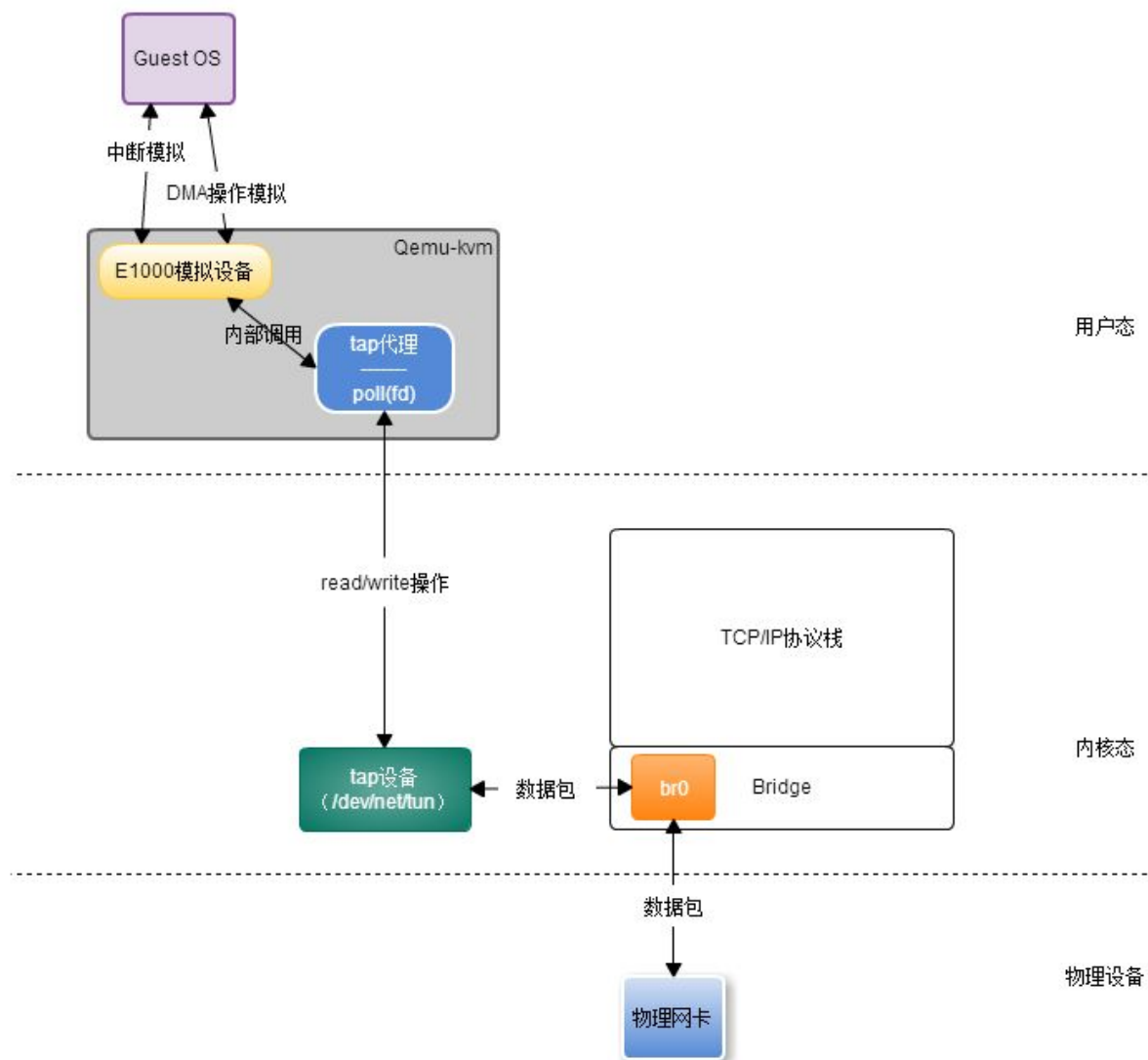
EPT

- ❖ 利用硬件自动翻译 -- GPA到HPA的映射
- ❖ TLB加快速度

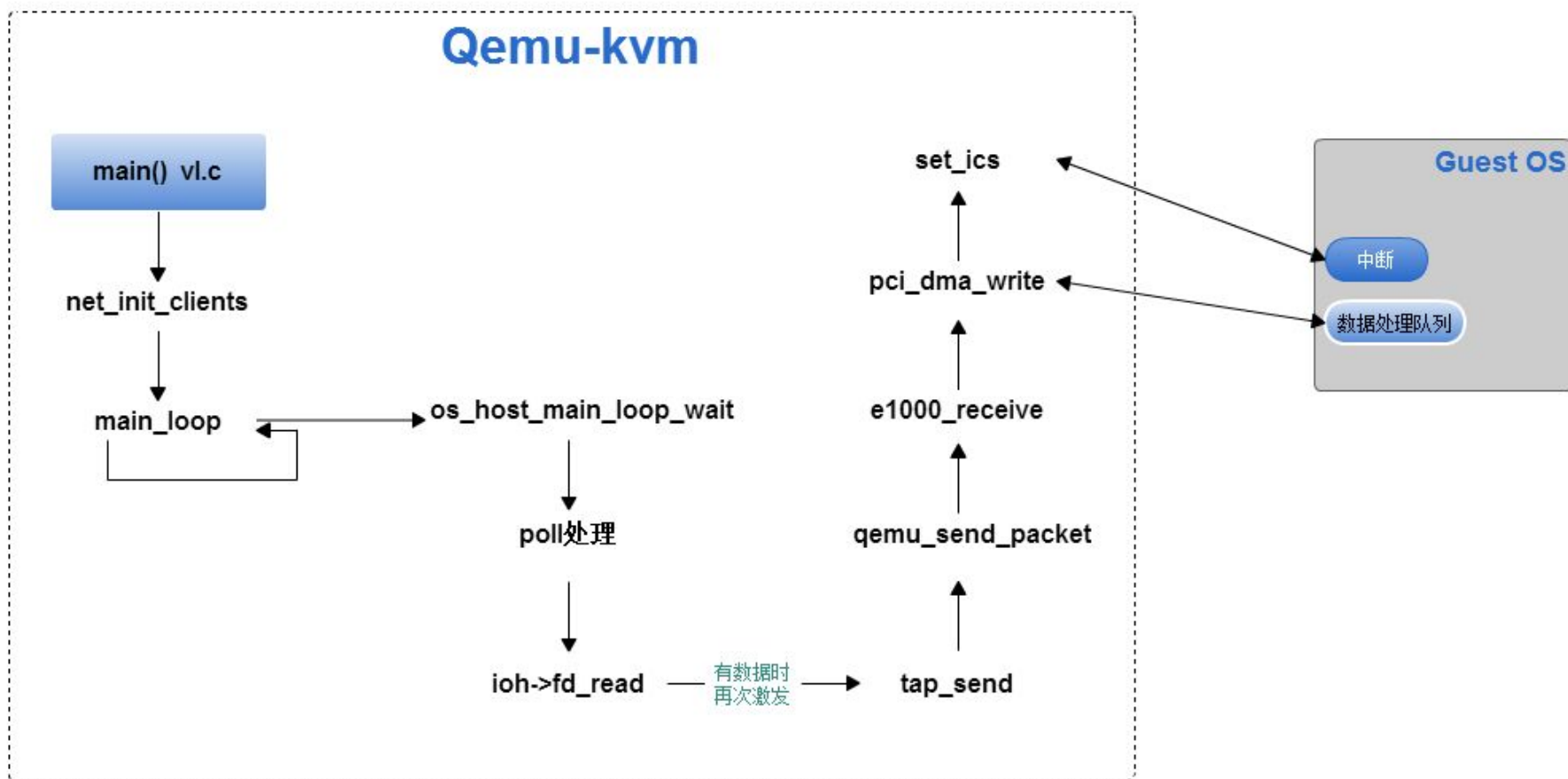
Filling EPT entry when EPT violate



