EuroSys'24

哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

CSAL: the Next-Gen Local Disks for the Cloud

Yanbo Zhou[1], Erci Xu[*1], Li Zhang[1], Kapil Karkra[2], Mariusz Barczak[2], Wayne Gao[2], Wojciech Malikowski[2], Mateusz Kozlowski[2], Łukasz Łasek[2], Ruiming Lu[1], Feng Yang[1], Lilong Huang[1], Xiaolu Zhang[1], Keqiang Niu[1], Jiaji Zhu[1], Jiesheng Wu[1]

[1]Alibaba Group; [2]Solidigm

***Keywords*:** NAND Flash, Caching, Cloud Storage

CSAL：下一代云本地盘

论文汇报：杨大荣

指导老师：夏文教授

目录

- 背景介绍 (Introduction)
- 研究动机 (Motivation)
- 设计实现 (Design)
- 测试评估 (Evaluation)
- 论文总结 (Conlcusion)

1.1 论文概述

1.背景：单节点云服务器通过**虚拟化**为多台虚拟机（VM）的方式为多个客户提供**ECS弹性云服务**

2.问题

- 有限的物理资源（存储、计算）**限制**单节点能够部署的VM数
- 计算资源快速发展，带来部署密度提升的**机遇**
- 存储资源面临困境，成为提升部署密度的**瓶颈**

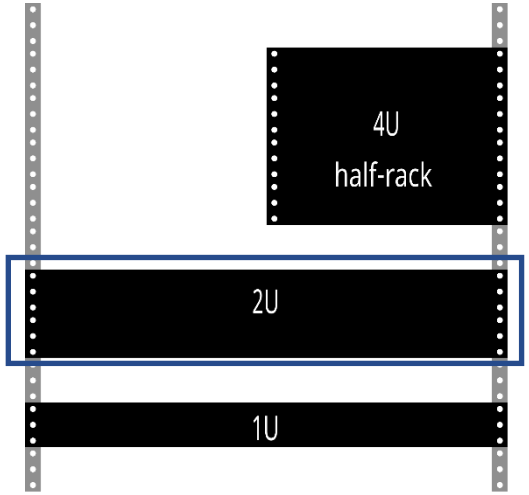
➡ **容量和性能无法兼得！**

3.目标：结合新兴的QLC-SSD设计同时**满足容量和性能要求**的存储系统，突破存储瓶颈，提升云服务器的部署密度

1.2 云本地盘(1/3)

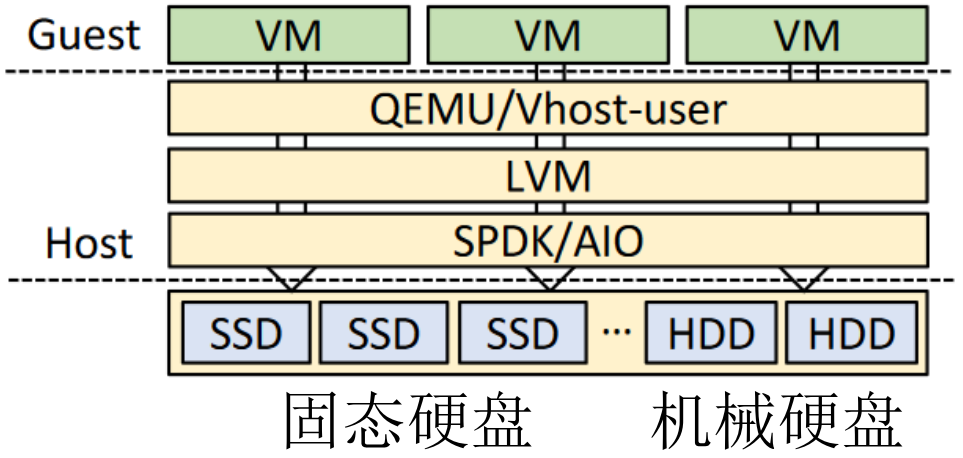
1.云服务介绍

- 虚拟化服务与物理资源
- 服务水平目标（SLO，Service Level Object）
- 机架单元与云服务器大小



2.云本地盘概念

- 挂载为块设备
- 通过SPDK/AIO访问NVMe SSDs/HDD
- LVM将其虚拟化为多个逻辑设备
- 每个VM使用若干逻辑设备



1.2 云本地盘(2/3)

3.资源划分

Amazon EC2 Overview Features Pricing Instance Types FAQs Getting Started Resources

Amazon EC2 D3服务

Model	vCPU	Memory (GiB)	Instance Storage (TB)	Aggregate Disk Throughput (MiB/s)	Network Bandwidth (Gbps)	EBS Bandwidth (Mbps)
d3.xlarge	4	32	3 x 2 HDD	580	Up to 15	850
d3.2xlarge	8	64	6 x 2 HDD	1,100	Up to 15	1,700
d3.4xlarge	16	128	12 x 2 HDD	2,300	Up to 15	2,800
d3.8xlarge	32	256	24 x 2 HDD	4,600	25	5,000

- 通常以物理资源的一定比例（如1/8）为粒度做选项配置
- 物理资源的固定配置能够让部署密度最大化

1.2 云本地盘(3/3)

4.发展机遇

- CPU的核心数和单核心算力迅速发展
- 单服务器能够实现**计算资源复用**
- 在保证SLO下，能实现多倍的**部署密度提升**

提升部署密度的关键

如何在满足SLO前提下实现**存储资源的复用**？

5.存储瓶颈

- 存储的SLO包括**性能**和**容量**两个方面
- **存储资源的资源复用是困难的**
- 单纯的加盘是不现实的（机架、成本）

存储器	容量SLO	性能SLO
HDD	满足	*无法满足
HP-SSD[1]	*无法满足	满足

[1] HP-SSD, High-Performance SSD

1.3 QLC-SSD

1.存储特性

- 与HDD可相当的存储容量
- 比HDD快10x的吞吐

	SLC	MLC	TLC	QLC	HDD
Space (TB)	0.6	1.2	6.4	23	22
Read (GB/s)	7.2	7.0	6.8	6.0	0.26
Write (GB/s)	6.1	5.2	4.2	2.5	0.26
IOPS-R (K)	1500	1300	1000	800	0.24
IOPS-W (K)	1350	800	150	5.7	0.46
Endurance (%)	2000	1500	333	100	-
Cost (%)	400	200	133	100	-

2.接口形态

(1)通用QLC-SSD

- 提供闪存转换层（FTL）
- 可进行顺序/随机读写

QLC-SSD是最理想的存储SLO解决方案 ✓

(2)ZNS[1] QLC-SSD

- 格式化划分为**若干的Zone**
- Zone内限制为**顺序写**
- 上层自行完成**随机到顺序写**的转换管理

[1] ZNS, Zone Namespace

2.1 动机概述

1.关键问题

- 存储资源的复用是困难的（HDD，HP-SSD）
- 最理想方案QLC-SSD的初步尝试，**都无法满足预期SLO（写性能）**

2.原因分析

（1）通用QLC-SSD的两级写放大

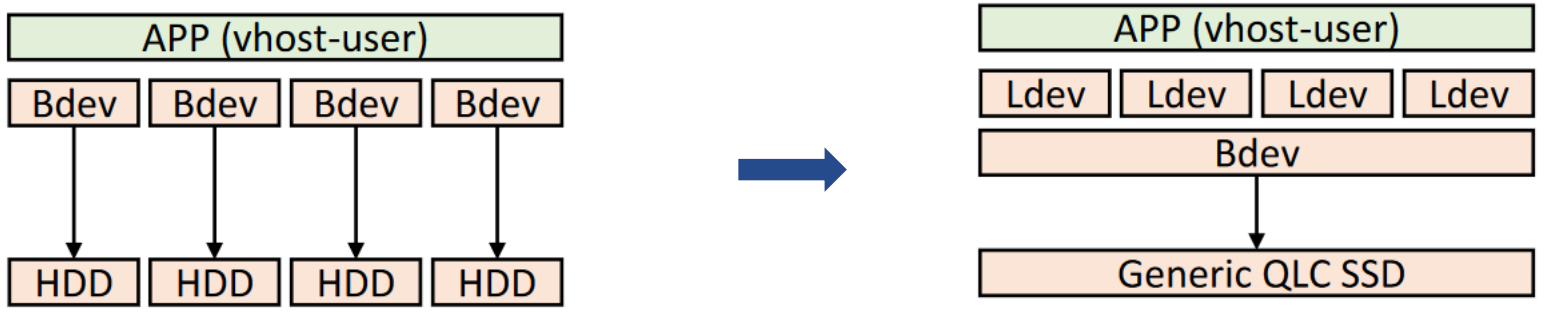
- **设备级写放大：**通用QLC-SSD上采用大粒度64KB管理使得4KB写失配
- **介质级写放大：**QLC-SSD内部垃圾回收

（2）ZNS QLC-SSD的随机写失效

- **随机写失效：**顺序Zone内只支持顺序写入

2.2 初始方案一(1/4)

1.方案1：通用QLC-SSD



(a) Legacy approach with HDD（现有方案）　(b) QLC as a drop-in replacement

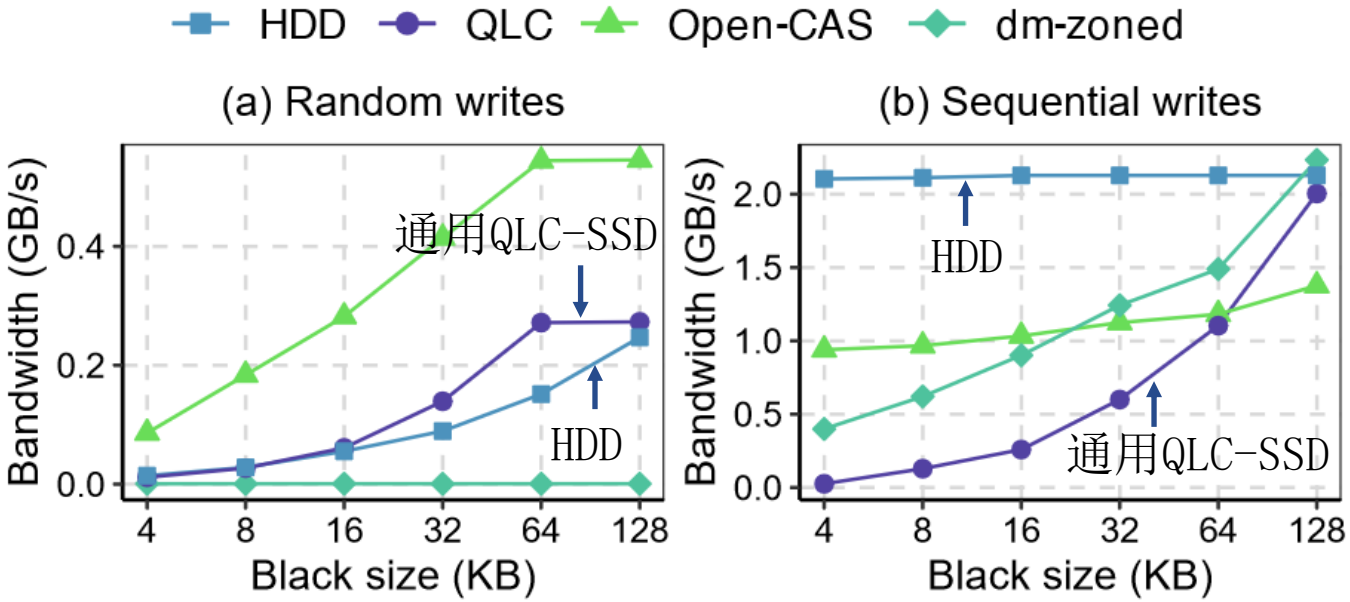
- 直接用通用型QLC-SSD做本地盘
- 上层应用通过LVM的划分来直接复用

测试：

- 设置了8-VMs（每个7-vCPU，28G内存）
- 在VMs内使用FIO测试（8-jobs，128队列深度）

2.2 初始方案一(2/4)

