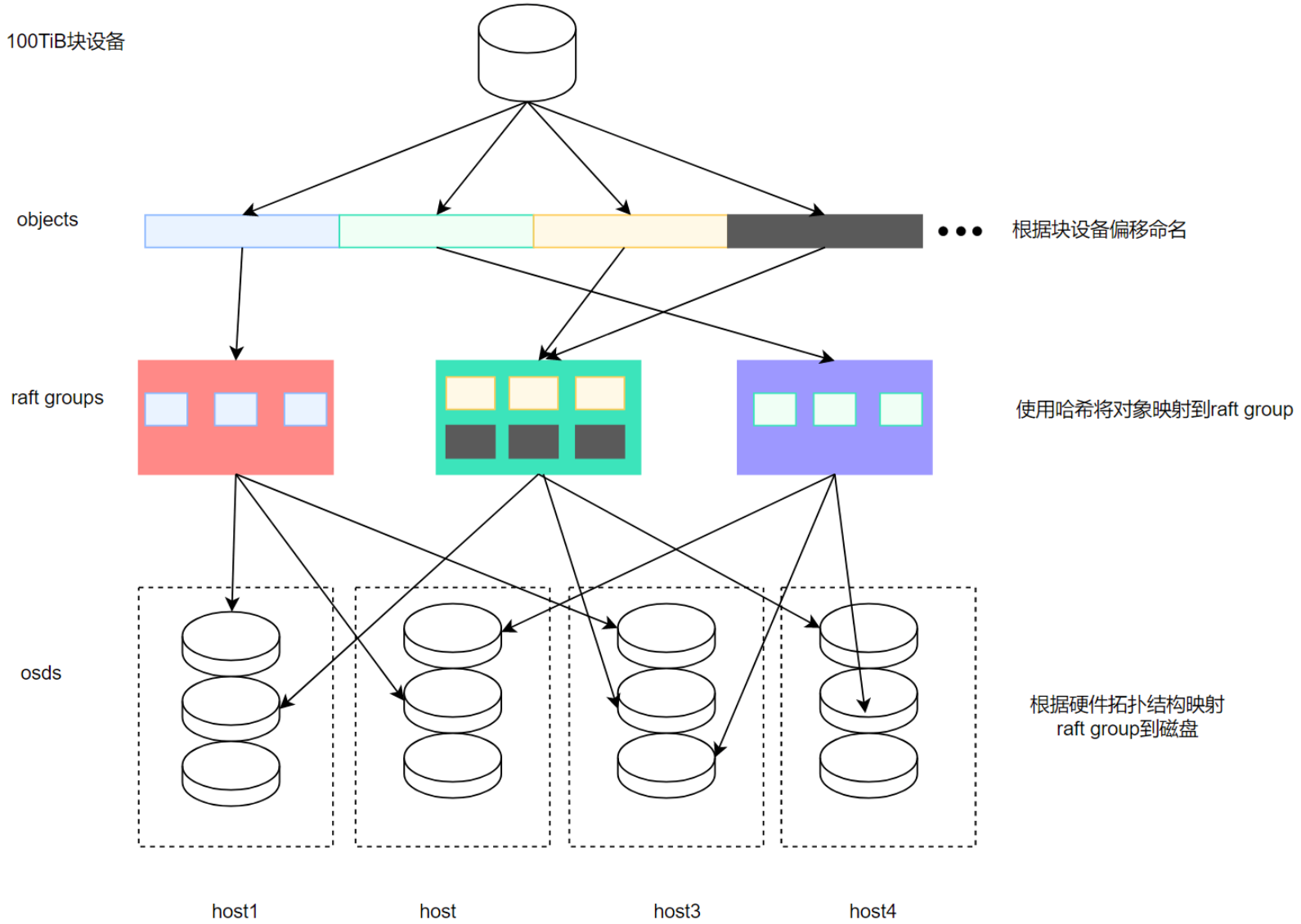
fastblock:高性能分布式块存储的设计与实现

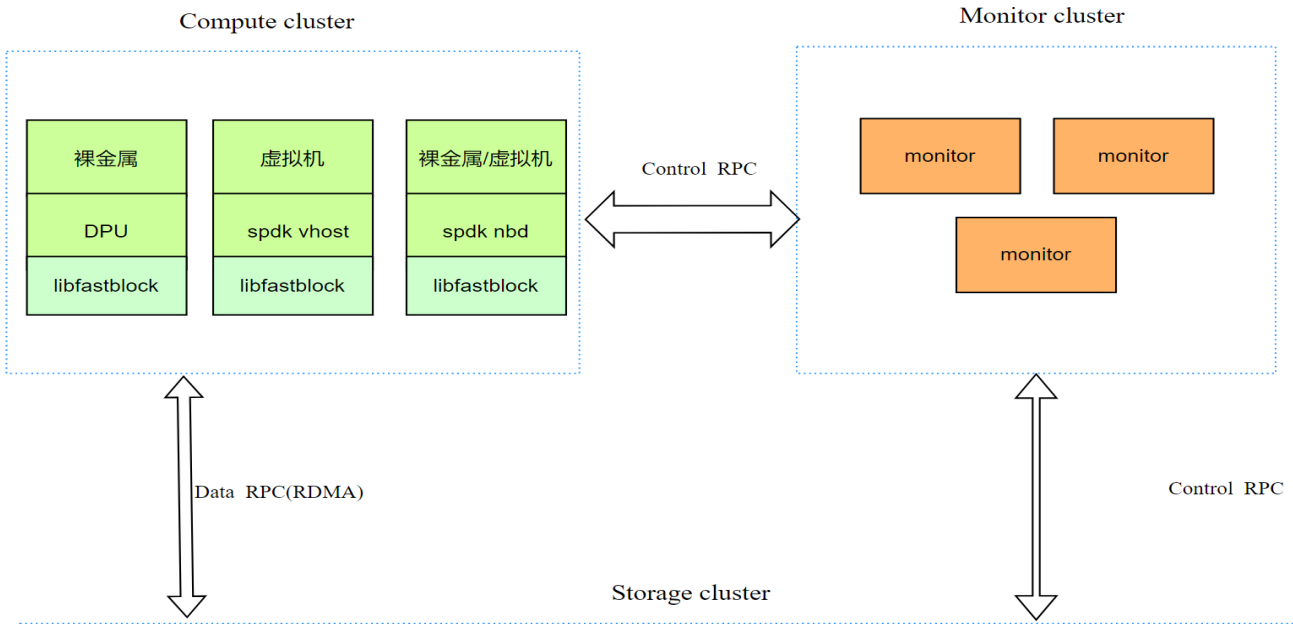
分布式块存储的需求与fastblock的解决方案

分布式块存储的需求:
1、IO接口能对接上qemu、裸金属等
2、IO均匀打散到众多磁盘
3、高IOPS
4、低延迟
5、高可用
6、高可靠
7、低成本

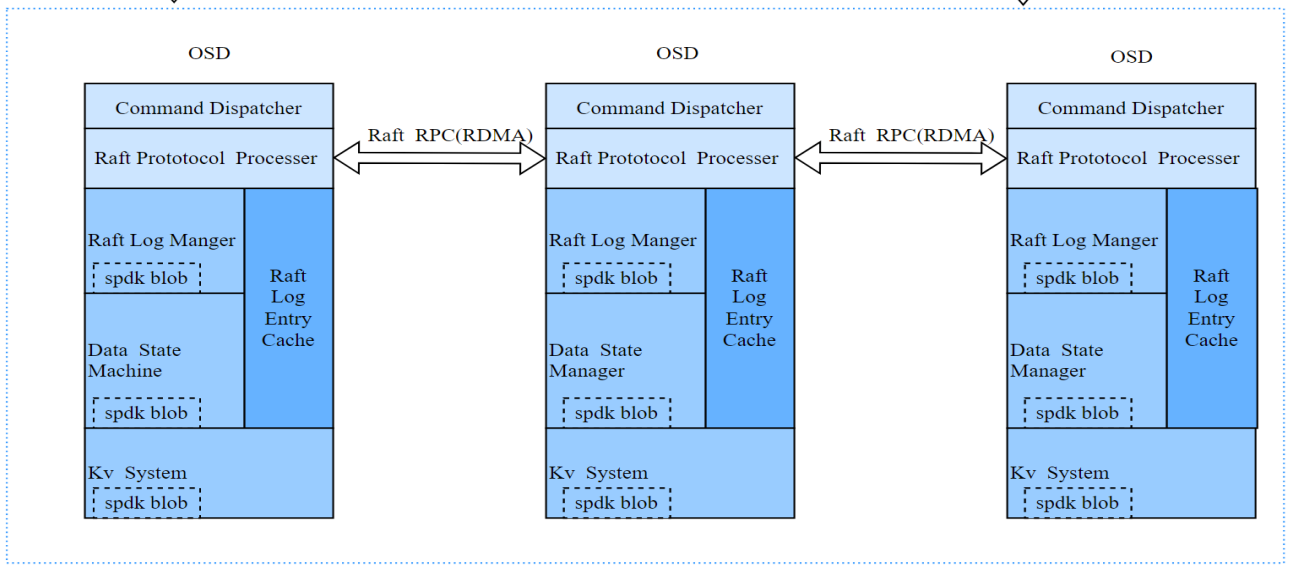
fastblock的解决方案:
1、块接口: 支持dpu、vhost、nbd对接
2、负载均衡: 管理面可定制，均衡分配
3、高IOPS: 用户态nvme驱动