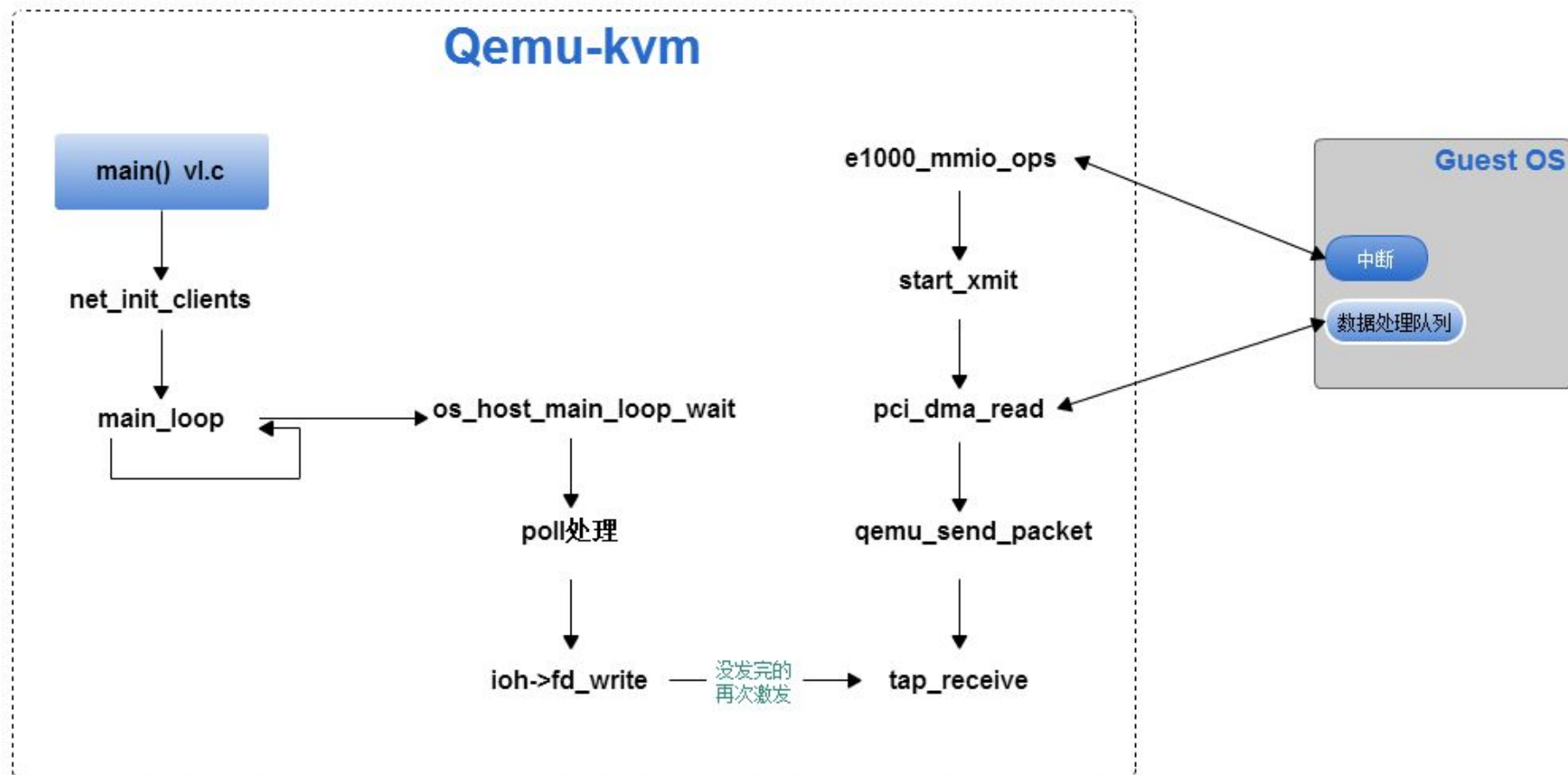
网卡虚拟化（E1000）



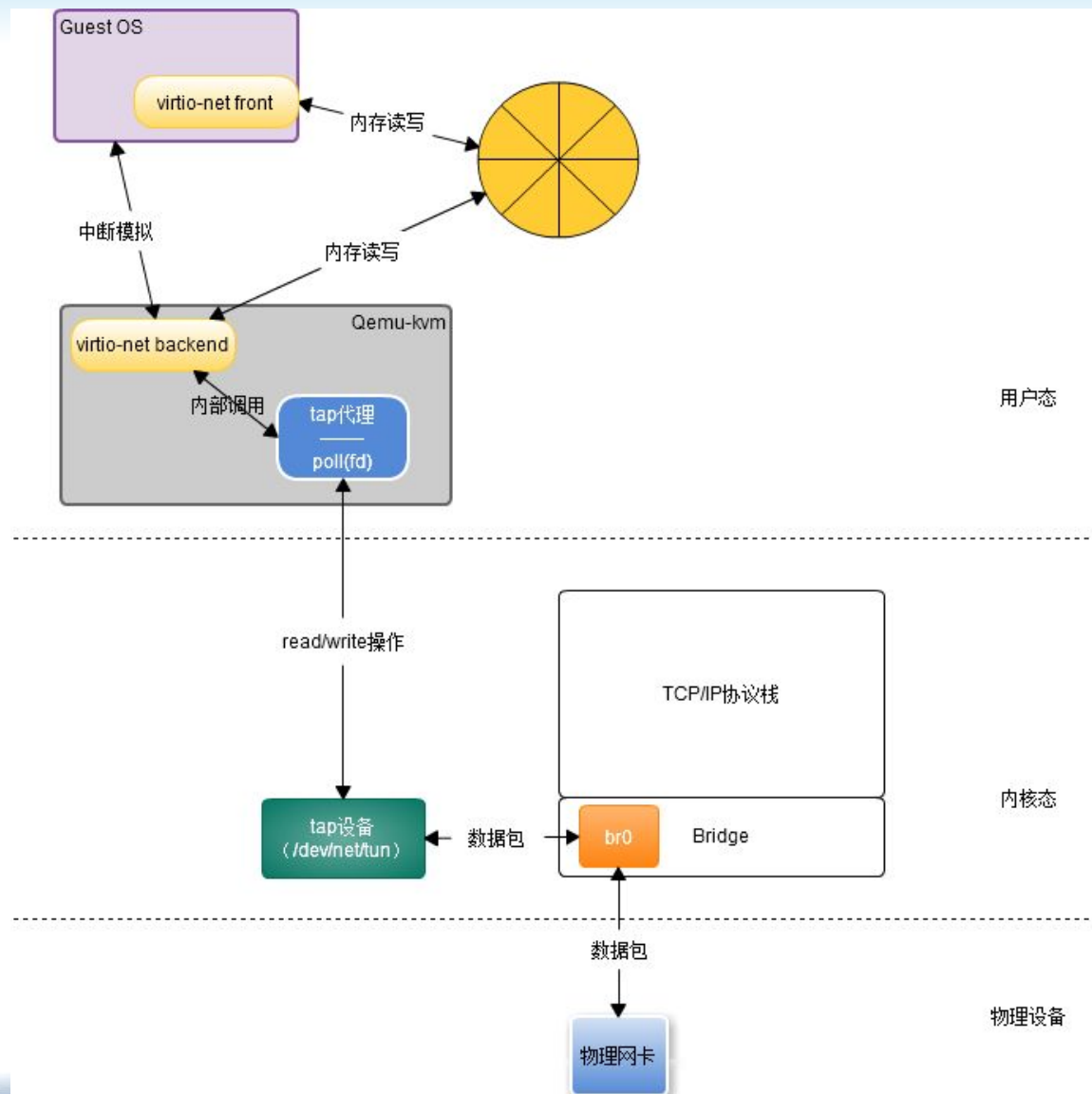
虚拟机收包



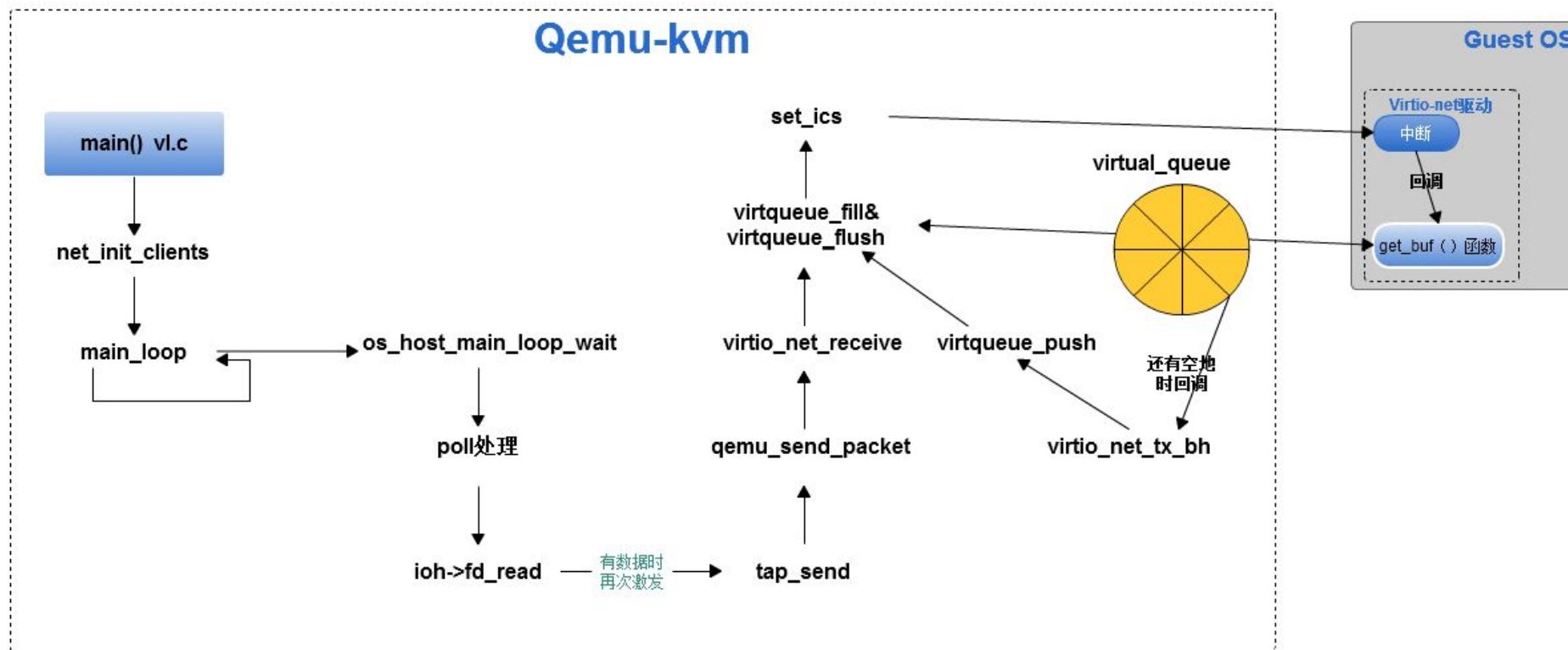
虚拟机发包