2.测试结果

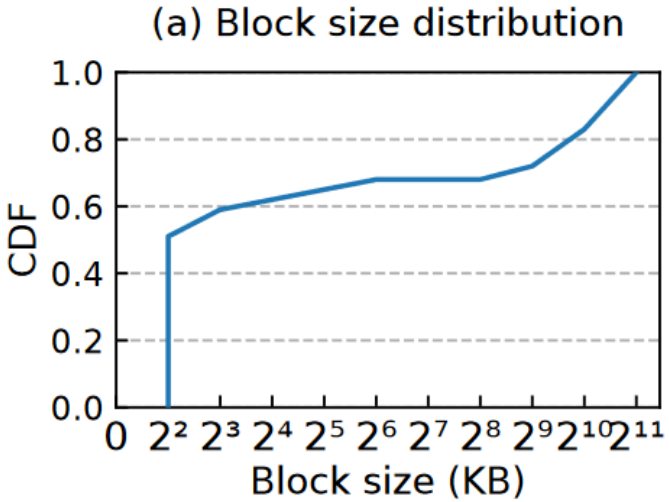


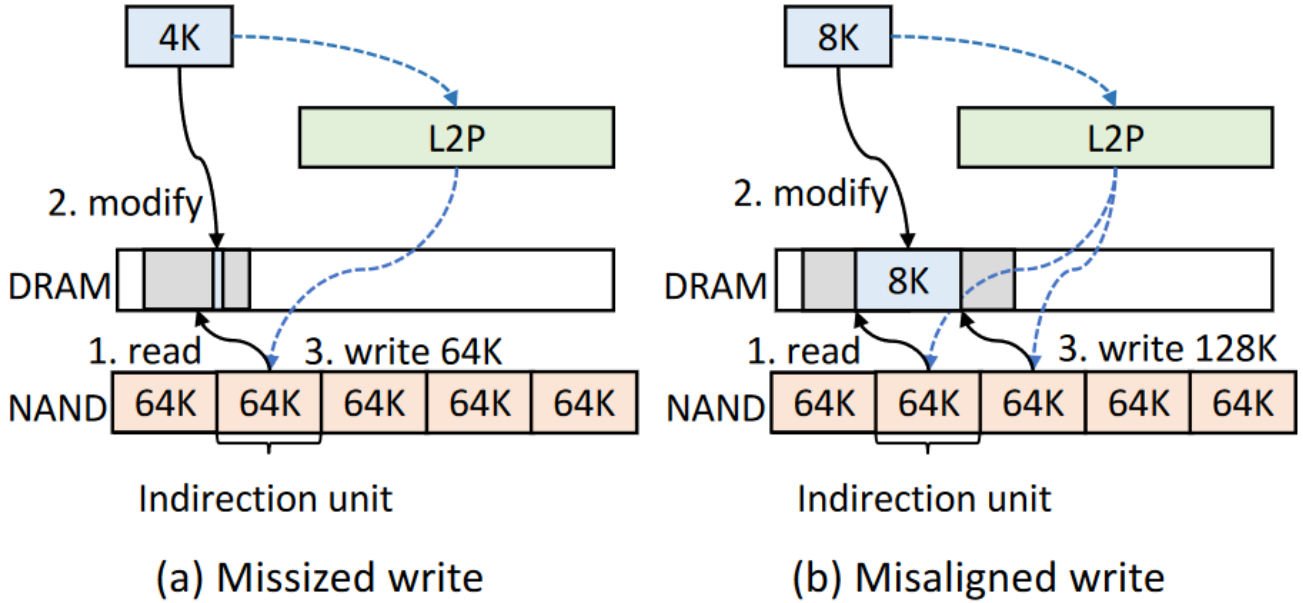
- 随机写，通用QLC-SSD和HDD性能相似（略有优势）
- 顺序写，通用QLC-SSD的性能差距落后很大（与I/O单元有关）

通用QLC-SSD无法满足写入性能上的SLO！

2.2 初始方案一(3/4)

3.原因分析：设备级写放大



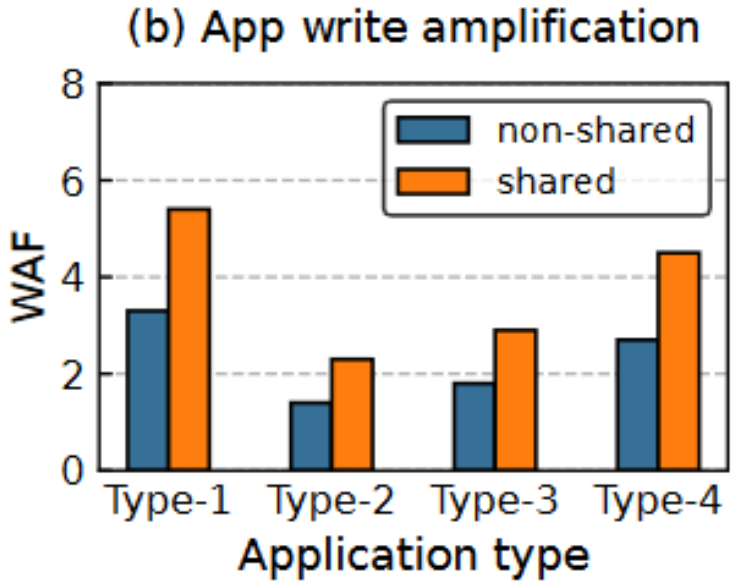


- OS页面为4KB，而通用QLC-SSD采用了更大粒度的页面，64KB
- 对实际本地盘跟踪，小于/等于8KB的小I/O在占据了60%以上
- 小I/O的“读-修改-写”会带来16x的写入放大（设备级写放大）

2.2 初始方案一(4/4)

3.原因分析：介质级写放大

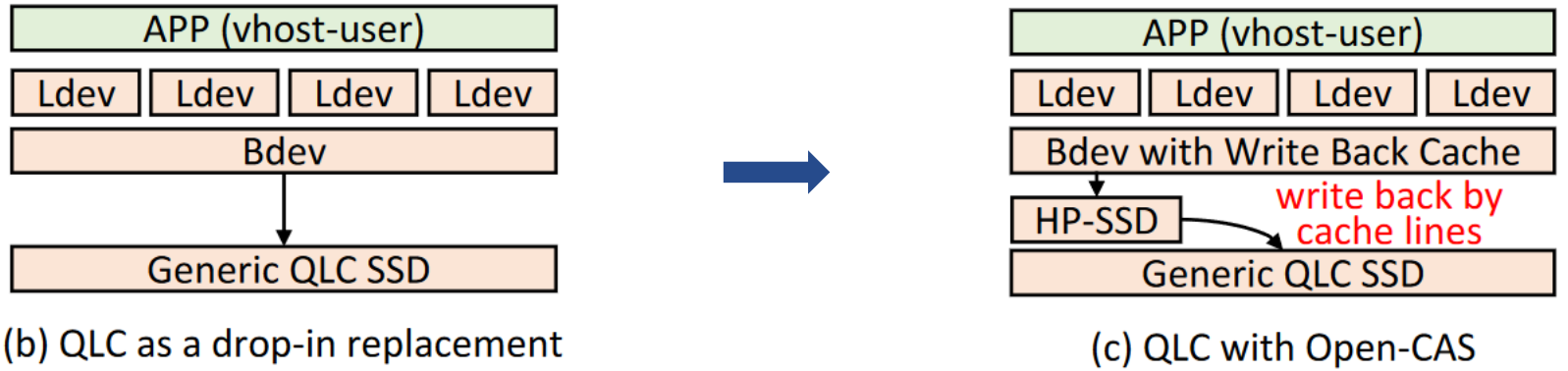
- QLC-SSD的擦除单元更大（如Intel P5316的67.4GB），写入寿命有限（15.36TB/日）
- 不同生命周期的数据混杂加剧GC时的写放大（介质级写放大，non-shared约2.3，shared约3.8）
- 为了满足99.9%用户的写入需求，QLC-SSD每日至少要写51.76TB的写入



	Logical Writes (TB)	NAND Writes (TB)
p50	1.23	25.07
p75	1.42	28.94
p90	1.58	32.20
p99	2.20	44.84
p999	2.54	51.76
p100	2.94	59.92

2.3 初始方案二(1/3)

