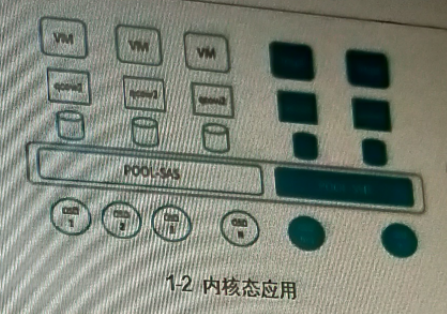
# 简介

# 分类

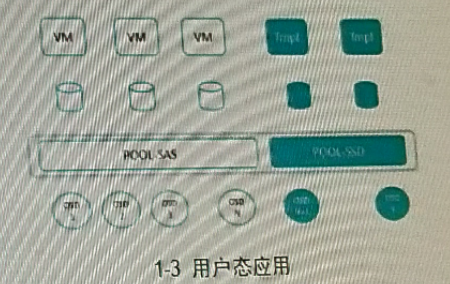
## 内核态RBD

 在内核态下，rbd块设备需要挂载至虚机运行的主机上，libvirt通过操作系统的通用块设备层，调用kmod-ceph内核模块对rbd进行读写操作。在使用的时候，需要在rbd块设备上创建qcow2磁盘格式，对于虚机磁盘数据的读写需要经过rbd和qcow2两层。

在内核态的使用方式下，rbd块设备只是作为qcow2磁盘的容器来使用，虚机磁盘的所有特性，如快照、克隆等，还是依赖于qcow2来实现。

## 用户态RBD

在用户态下，rbd块设备对于主机是不可见的，虚机qemu-kvm进行对rbd的读写是直接调用librbd接口来实现，不经过操作系统内核。虚机的磁盘数据是以raw格式存放在rbd上，相对于内核态，少了一层qcow2解析。



用户态下，qemu不再使用通用的qcow2磁盘格式，libvirt针对性的开发了基于librbd的生命周期接口，包括虚机启动、挂起、快照、在线克隆等。离线状态下对于虚机磁盘的操作也无法使用qemu-img工具，必须使用rbd提供的快照、克隆等接口。

RBD块设备有三种使用方式：内核rbd、nbd、用户态rbd。用户态方式是ceph开源社区最推荐的一种。无论是内核态rbd还是nrbd，都是㤇在其上创建qcow2格式才能实现kvm虚机的在线生命周期功能，另外内核态的读写IO路径相对于用户态而言要长，所以从读写效率而言，librbd是三种方式中最高的。

从代码可维护性角度而言，内核态是跟操作系统内核关系很大，可维护性很差。而且linux内核代码的管理非常严格，有些定制化的需求几乎是不可能提交成功的，这方面用户态librbd是最好的。

从使用角度来讲，rbd需要map到linux主机上，map过程牵涉到的模块也非常的多，除了问题排查难度很大。

# 源码