4、低延迟: RDMA通信、极简IO路径
5、系统高可用: raft
6、数据高可靠: 多副本且定期数据验证
7、低成本: SPDK编程模型非常省CPU



fastblock组件架构



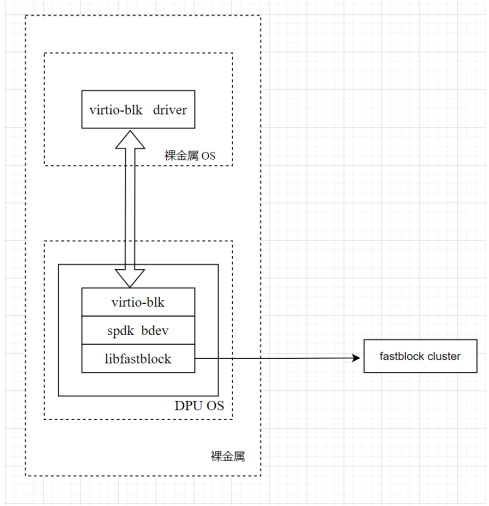
- Compute Cluster对应计算服务；
- Monitor Cluster负责维护集群元数据；
- Storage Cluster对应存储集群，每个存储节点中运行多个OSD；
- DPU和vhost客户端延迟较低用于线上使用，nbd客户端仅作为补充用于调试
- Data RPC走RDMA网络，主要用于客户端读写和raft复制
- Control RPC走TCP网络，主要用于获取集群元数据；
- Command Dispatcher是消息分发模块，对外处理OSD RPC和客户端的读写RPC；
- Raft Protocol Processer实现了raft协议，支持raft成员变更和raft组增删等；
- Raft的本地存储包括了raft日志、raft状态机和raft kv系统，
- Raft Log Manager和Data State Manager分别负责管理raft日志和raft数据状态机；
- Raft log Entry Cache是一个raft log缓存，可提高状态机性能；
- Kv System负责存储raft元数据(如votedFor、term、commitIndex等)