1.方案2：添加写回缓存

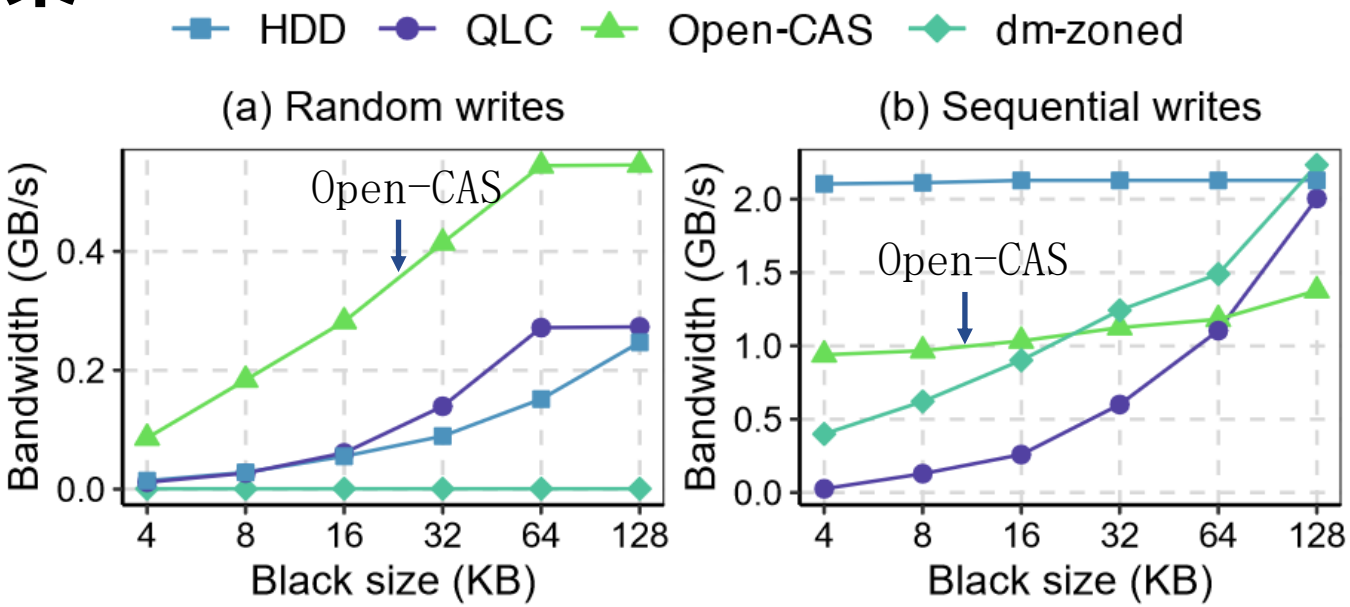


(b) QLC as a drop-in replacement

(c) QLC with Open-CAS

- 沿用通用型QLC-SSD做本地盘
- HP-SSD做写回缓存吸收小I/O，对齐通用型QLC-SSD的64KB粒度
- 通过Open-CAS构建了LRU缓存

2.测试结果

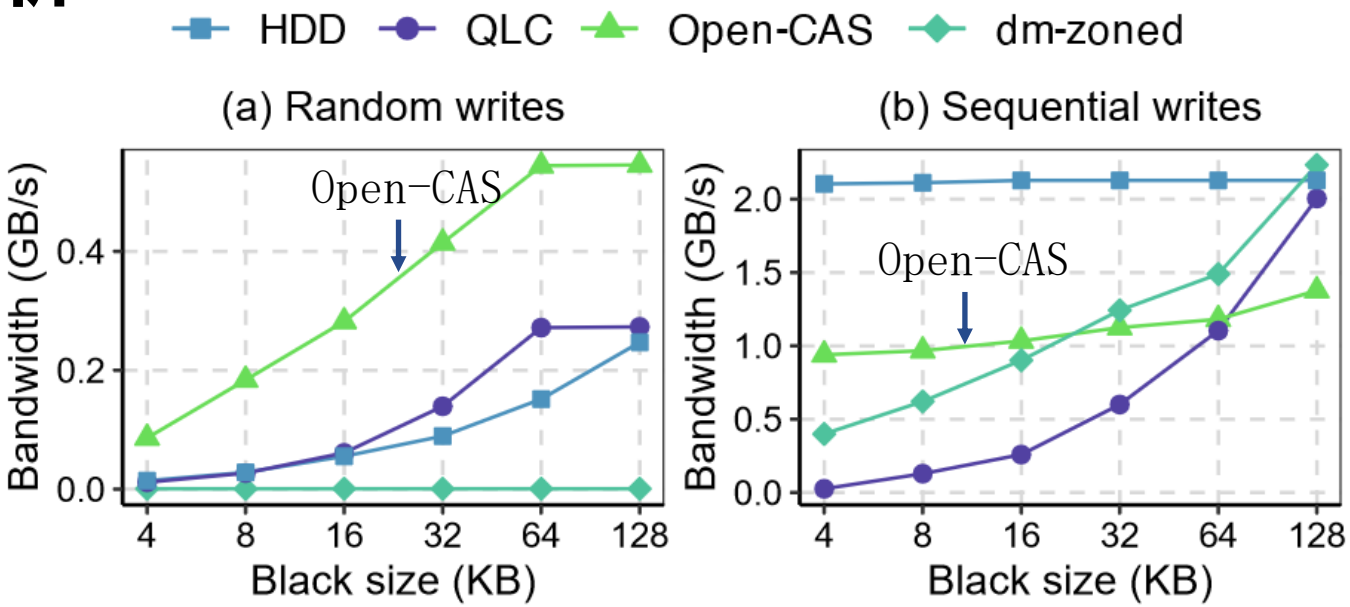


- 随机写，Open-CAS性能有明显的提升
- 顺序写，Open-CAS比通用QLC-SSD在小I/O上有所提升，但仍落后于HDD；大I/O上逐渐落后于通用QLC-SSD方案

Open-CAS仍无法满足写入性能上的SLO!

2.3 初始方案二(3/3)

3.原因分析

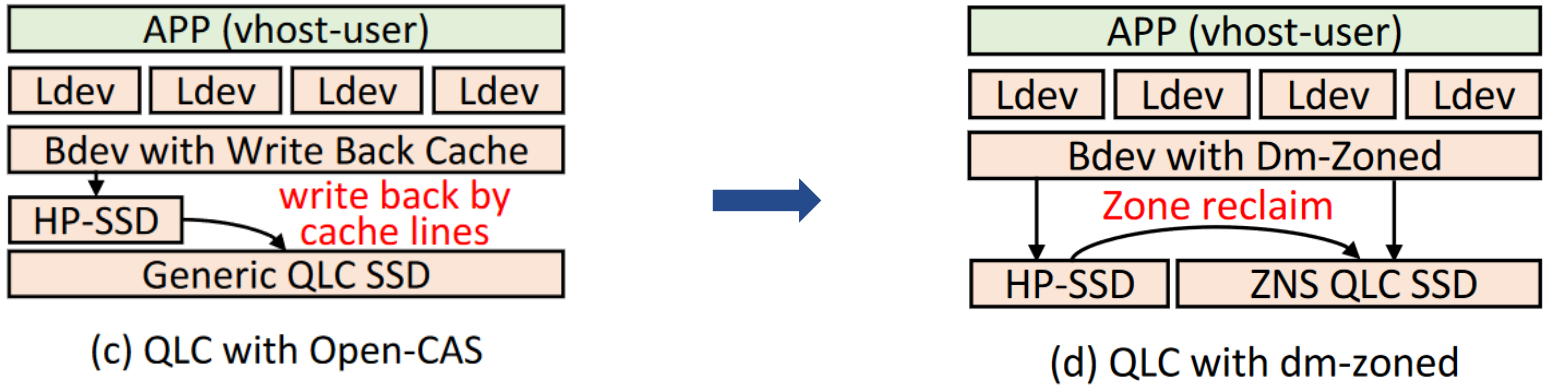


- Open-CAS聚集小I/O带来了一定的性能提升
- 但HP-SSD容量有限，需要定期刷回QLC-SSD，仍然会带来设备级写放大，并且有性能损失

I/O聚合没有改变大粒度（64KB）的本质

2.4 初始方案三(1/3)

1.方案3：dm-zoned ZNS方案



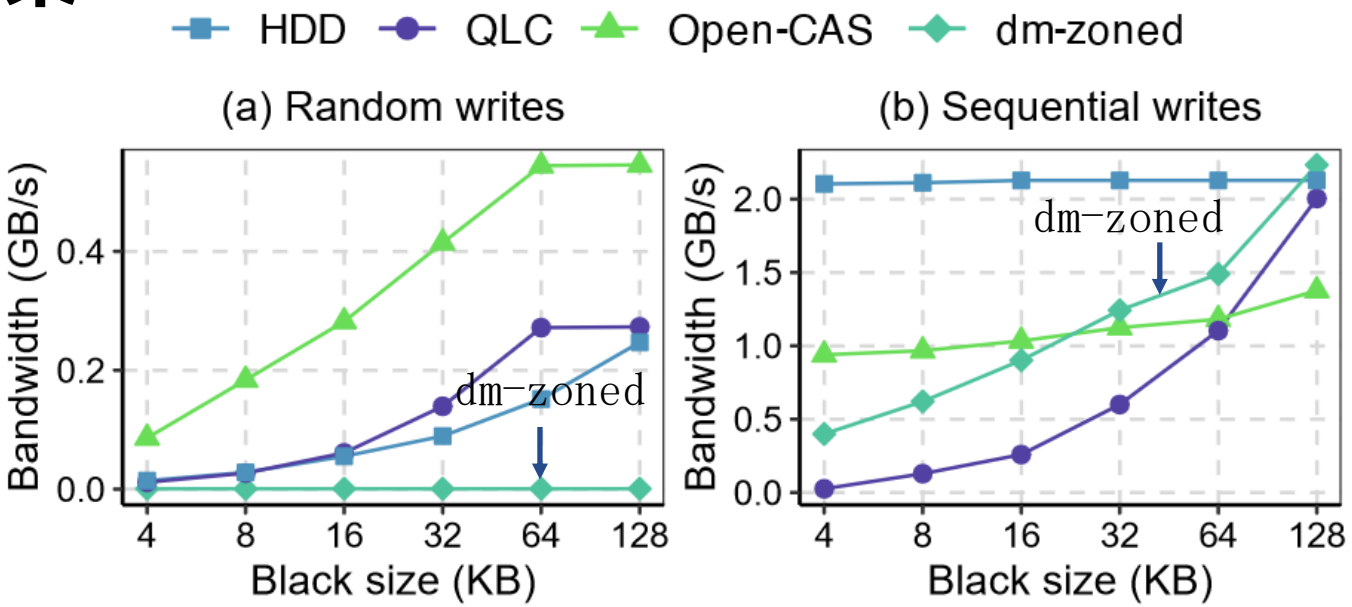
(c) QLC with Open-CAS　　(d) QLC with dm-zoned

ZNS方案打破大粒度（64KB）的限制

- 改用ZNS QLC-SSD做本地盘，Zone内只支持顺序写入
- HP-SSD做Zone缓存，Zone缓存支持随机写
- 通过dm-zoned构建上述方案，随机写路由到HP-SSD，顺序写则直接路由到ZNS QLC-SSD

2.4 初始方案三(2/3)