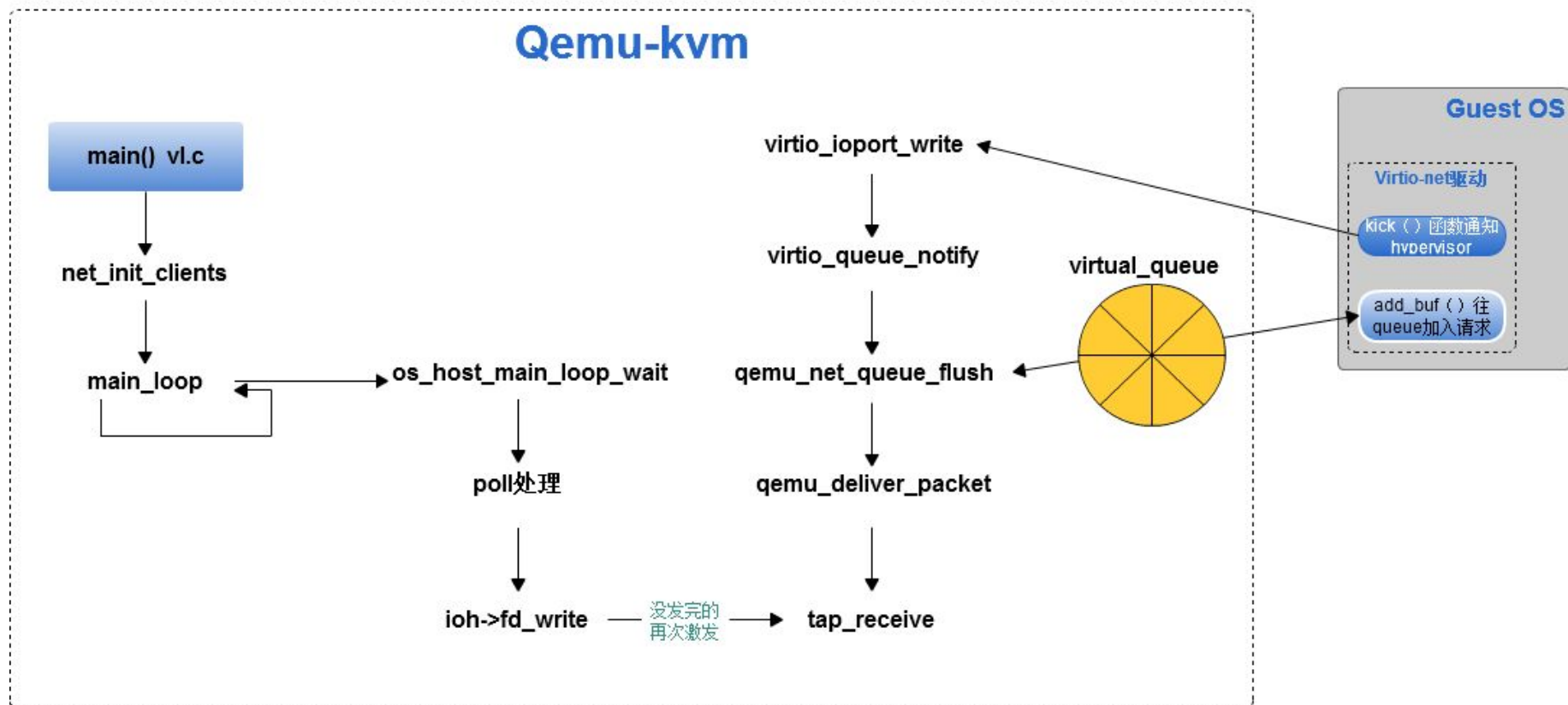
Virtio-net



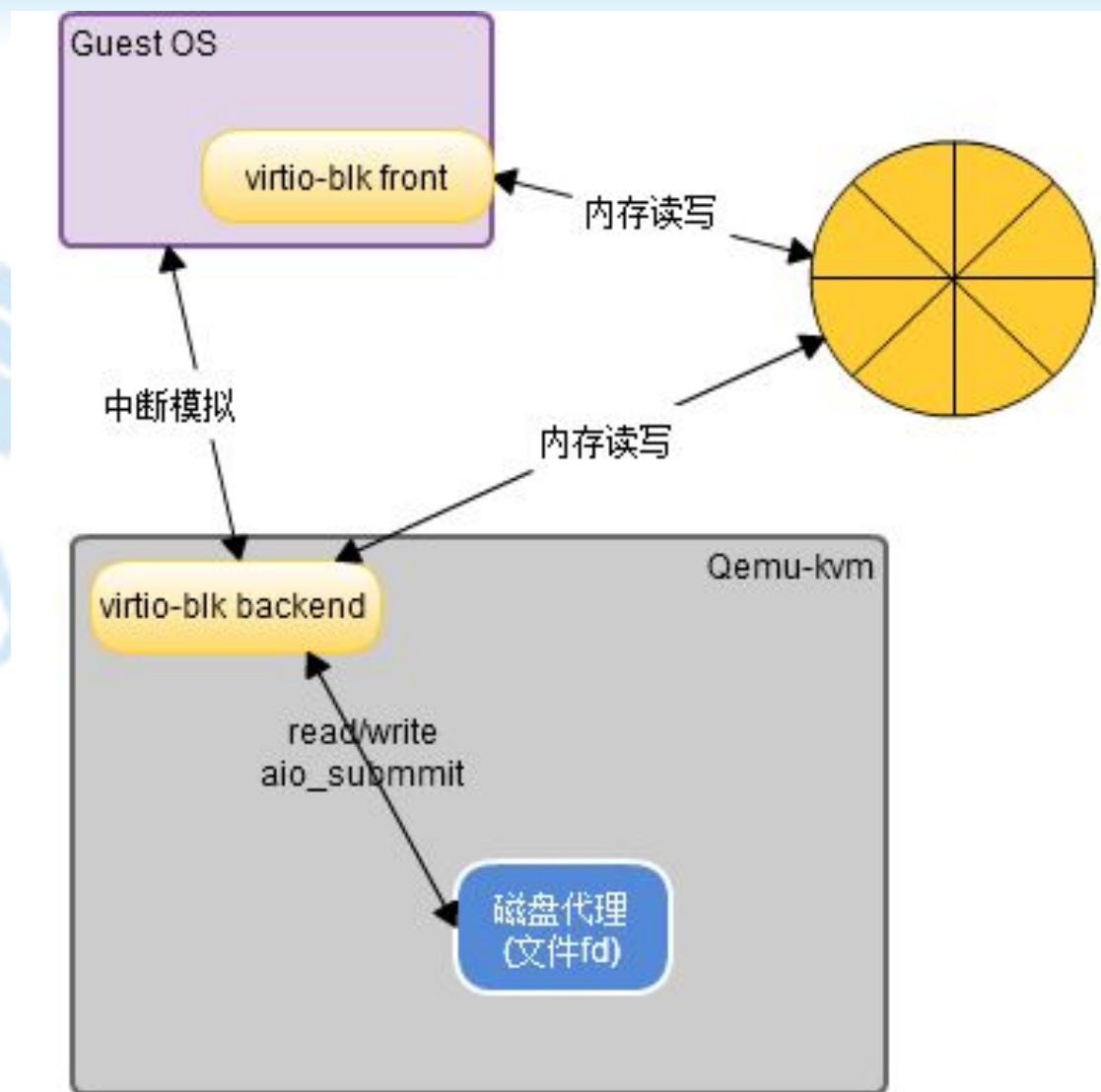
Virtio-net收包



Virtio-net发包



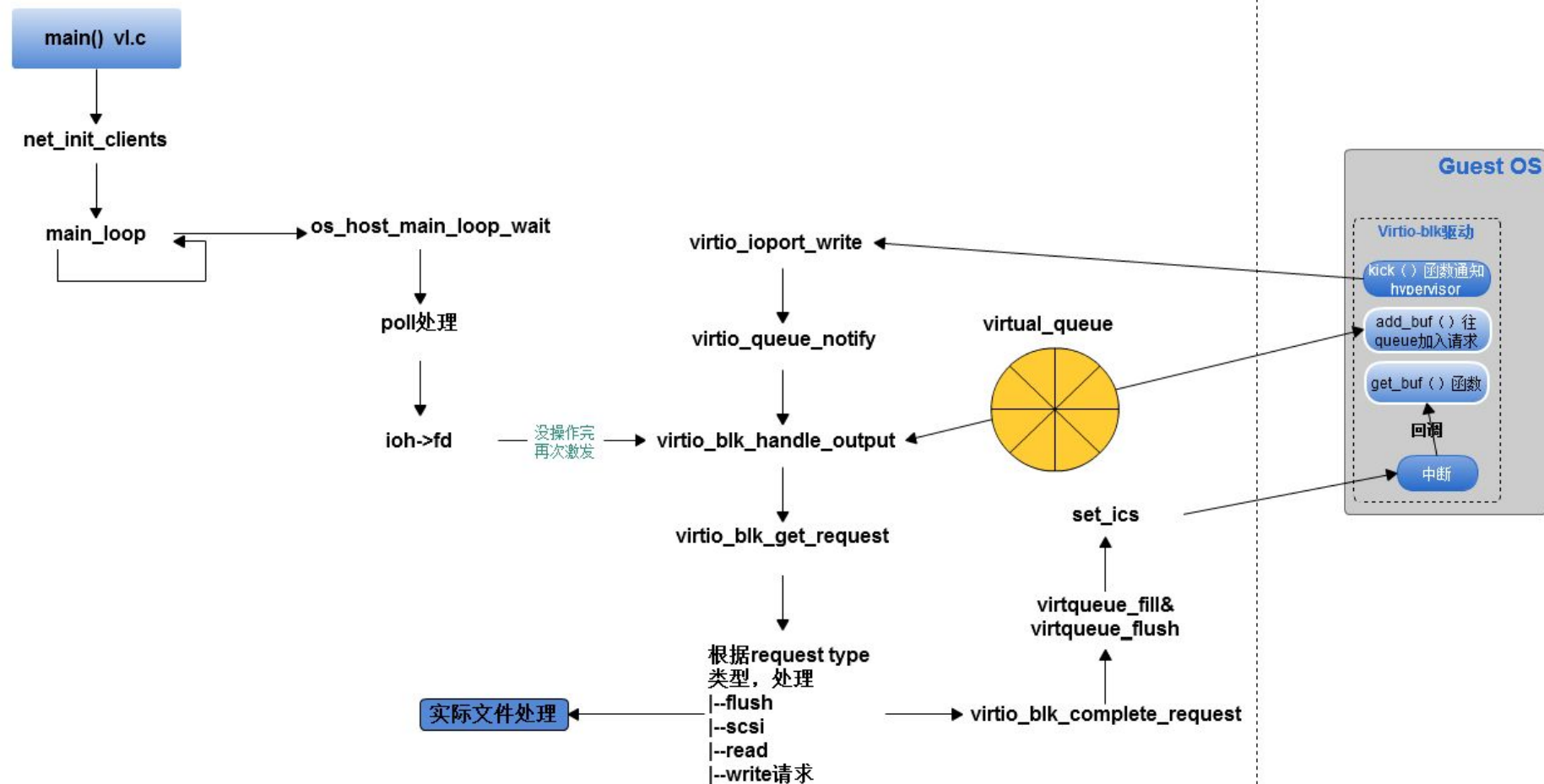
Virtio-blk