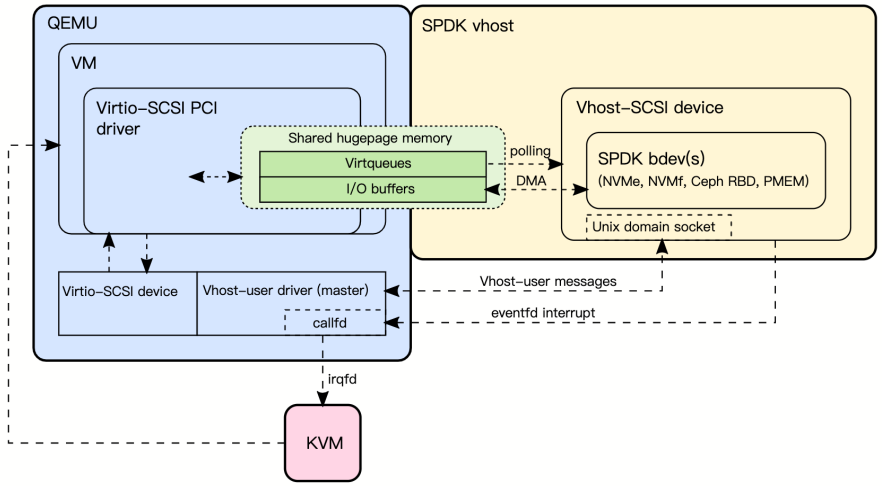
- OSD使用spdk编程框架编写，利用用户态nvme驱动、无锁队列等特性降低IO路径延迟
- OSD中使用RDMA进行零拷贝、内核旁路、无需CPU干预的网络通信
- OSD使用multi raft进行数据复制，保证数据高可用且可靠性
- Monitor采用golang编写，实现了简单、可靠、易定制的集群元数据管理