2.测试结果

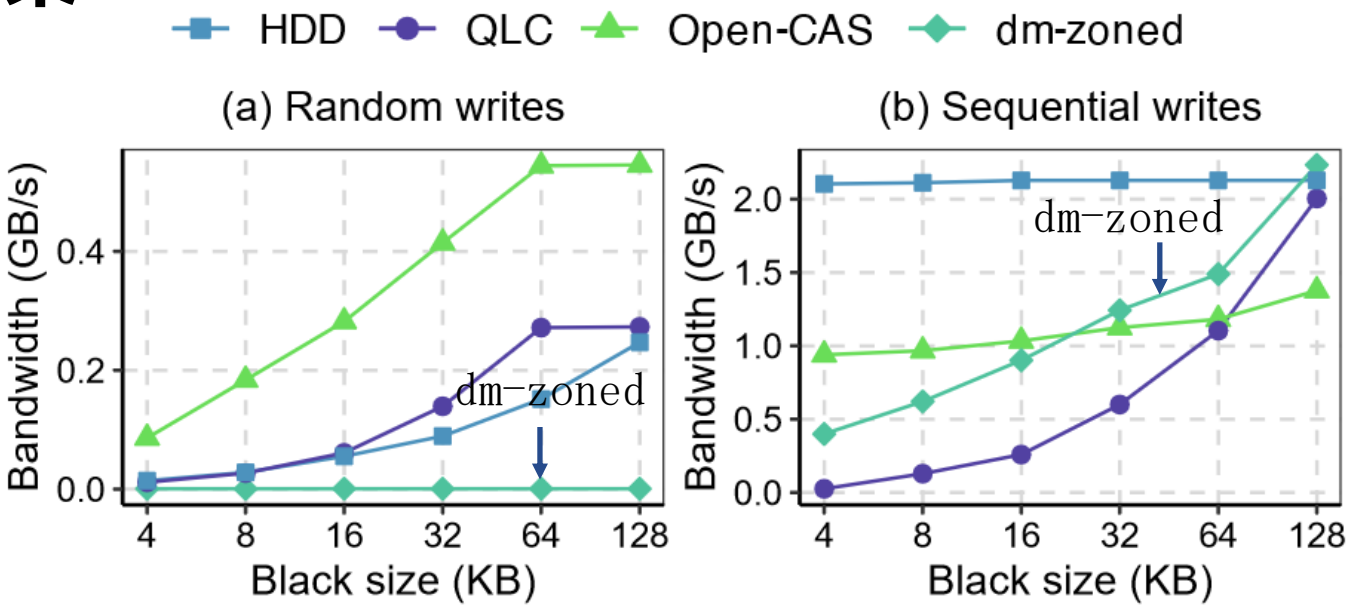


- 随机写，dm-zoned ZNS方案极为低效（10MB/s）
- 顺序写，dm-zoned ZNS方案整体比通用QLC-SSD方案有了一定提升，但仍然落后于HDD

dm-zoned ZNS方案无法满足写入性能上的SLO!

2.4 初始方案三(3/3)

2.测试结果



- 小的顺序写可被HP-SSD聚合一起写入，加速顺序写
- “一对一”的Zone缓存有限，会频繁填充、迁移与回收，几乎无法支持随机写（**随机写失效**）

顺序Zone无法支持随机写

2.5 关键观察

（1）通用QLC-SSD的两级写放大 ⟶ 通用QLC-SSD、Open-CAS方案的观察

- 设备级写放大：通用QLC-SSD上采用大粒度64KB管理使得4KB写失配
- 介质级写放大：QLC-SSD内部垃圾回收

（2）ZNS QLC-SSD的随机写失效 ⟶ Dm-zoned ZNS方案的观察

- 随机写失效：顺序Zone内只支持顺序写入

（3）ZNS QLC-SSD的机遇

- ZNS方案的Zone概念允许host侧做负载感知的冷热数据分类

3.1 设计概述

1.现有观察

- 设备写放大
- ZNS随机写失效

- 介质写放大
- ZNS数据分离

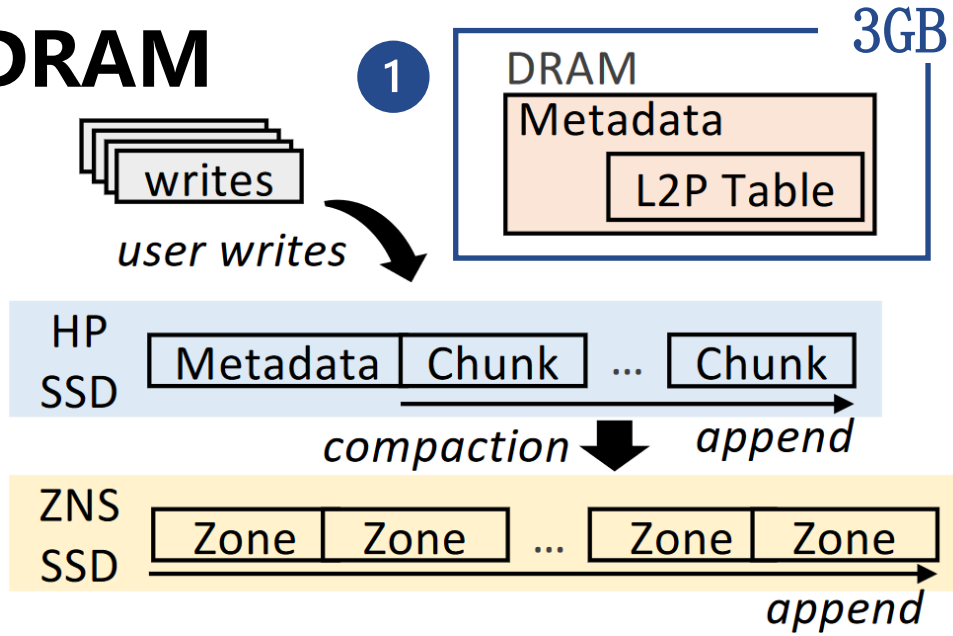
2.核心设计

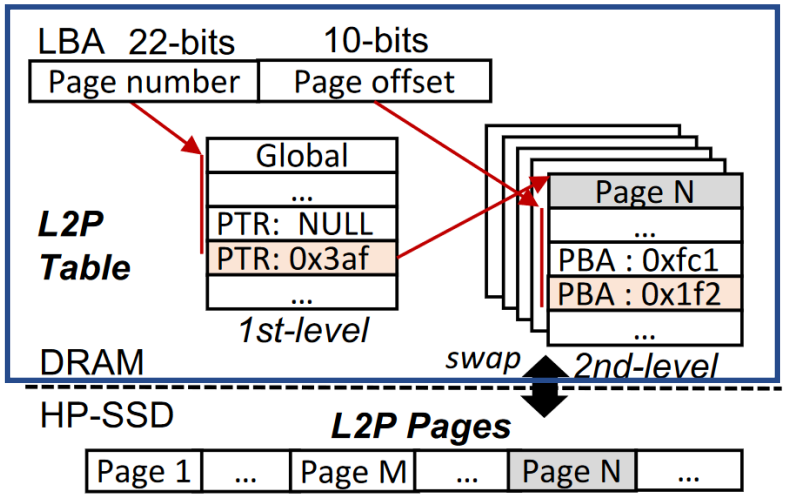
核心设计1：两级L2P表进行细粒度页面管理

核心设计2：缓冲数据的迁移、分离和整理

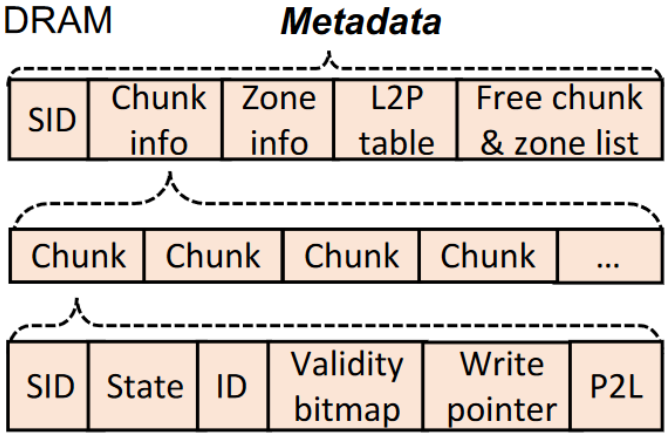
3.2 CSAL的三级存储架构(1/3)

1.DRAM





(a) Two-level L2P Table Hierarchy

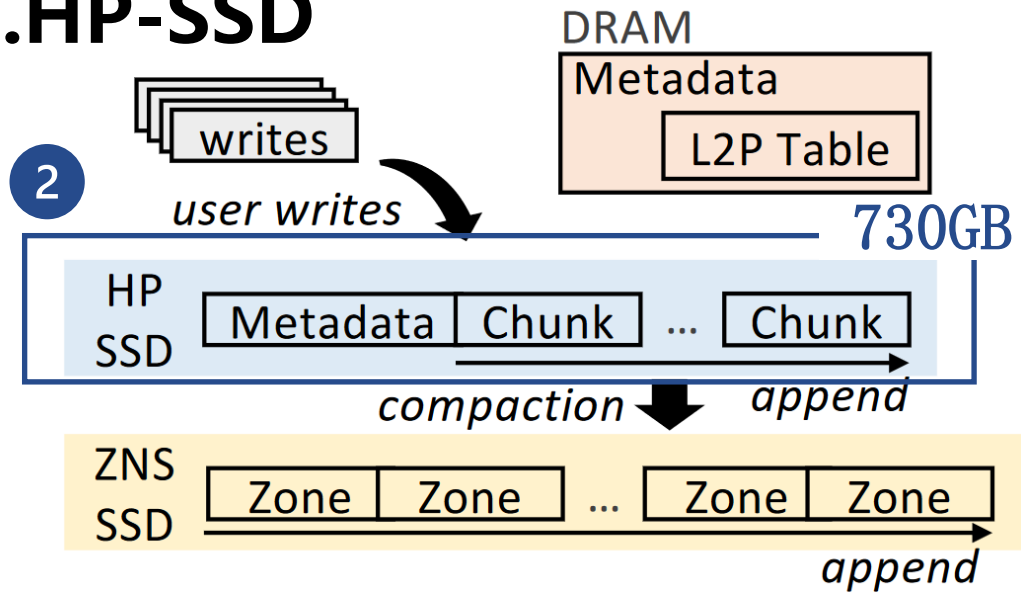


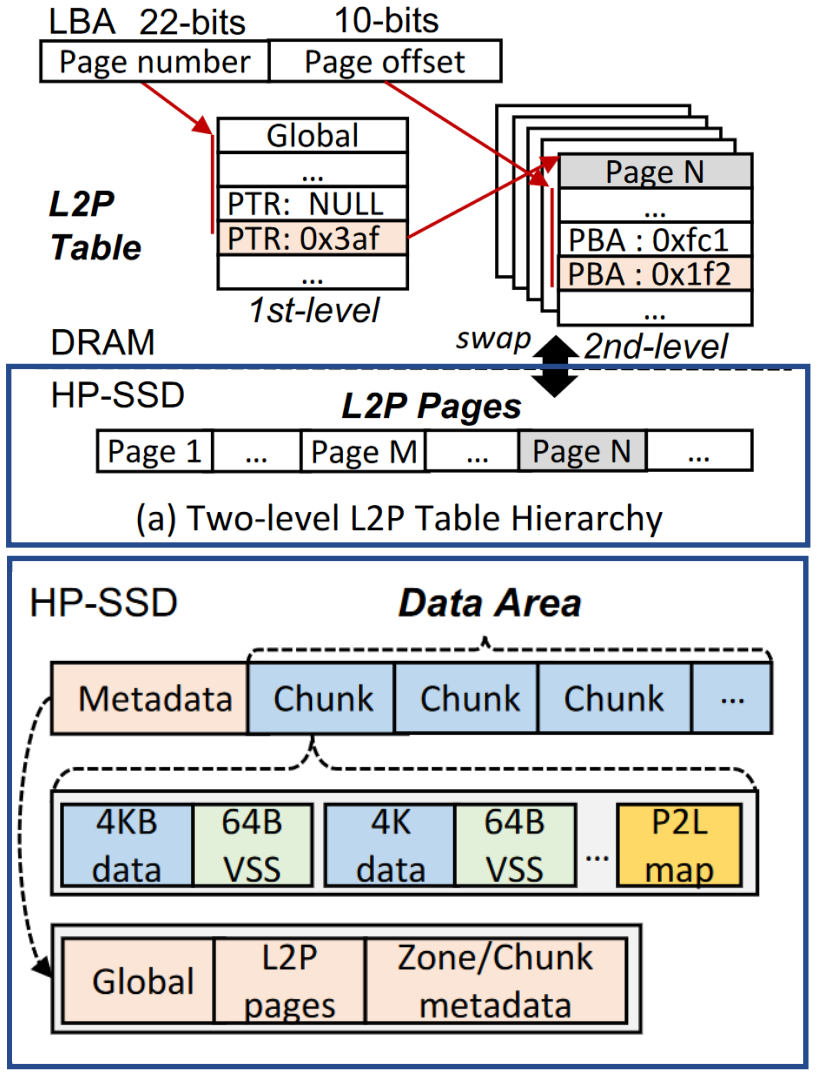
(b) Runtime Metadata

- 两级L2P表（4MB + 16GB），细粒度页面
- DRAM缓存**所有顶级表项**（4MB）和**部分二级表项**（2GB），L2P offload
- DRAM缓存全局必需的其他**元数据信息**（约120MB）