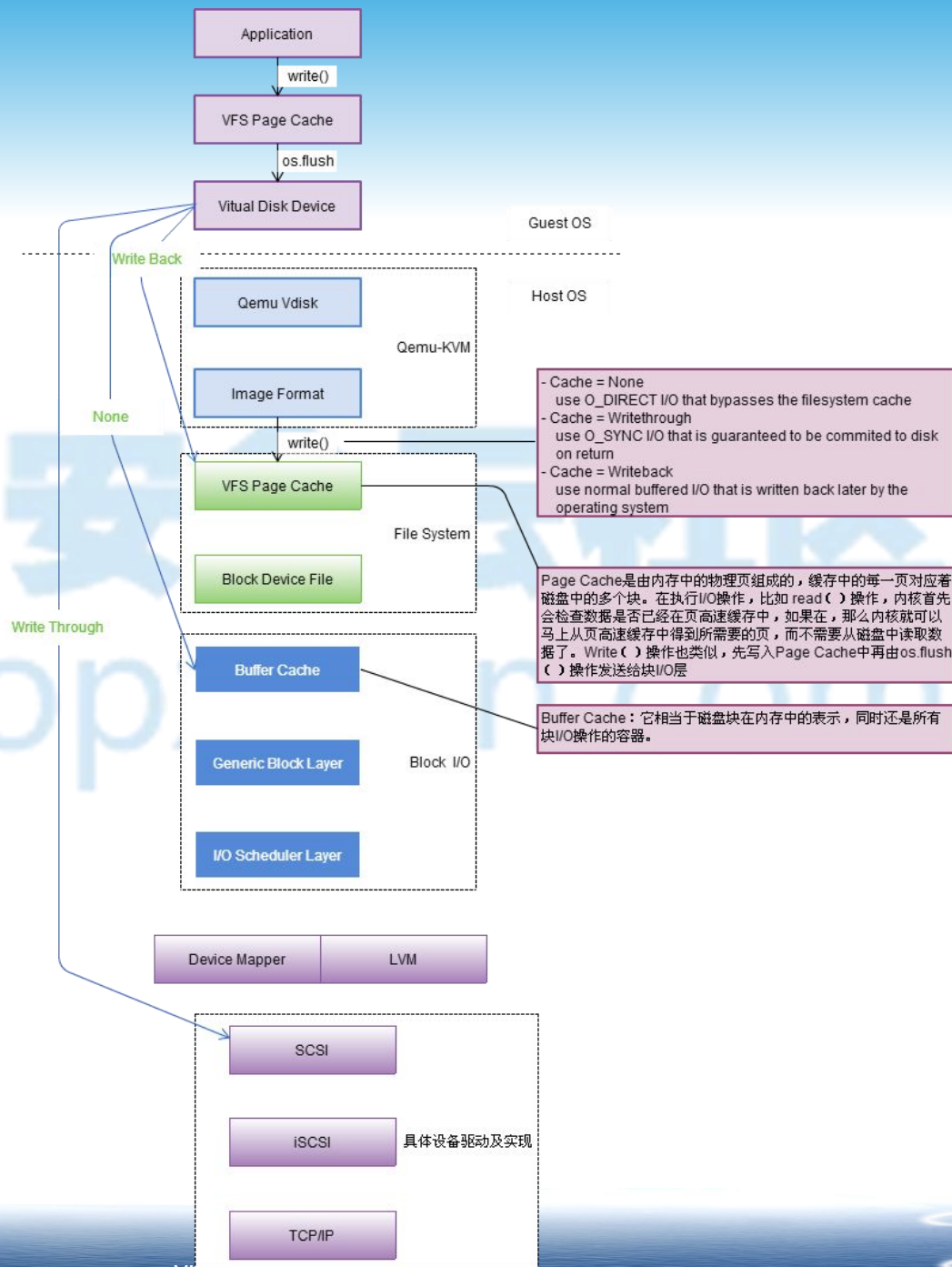
Virtio-blk读写流程

virtio_blk_get_request

Qemu-kvm

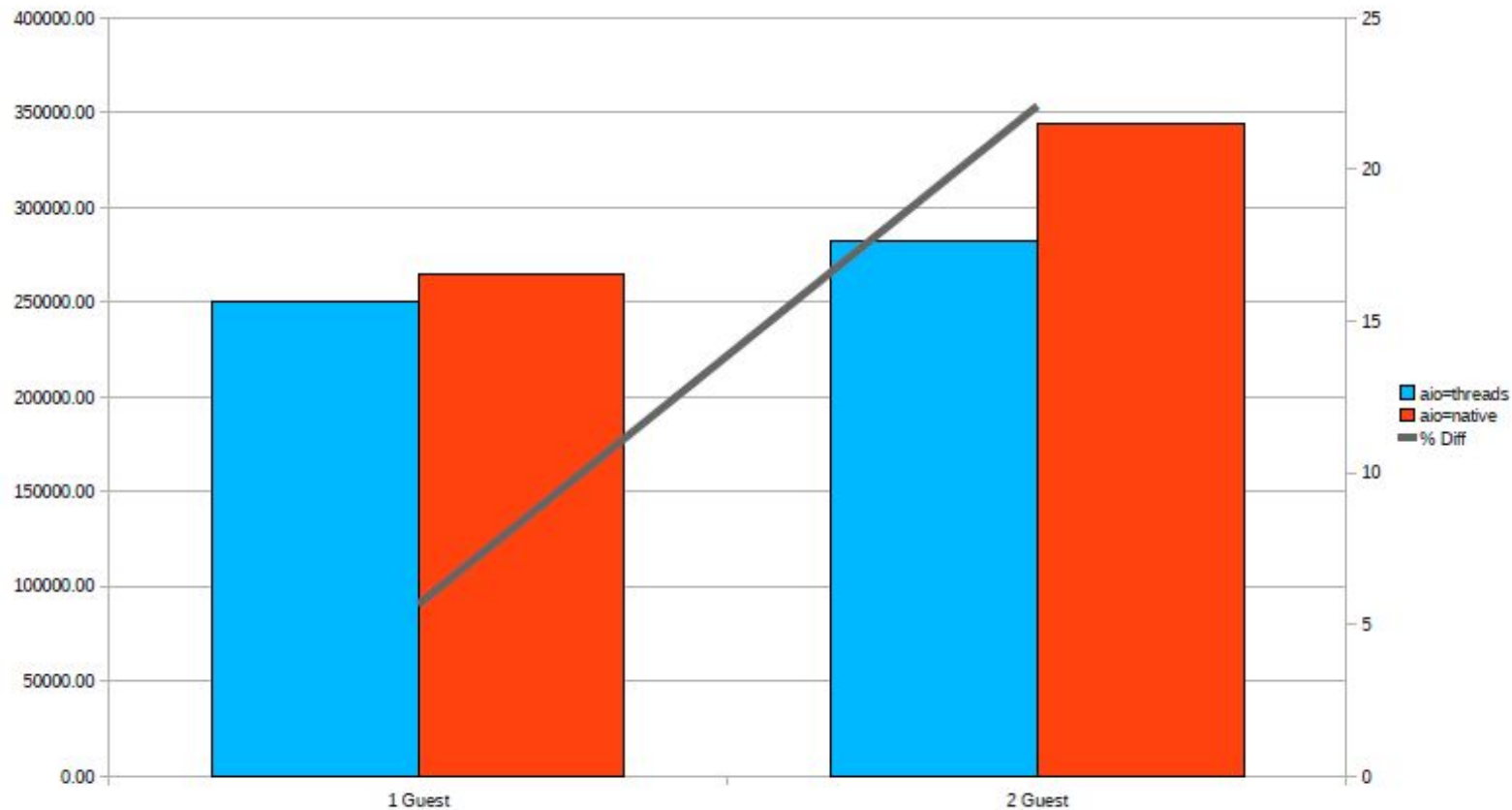


存储软件栈



用Native的性能会比threads模型的性能更高