fastblock对接接口及IO路径

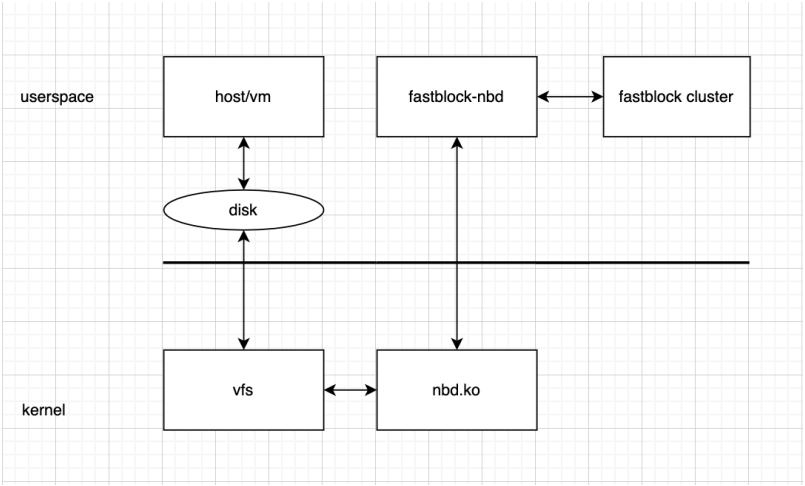
dpu

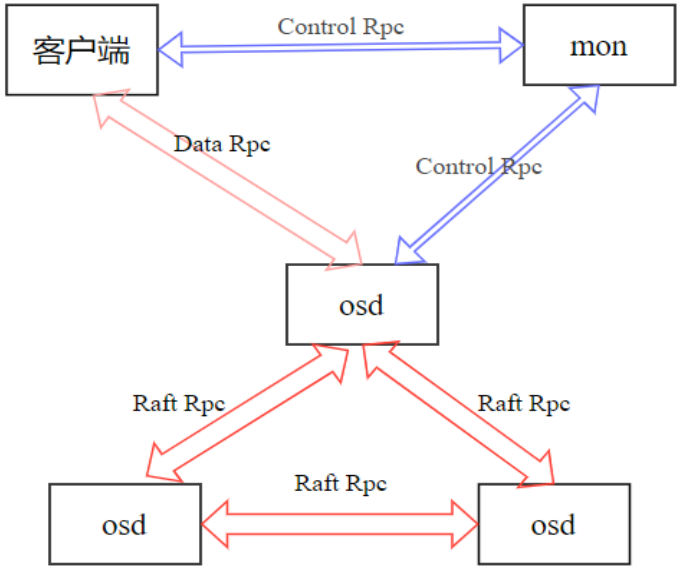


vhost



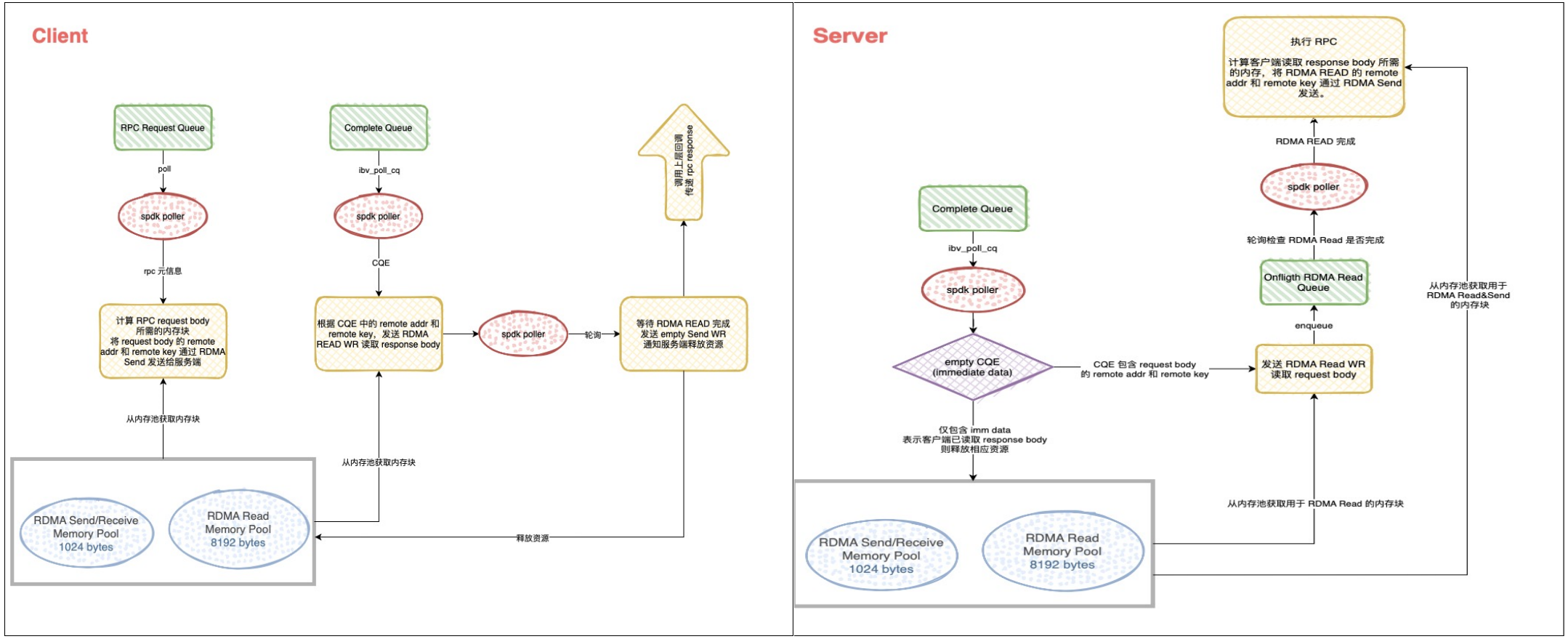
nbd



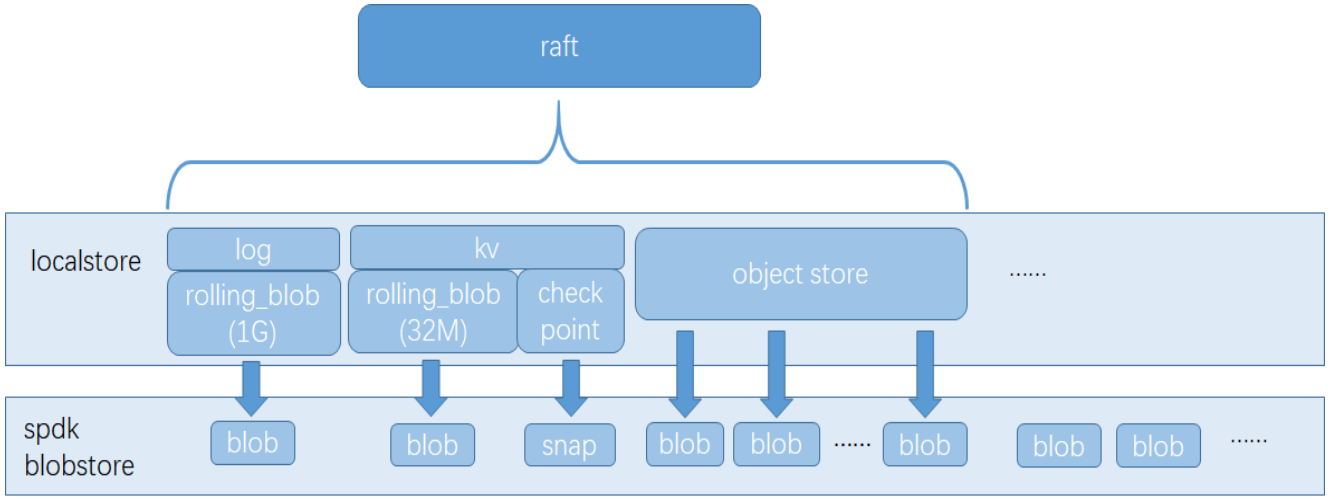


fastblock RDMA RPC系统

第一个版本复用了spdk的nvmf通信传输层，第二个版本进行了大规模重构。
1、第二版本避免包含大量spdk处理nvme命令字的代码和nvme状态机；
2、能够满足 fastboot rpc 应用层诸如超时、错误处理等需求；
3、且减少了内存拷贝，性能优于spdk nvmf版本， 延迟约为12us左右；
4、新版本传输层结合使用 RDMA Send 和 RDMA Read，有利于大规模网络场景下的流控



fastblock本地存储引擎



- disk_log: 存储raft log，一个raft group对应一个spdk blob；
- object_store: 存储数据状态机，一个对象对应一个spdk blob；
- kv_store: 每个cpu核拥有一个spdk blob。保存当前cpu核上的需要保存的所有kv数据，包括raft的元数据、存储系统本身的数据；
- rolling_blob是一个循环使用的spdk blob

现状及未来的开发计划

现状:
1、目前端到端4k单线程随机读写延迟低于100微秒，4k随机读写并发IOPS达到百万级;
2、已完成对接qemu、裸金属和openstack云平台;
3、已基本完成监控数据导出、卷快照等重要功能的开发，即将发布;
4、正添加更多测试以验证raft成员变更、故障恢复、数据一致性等;

未来的开发计划:
1、实现共享卷功能，支持多写多读，为数据库系统提供更多功能;
2、实现卷QoS、卷加密功能;
3、开发替代blobstore的性能更好的本地存储引擎;
4、实现可定制的monitor的raft group分配插件;
5、部署工具开发及系统配置文件简化;
6、丰富上层API，支持类似librados的对象API和类似tikv的支持分布式事务的kv API

欢迎加入: https://gitee.com/openeuler/fastblock

THANKS

THANKS

THANKS