3.2 CSAL的三级存储架构(2/3)

2.HP-SSD

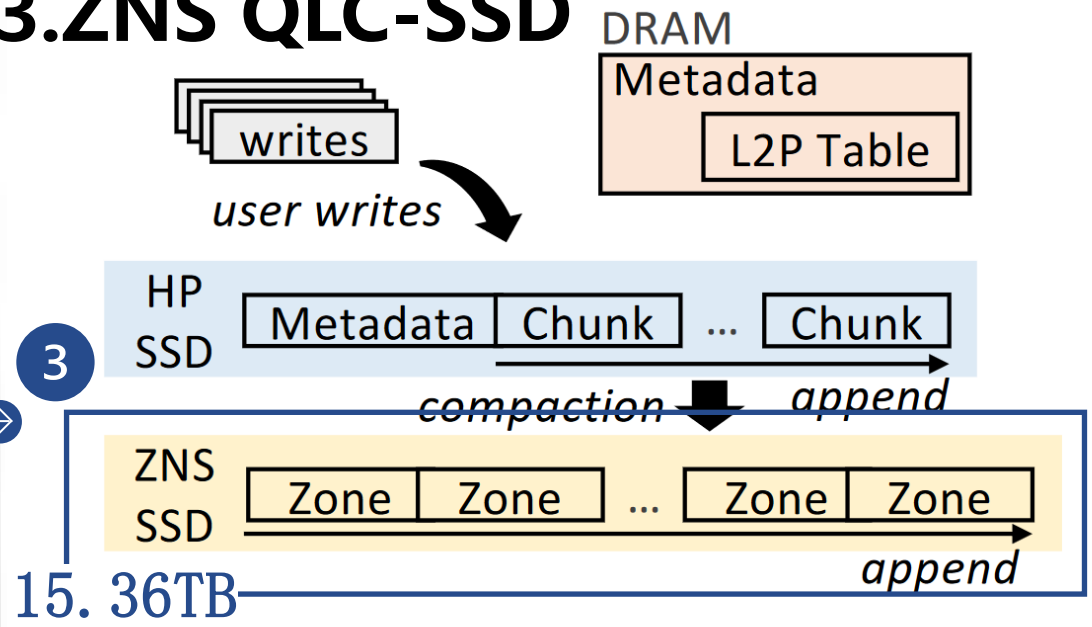


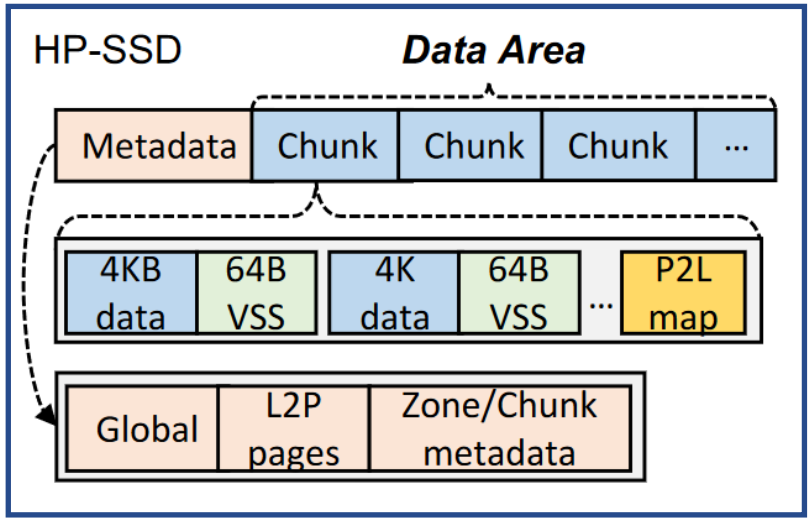


- HP-SSD存储**全部L2P表**（4MB + 16GB）
- HP-SSD存储统一的**Zone缓存（Chunk）**
- HP-SSD存储**全局元数据**持久化
- Chunk大小和I/O方式均和Zone保持对齐，即1GB/Chunk，顺序写；但Chunk数据混杂。

3.2 CSAL的三级存储架构(3/3)

3.ZNS QLC-SSD





- ZNS QLC-SSD全部用于数据区域，划分若干只能顺序写的Zone
- 用于接收从HP-SSD中迁移、分类和合并的数据
- Zone的布局和Chunk布局在磁盘上的布局一致

3.3 CSAL的数据流设计(1/2)

1.读-数据流

最优: DRAM + HP-SSD
最差: DRAM + HP-SSD + QLC-SSD

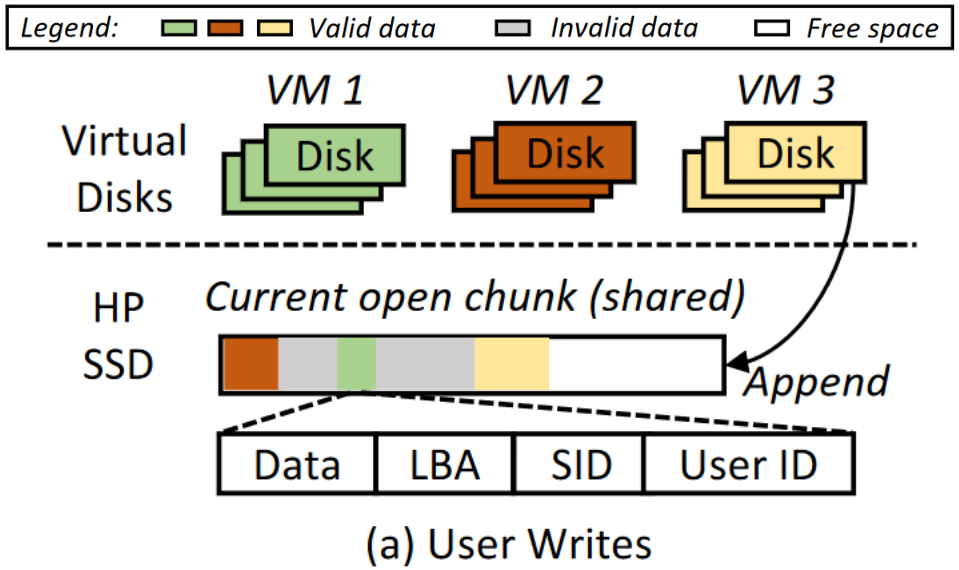
QLC-SSD的读性能不是SLO瓶颈

- 查询两级L2P表
- 根据页面物理地址，从HP-SSD/ZNS QLC-SSD读取

2.三种写-数据流 ⟷ 三级存储架构

(1)用户在线写

- 所有用户（VM）的所有写入（随机写或顺序写）都写到HP-SSD（共享）
- 打上标签并追加到打开的Chunk的尾部
- 更新有关的L2P表项



(a) User Writes

3.3 CSAL的数据流设计(2/2)

2.三种写-数据流 ⟷ 三级存储架构

(2)数据合并

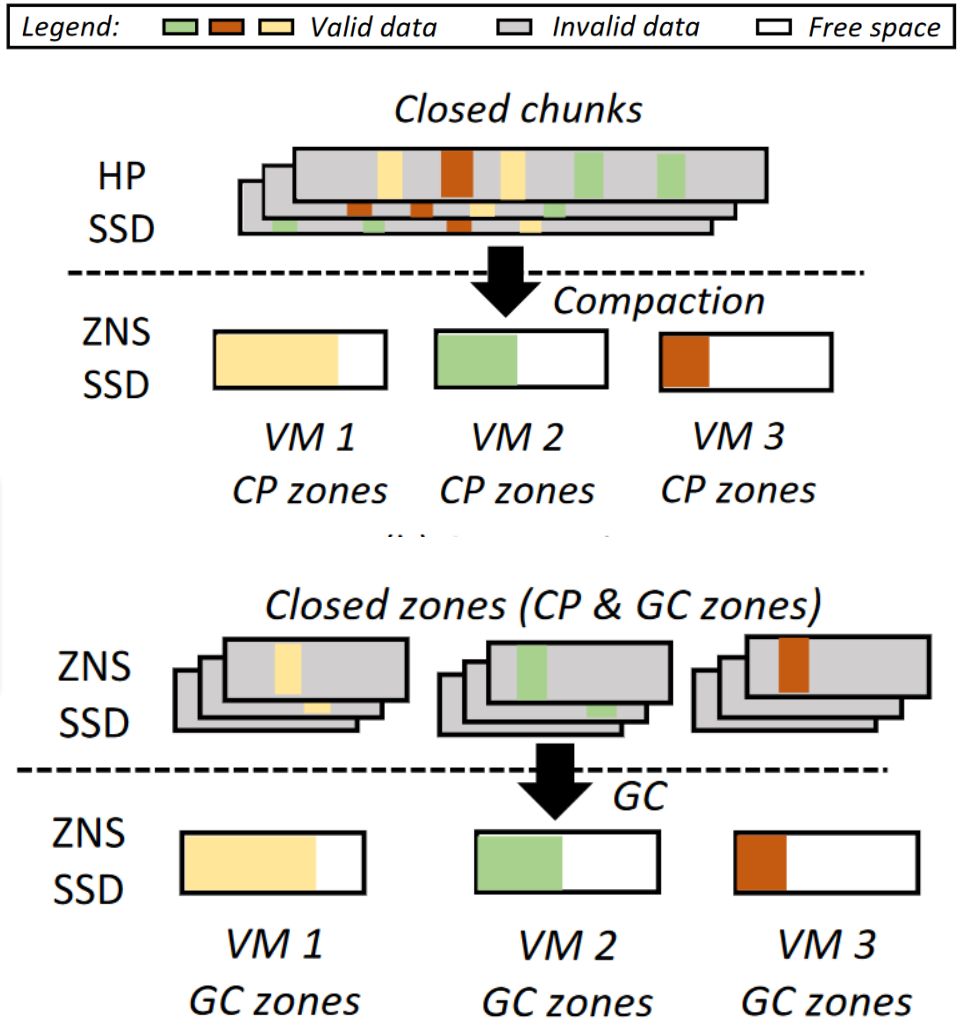
- 读取一个Chunk的有效块及UID到DRAM
- 根据UID做分类和合并，达到512KB迁移追加到用户对应的Zone
- 更新有关的L2P表项