Multi guest database testing with different AIO settings



目录

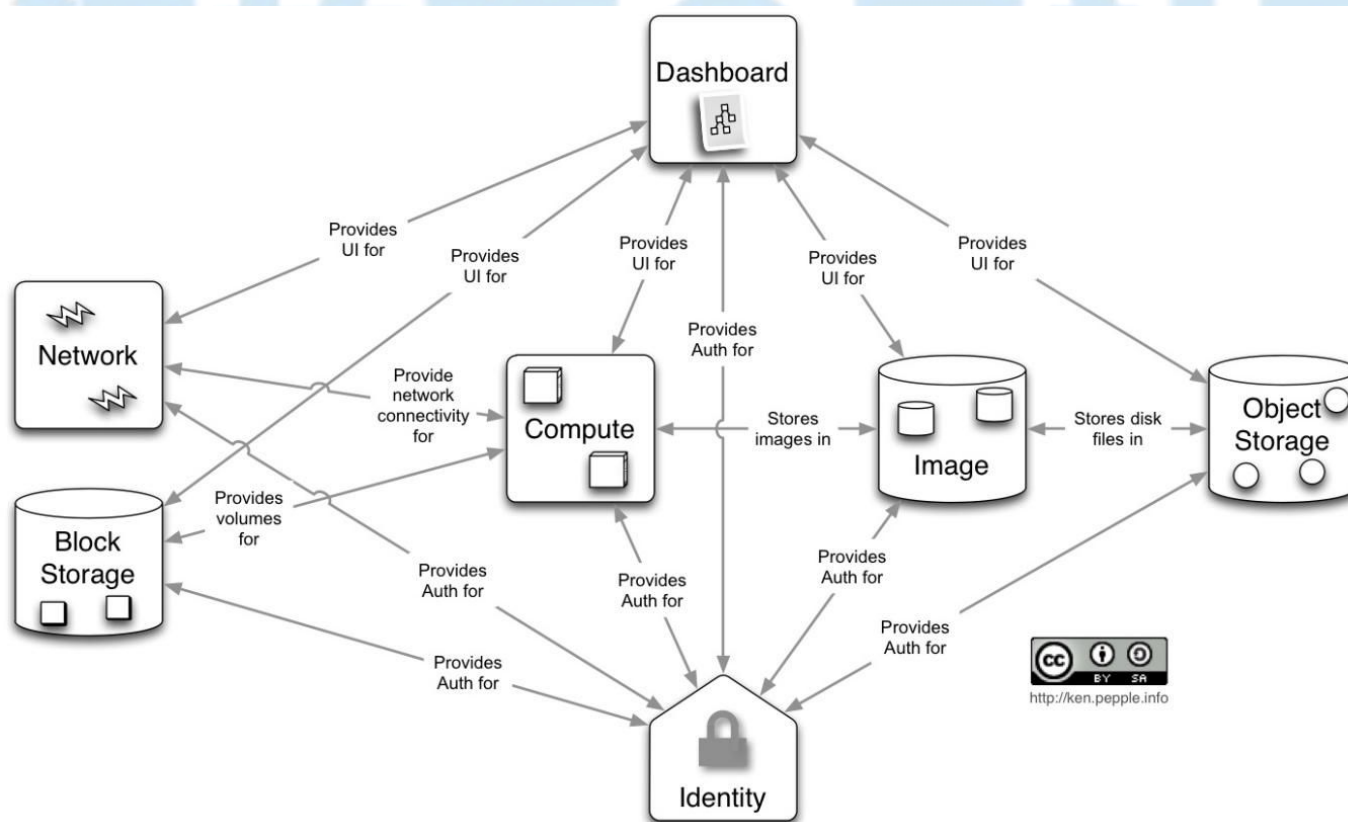
KVM技术学习

OpenStack, Docker的应对之策

广告—汉柏OPV-Suite

OpenStack

OpenStack是一个开源的云计算管理平台项目，由几个主要的组件组合起来完成具体工作。**OpenStack**支持几乎所有类型的云环境，项目目标是提供实施简单、可大规模扩展、丰富、标准统一的云计算管理平台。**OpenStack**通过各种互补的服务提供了基础设施即服务（**IaaS**）的解决方案，每个服务提供**API**以进行集成。