Challenge

L2P表项的并发更新带来**LBA冲突**！

(3)垃圾回收

- 与数据合并相隔离，开一个独立的Zone进行垃圾回收
- 更新有关的L2P表项



3.4 CSAL的实现

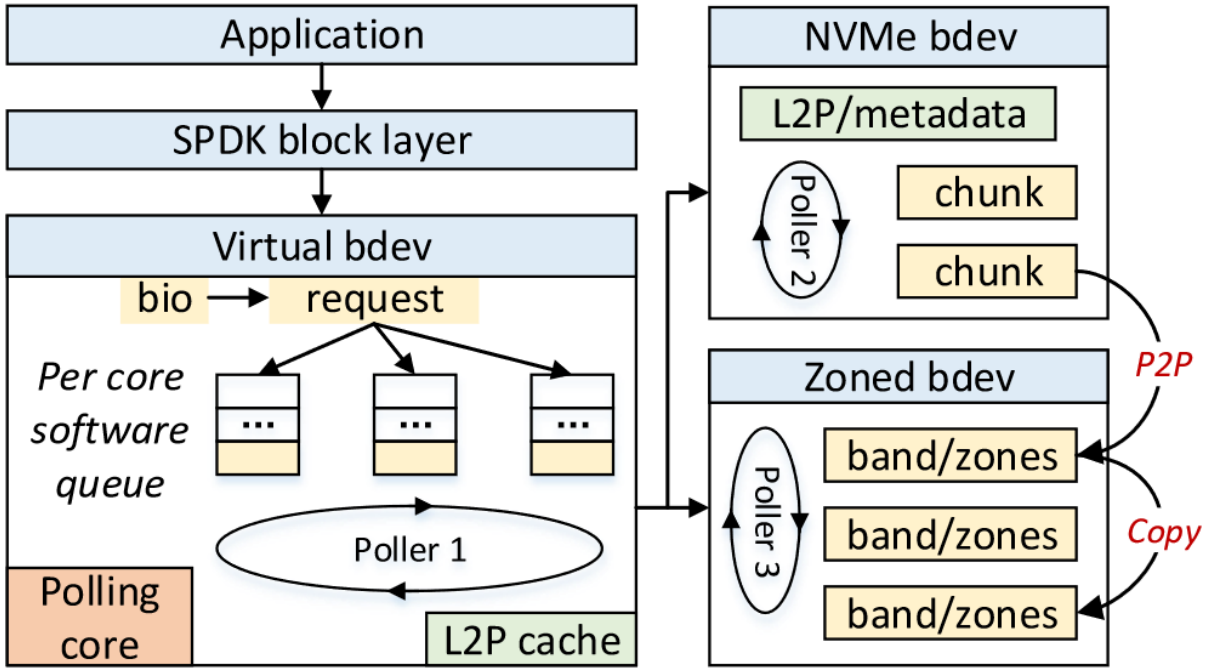
1.一致性和LBA冲突

- 方法：遍历全盘重构L2P表，每个Chunk打开时记录SID来解决LBA冲突
- 优化1：每个Zone/Chunk维护一个P2L表加速恢复
- 优化2：使用检查点加速恢复

服从最高的SID，SID相同服从最高PPA

2.实现细节

- 通过SPDK注册块设备
- 多个I/O队列
- 3个Poller用于在线写、合并和垃圾回收



4.1 测试环境

1.实验平台

	CPU	2x Intel 8369B @ 2.90GHz
	Memory	256GB DDR4 Memory
	High Performance SSD	1× Intel P5800X 800GB
CSAL-BLK	Non-ZNS QLC SSD	1× Intel P5316 15.36TB
CSAL-ZNS	ZNS TLC SSD	4× WD ZN540 4TB (ZNS)
	OS	Linux CentOS Kernel 4.19

通过SPDK模拟出ZNS

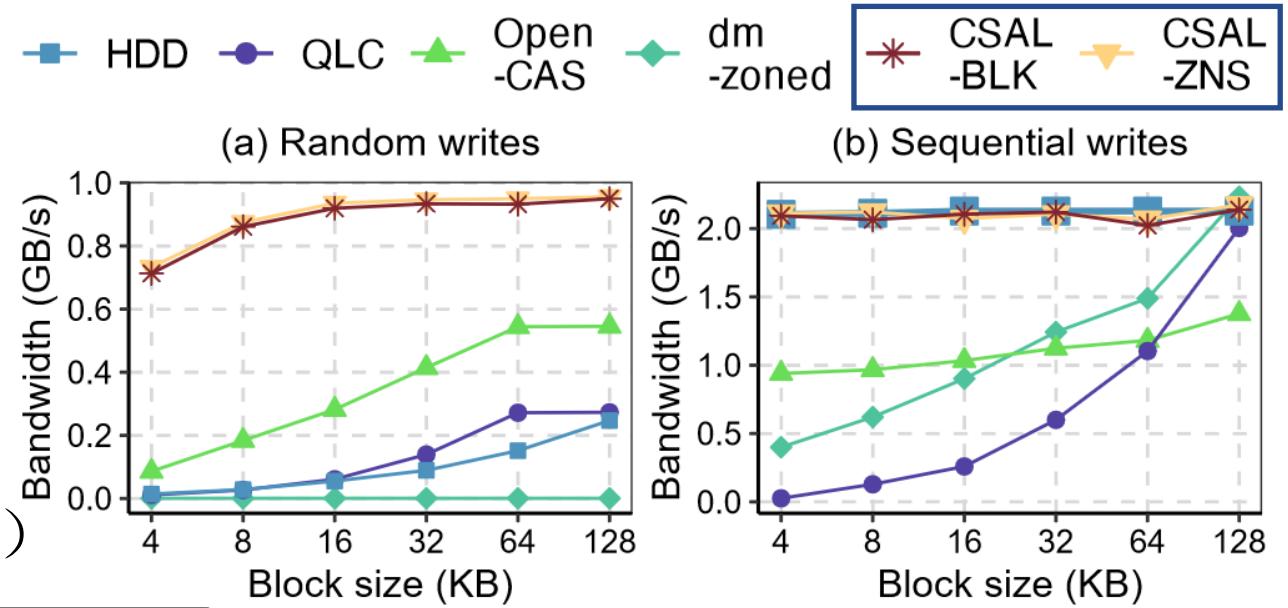
2.对比对象

- HDD
- 初步尝试1：通用QLC-SSD
- 初步尝试2：Open-CAS(通用QLC-SSD+写缓存HP-SSD)
- 初步尝试3：dm-zoned(ZNS QLC-SSD+写缓存HP-SSD)
- 本论文：CSAL-BLK和CSAL-ZNS

4.2 裸设备性能测试(1/2)

1.均匀写入

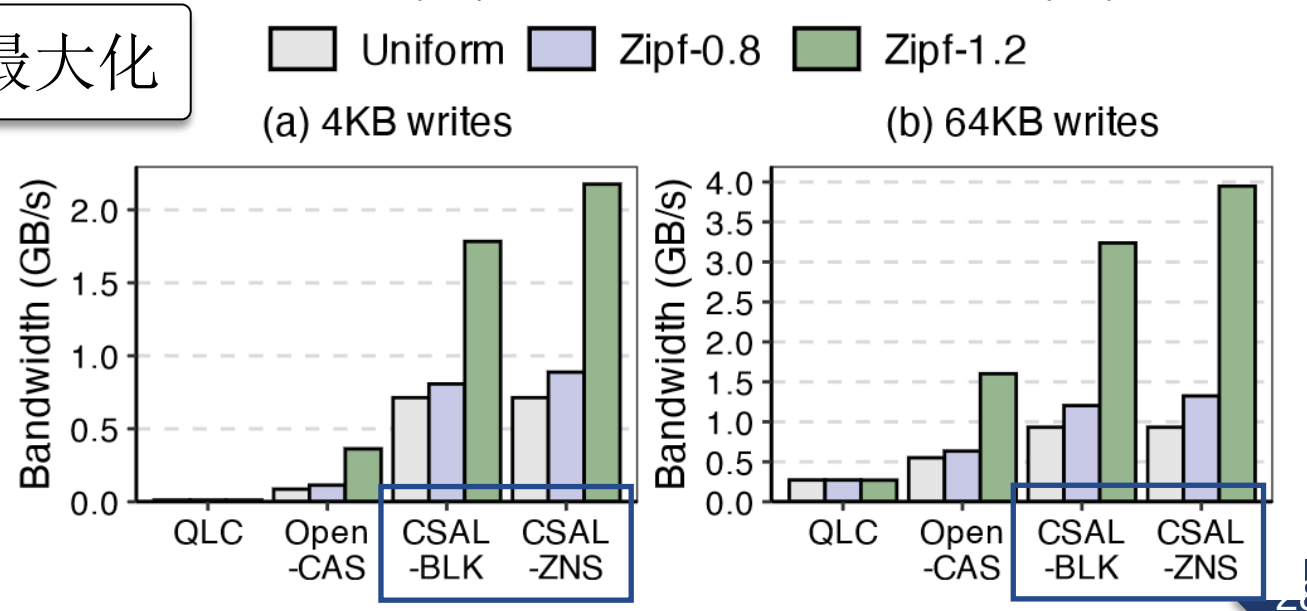
- 随机写，CSAL相比Open-CAS带来8.28x的提升
- 顺序写，CSAL相比dm-zoned带来5.24x的提升（追平HDD）



2.倾斜写入[1]