OpenStack为我们带来什么

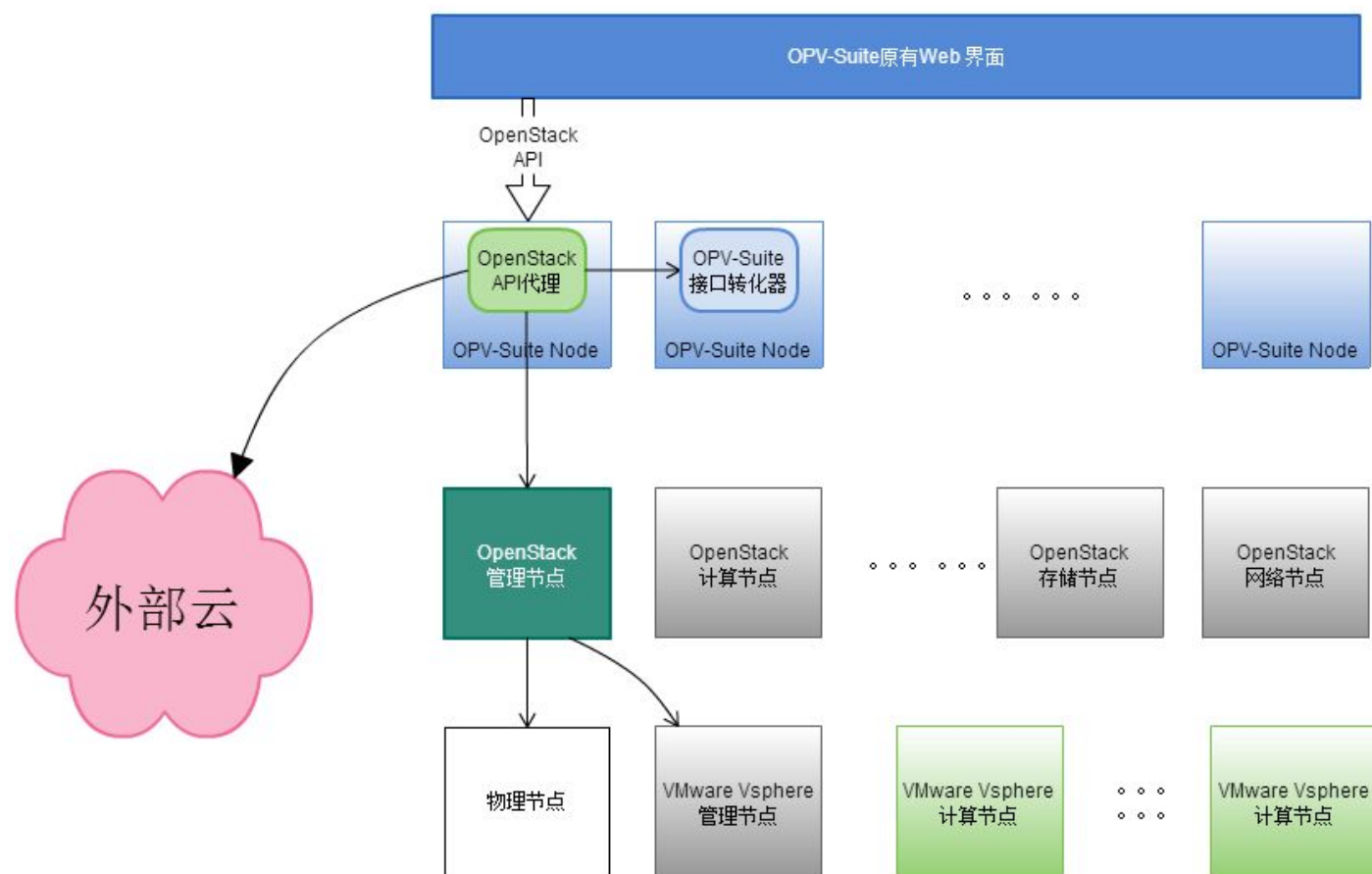
- ❖ 牛X的体系结构设计
- ❖ 全世界的高手在给你写代码
- ❖ 开放通用的接口
- ❖ 技术讨论的平台和社区
- ❖

OpenStack不能给我们带来什么

- ❖ 不是一个产品
- ❖ 不能解决“客户现场”的问题
- ❖ 方向&代码不由你控制
- ❖ 为“大千万众”服务，而不单独为你
 - 很多东西你都用不上
- ❖

其中一个方向

❖ 产品化的有自己特色的接口通用的自有产品



容器与Docker

- ❖ **Docker**基于**LXC**（容器--提供轻量级的虚拟化，以便隔离进程和资源，而且不需要提供指令解释机制以及全虚拟化的其他复杂性）技术

Docker在**LXC**容器上加入了：应用分发机制，

COW根文件系统 --> 部署多个类似镜像快上加快，

补充基本功能--**shell**，日志

- ❖ 相对于虚拟化技术

- ❖ **Pros:**

- 密度大，启动快 -- 软件栈减少两层 -- **GuestOS + KVM**
- 更容易部署**SAAS**和**PASS** -- 解耦“应用”与“操作系统”，
- 为应用分发提供可能

- ❖ **Cons:**

- 安全风险，因为共享内核 -- 被诟病最多
- 资源（**CPU, Memory, Disk, Network**）控制依赖于**Host OS**

Docker与KVM

- ❖ 1) **Docker**和**LXC**与**KVM**考虑的层次是不一样的:

前者主要考虑应用如何快速, 高效 (比如**COW**) 发布和部署;

后者主要考虑如何快速, 高效交付资源 (**CPU, Memory, Disk, Network**), 并最好做到灵活控制, 自动化控制。

- ❖ 2) **Docker**可构建在**OPV**上, 实现联动

-- 用户既能控制资源, 也能控制应用

- ❖ 3) **Docker**等技术后续会取代目前的**kvm**等虚拟化技术吗? 天马行空最理想的状况是什么呢?