小I/O上优化最大化

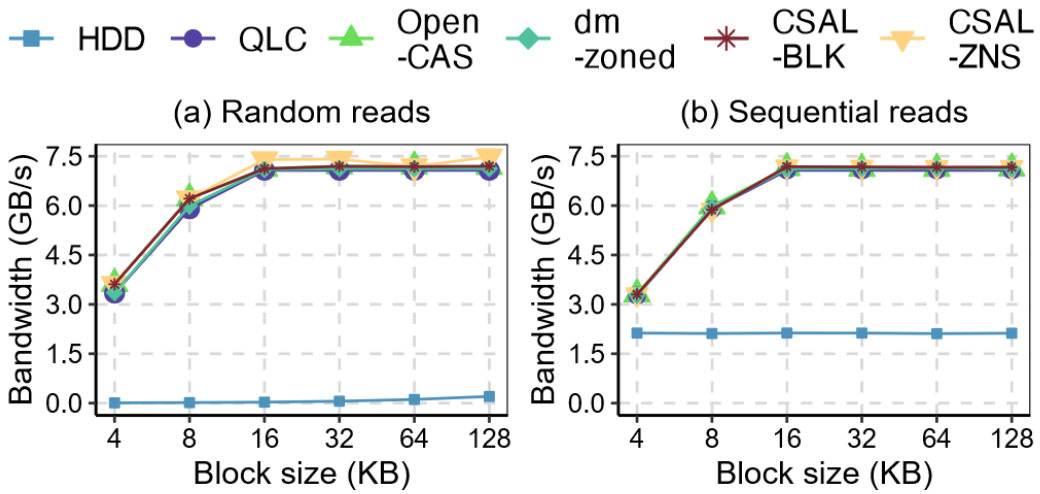
- 4KB和64KB写入，CSAL均维持差距显著的最高带宽

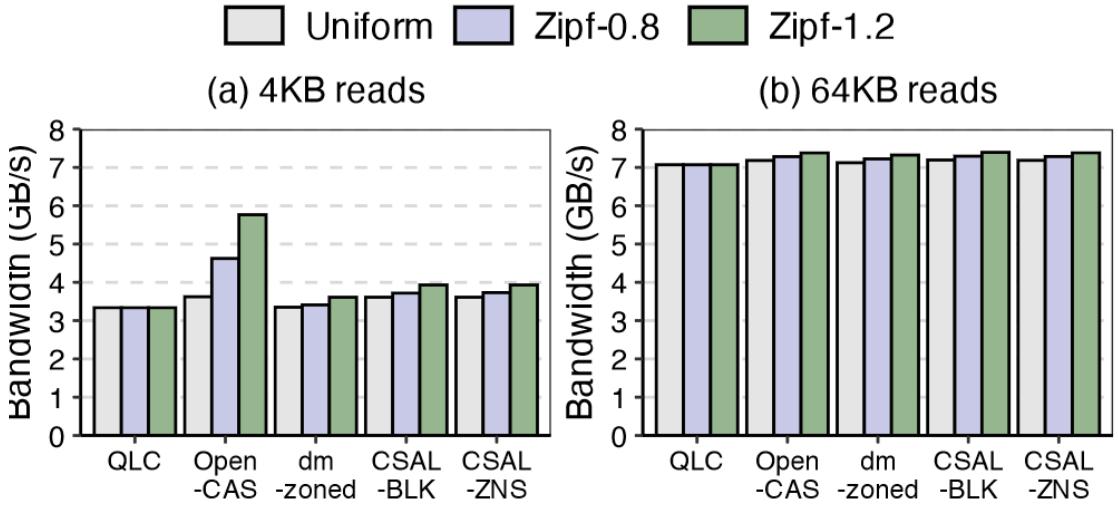


[1] dm-zoned的随机性能太差（10MB/s），不做展示

4.2 裸设备性能测试(1/2)

3.均匀读取/倾斜读取

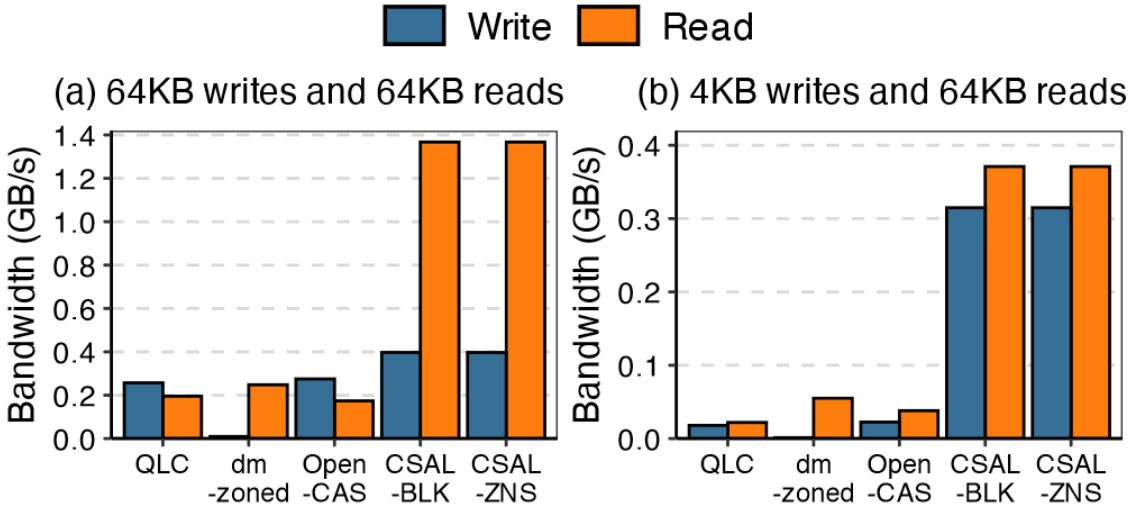




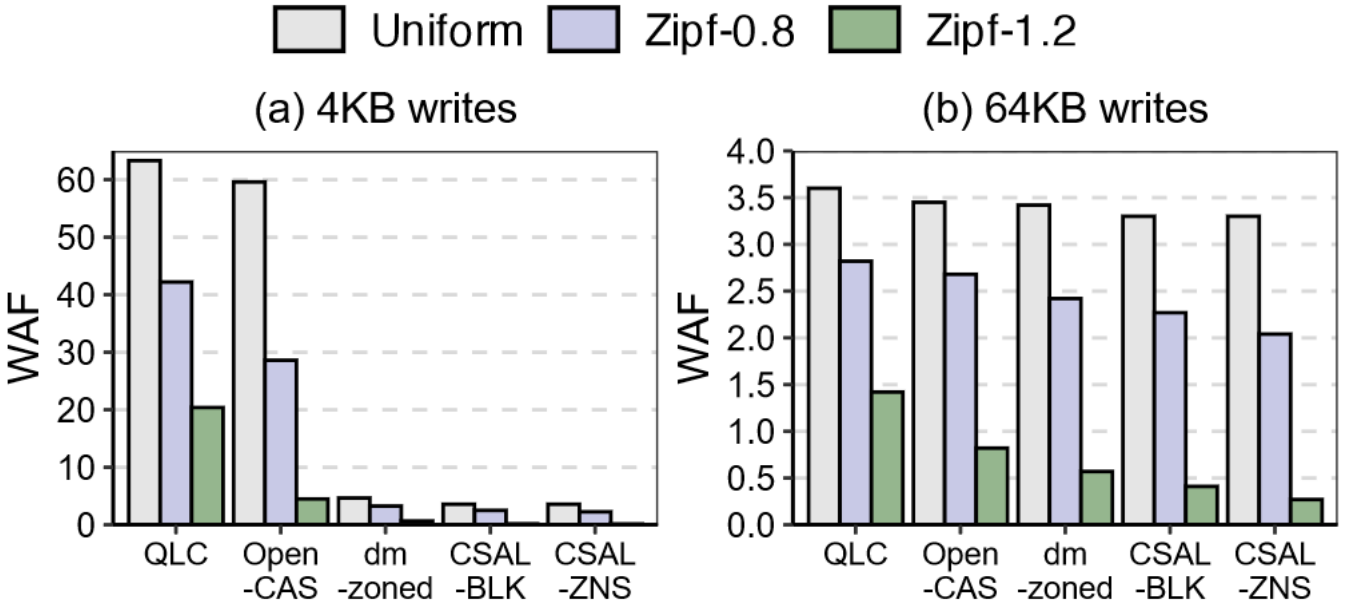
- 读取性能均保持最优，远超HDD，不是SLO性能瓶颈

4.随机读写混合

- CSAL均维持差距显著的最高读写带宽



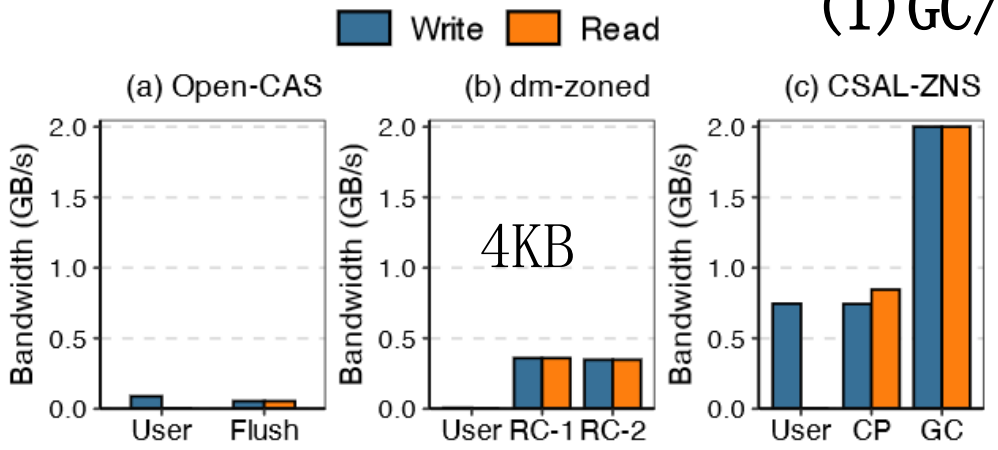
4.3 写放大评估

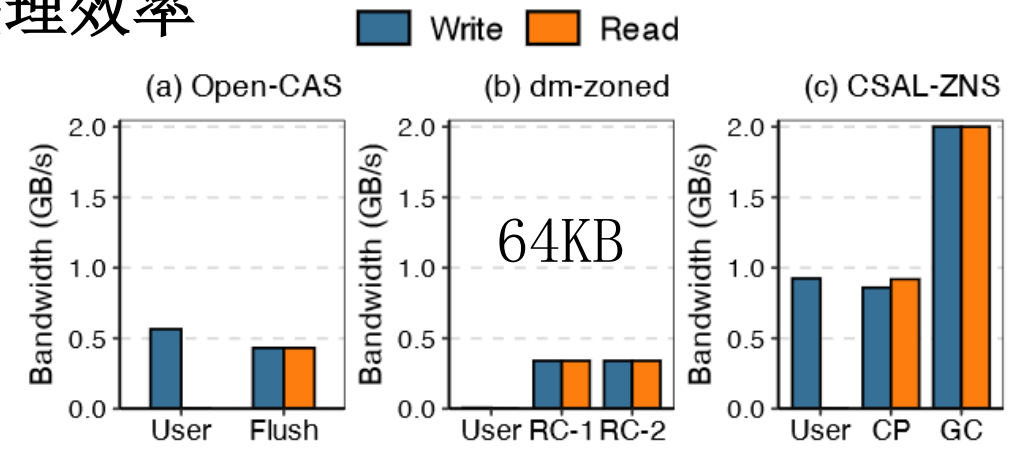


- 在4KB写入下，基于通用QLC-SSD的方案会面临严重写放大（64KB大粒度页面，两级写放大）
- CSAL的写放大很低（采用ZNS方案）
- 在64KB写入下，差距并不明显