更多的状况可能是: 有些地方单独用**Docker**--不关注资源; 有些地方单独用**KVM**--不关注应用; 有些地方一起用--两套系统, 分别关注资源和应用。

那能不能有一套系统和技术: 应用部署和资源分配的单位统一为一个呢? --> 貌似不可能, 因为应用需要跑在资源的管理系统 (也就是操作系统) 上, 所以, 应用的单位本身就应该小于资源的单位。

目录

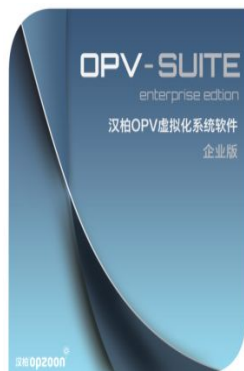
KVM技术学习

OpenStack, Docker的应对之策

广告—汉柏OPV-Suite

产品（OPV-Suite）定位及构成

产品定位

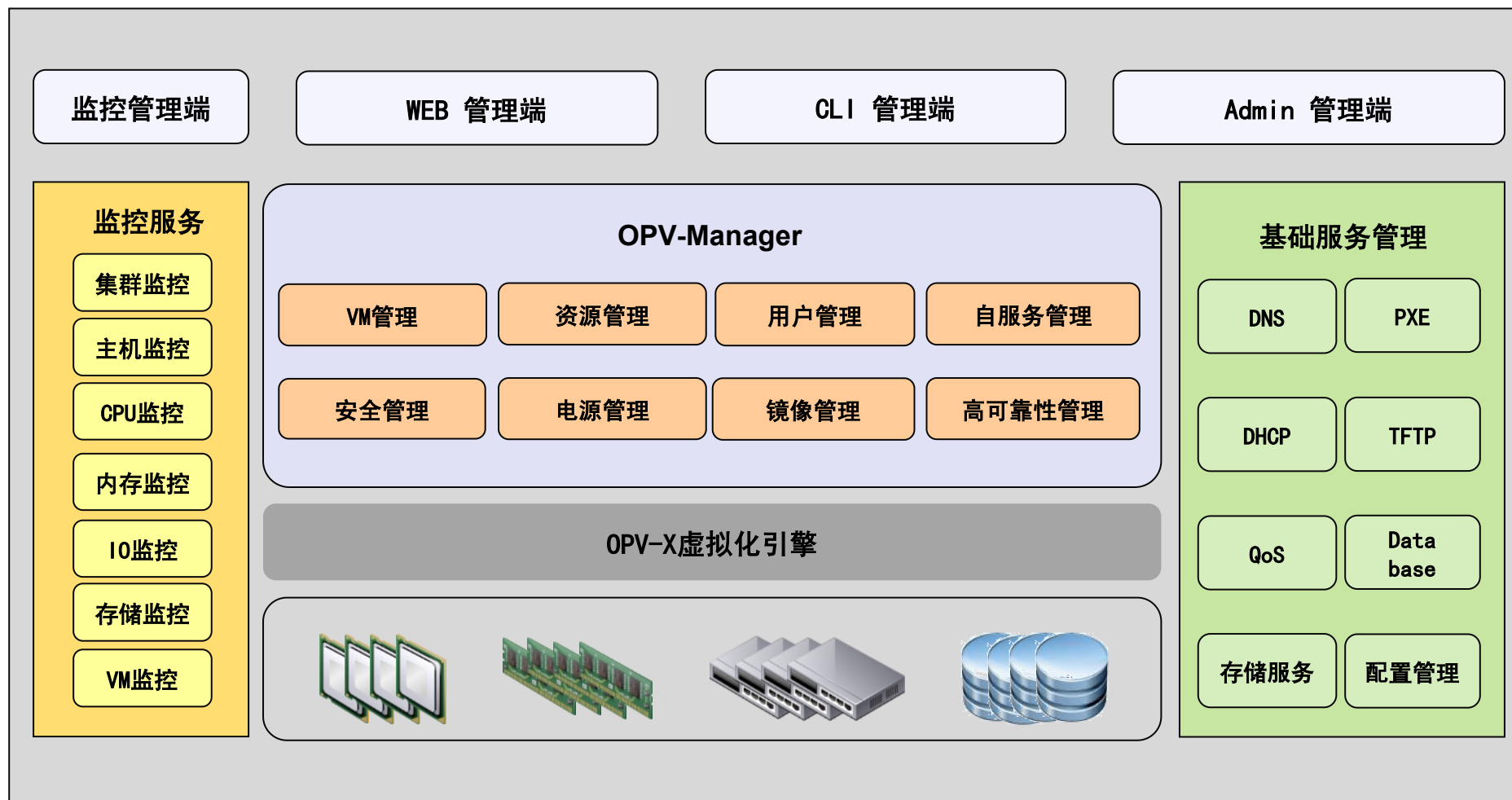


汉柏虚拟化系统软件（简称：OPV-Suite）是汉柏科技有限公司面向数据中心云计算基础架构推出的业界领先的云计算基础架构软件系统。OPV-Suite将所有服务器整合为可以按需取用的动态资源池，从而将传统数据中心改变为可扩展的、动态的、绿色的云计算数据中心。

产品构成

该系统软件由虚拟化引擎OPV-X和虚拟化管理系统OPV-M组成，OPV-X将物理硬件资源（CPU、内存、硬盘、网卡等）虚拟化并形成资源池，OPV-M实施统一的资源分配和资源管理、用户身份管理、虚拟机基础安全管理、基础网络服务管理、物理资源及虚拟机资源监控。

OPV-Suite虚拟化系统软件逻辑架构



OPV-Suite产品关键特性

计算

- 虚拟机
 - ① 支持vCPU数量32个
 - ② 支持vRAM数量32G
 - ③ 支持vDISK数量24个
 - ④ 支持vNIC数量8
- 物理机
 - ① CPU内核数量无限制
 - ② RAM数量无限制
 - ③ 集群内节点数量32台
- 支持虚拟资源热变更
- 支持物理机资源热添加

网络

- 分布式虚拟交换机
- 专用虚拟网络vLAN
- 物理网络QOS
- 虚拟网络QOS
- vDHCP服务
- vDNS服务
- 物理网卡多模式绑定

存储

- 完整模式克隆
- 链接模式克隆
 - 增量快照
- 磁盘自动精简配置
 - 共享存储
- ① FC SAN
- ② IP SAN
- 本地存储
- 分布式存储
- 存储镜像备份与恢复

管理

- 管理平台随主机安装虚拟化系统而自动安装
- 虚拟化四层管理结构
- 多种接入方式管理
 - ① B/S
 - ② C/S
 - ③ CLI
 - ④ API集成
- 平台全方位监控
- 自定义告警内容和告警方式

安全

- 虚拟防火墙
 - ① 基于L4-L7访问控制
 - ② 基于vNIC访问控制
- 基于本地和域用户的权限管理

可靠性

- 主机/虚拟机级别HA
- 管理平台分布式部署，无单点故障
- 存储多路径保障

兼容性

- 兼容业界主流IT厂商服务器、存储、网络硬件
- 兼容主流Windows、Linux操作系统

自动化

- 虚拟机迁移
- 分布式资源调度
- 智能化节能调度
- 大规模高效自动化部署

我们的成就

2项软件著作权

73项虚拟化核心技术专利


4项认证证书

3项大奖



The top half of the image features a world map in a light blue color, centered on the Atlantic Ocean. The map is set against a background of a bright sunset or sunrise with a blue sky and white clouds. The text "Thank You !" is written in a large, blue, sans-serif font with a white outline, positioned in the center of the map.

Thank You !

The bottom half of the image shows a range of snow-capped mountains under a clear blue sky. The mountains are rugged and have patches of snow and ice. The text "合作共贏" is written in a black, sans-serif font on the left side of the image.

合作共贏