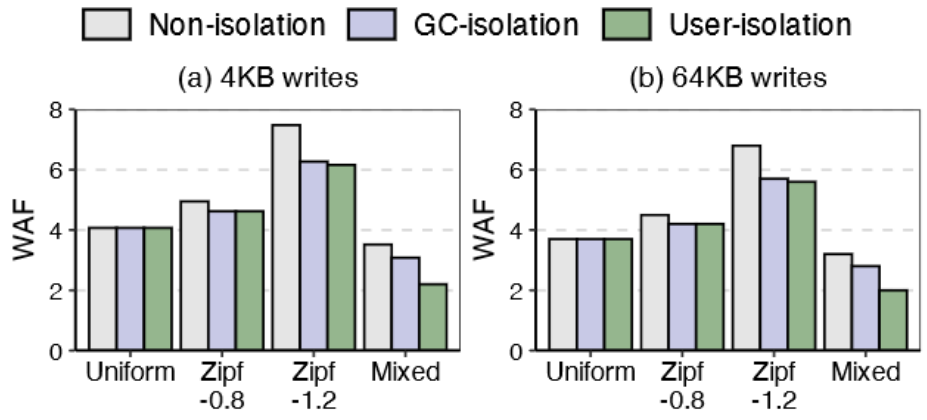
4.4 分解测试

(1) GC/整理效率

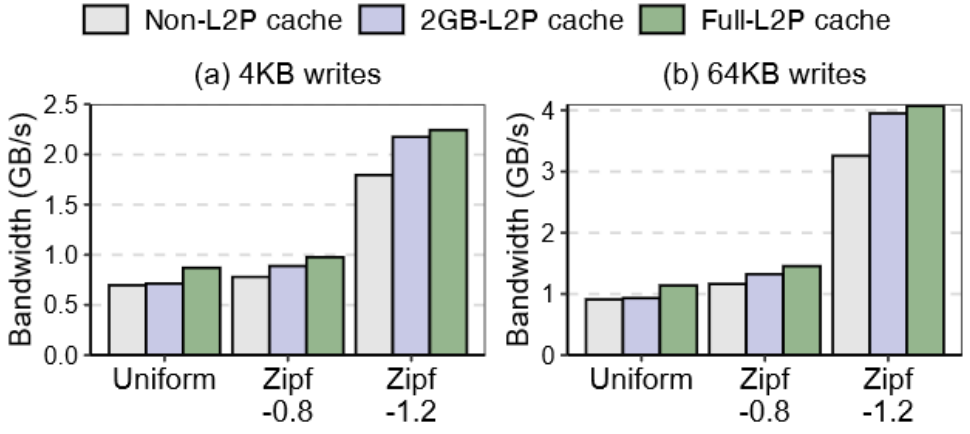




(2) 隔离WAF优化



(3) L2P缓存测试



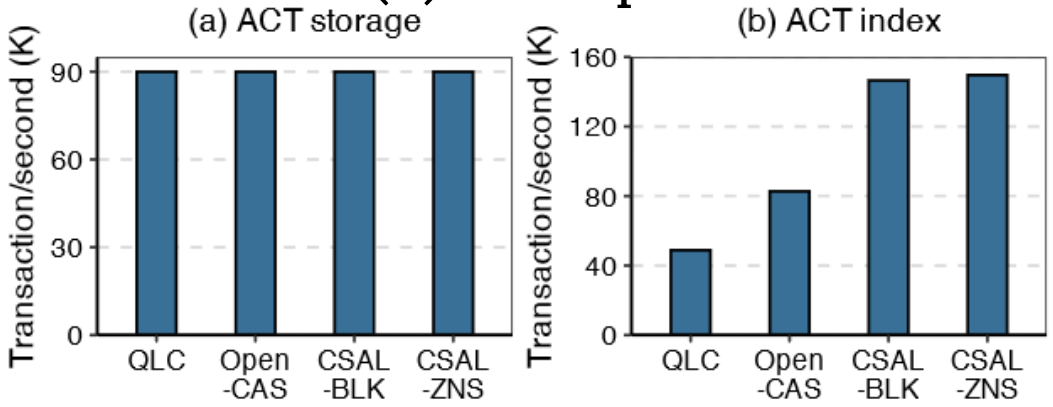
	Disk	P2L	L2P	Others	Total
Naive (s)	2621	-	-	1.64	2622.64
OPT-1 (s)	0.5	5.3	-	1.64	7.44
OPT-2 (s)	0.5	0.33	2.67	1.64	5.14

(4) 崩溃恢复测试

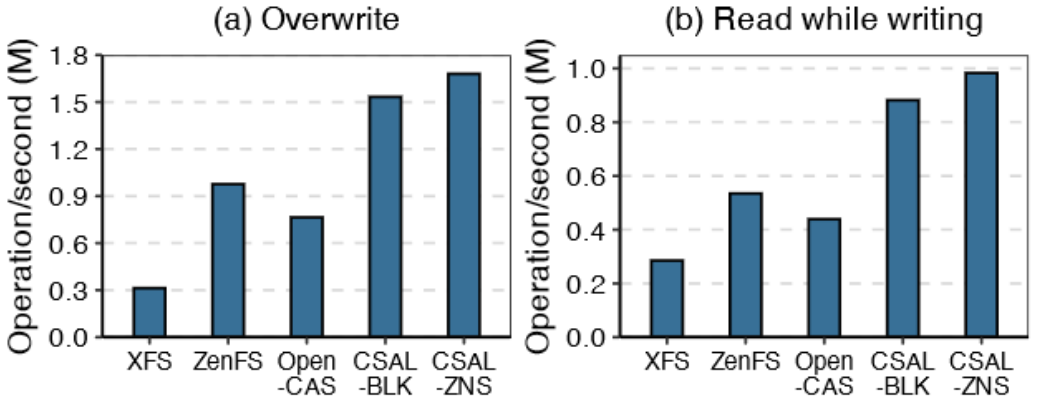
4.5 真实负载&资源开销

1.真实负载

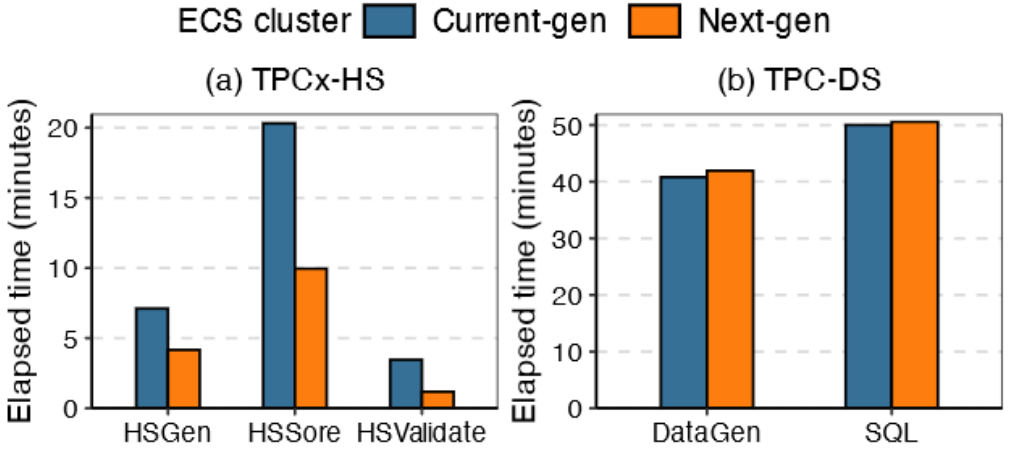
(1) Aerospike DB



(2) RocksDB



(3) 实际生产环境



2.内存开销[1]

CSAL：2.12GB

Open-CAS：1GB

dm-zoned：20MB

[1] 通用QLC-SSD设备内部会有内存（约1G）

论文总结(1/3)

1.问题： 存储资源无法复用，成为提升云服务提升部署密度的**瓶颈**

2.挑战

最理想替代方案QLC-SSD的初步尝试，**都无法满足预期的写入SLO**

- 通用QLC-SSD方案的**两级写放大**（设备级和介质级）
- 现有ZNS方案的**随机写失效**

3.本文方案

- 采用**基于ZNS的多级存储架构**卸载设备内L2P，应对设备级写放大
- 采用**两级L2P表**进行**细粒度页面**管理，支持随机写
- 缓冲数据的**迁移、分离和整理**，减少介质级写放大

论文总结(2/3)

4.本文优点&启发

（1）写作

- Attempts加Lesson Learned的写作方式作为观察与动机
- 将纯粹的存储问题，套在宏大的云场景故事下，逻辑自洽

（2）工作

- 通用QLC设备上的大页趋势

1.Zone的划分
2.Zone内物理块顺序追加

- ZNS SSD设备是很灵活的，打破固有思维，拓展Host对ZNS管理的了解
  - **面向文件抽象：ZNS文件系统**。通常采用log-structure的方式管理所有文件，减少甚至避免全部随机写。
  - **面向块设备抽象：拓展的ZNS本地盘**。兼具随机写和顺序写。

论文总结(3/3)

5.本文不足&疑惑

- 本文的关键挑战，LBA冲突**并不挑战**；围绕其的诸多设计点是**很简单且并非首创**的，如SID，P2L等。
- 本文创新点**信息零碎**且**不直接**，需要读者反复揣摩总结
- 是否应该**考虑成本**？多级存储架构引入了异构存储，如HP-SSD。没有成本的对比个人觉得并不公平。
- 是否存在4KB的通用QLC设备？该设备和本文方案对比如何？

为什么细粒度？为什么用L2P？为什么采用两级L2P？为什么HP-SSD采用log-structure？

6.工作展望

- 代码已经纳入SPDK（开源）
- 现在的数据分类方案**基于VMs**，效果有限，探索其他**数据分类方式**

EuroSys'24

哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

CSAL: the Next-Gen Local Disks for the Cloud

Yanbo Zhou[1], Erci Xu*[1], Li Zhang[1], Kapil Karkra[2], Mariusz Barczak[2], Wayne Gao[2], Wojciech Malikowski[2], Mateusz Kozlowski[2], Łukasz Łasek[2], Ruiming Lu[1], Feng Yang[1], Lilong Huang[1], Xiaolu Zhang[1], Keqiang Niu[1], Jiaji Zhu[1], Jiesheng Wu[1]

[1]Alibaba Group; [2]Solidigm

CSAL：下一代云本地盘

谢谢大家

请老师和同学批评指正

- 论文汇报：杨大荣
- 指导老师：夏文教授