存储稳定性测试与数据一致性校验工具和系统

DISK/MEMORY stability testing & DATA consistency verifying tools and system

--YOUPLUS(微信：YOUPLUS2022)

Email: zhang_youjia@126.com

个人简介：

原深交所上市科技公司，员工持股平台合伙人之一(9000+员工，100+合伙人)，资深技术专家；

原云计算公司，员工持股平台合伙人之一，深圳研发中心负责人、虚拟化团队负责人，资深架构师、技术专家；

云计算公司总工(首席工程师)

16+年研发经验：

3年网络安全；

13+年虚拟化、内核、存储；

目前仍在研发一线；

工欲善其事，必先利其器！

遥遥领先！

➢ 使用过LBA工具及自动化测试系统之前：

Q: fio和vdbench等工具已经有了verify功能，为什么还要自已写一个？
A: fio和vdbench等工具，很多场景测试不了，很多场景测试发现不了问题。

Q: LBA工具的数据校验功能与fio和vdbench等工具的verify功能相比较如何？
A: fio和vdbench等工具能测试出来的问题，LBA工具都能测试出来；
LBA工具能测试出来的很多问题，fio和vdbench等工具却不一定能测试出来；

➢ 使用过LBA工具及自动化测试系统之后：

-- fio,vdbench等工具的verify功能不香了！
-- LBA工具的实现原理和使用方法？
-- 存储稳定性测试和数据一致性校验，LBA工具及自动化测试系统功能丰富、完善并且强大！
（十多年，经历上千例存储稳定性和数据一致性问题，不断总结经验和持续改进后的产品）

目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

分布式存储三大难：什么工具可以高效地测试并发现问题？

同样经过两年多的研发，华为存储在2019年正式发布了新一代智能分布式存储-OceanStor分布式存储。华为北京、上海、深圳、成都等地的存储研发团队跨地域紧密协同，设计出一套全新的存储架构，进行了数千万行代码开发，并先后进行了近十个版本的迭代，反复打磨、优化的成果。

华为OceanStor分布式存储产品不仅实现分布式存储性能全球第一，单节点性能高端16.8万 IOPS和1毫秒以内时延，在业界标准SPC-1测试中排名第一；还首次在一套存储中实现了同时支持块、文件、对象、HDFS协议；并且，率先将人工智能技术融入到存储全生命周期管理。

在OceanStor分布式存储为用户数据中心提供一套简单却功能强大的存储背后，其实是研发团队反复打磨与优化的成果。比如，为解决性能波动的“隐形瓶颈”，华为存储研发团队动用了全球研发体系内的操作系统调度专家、网络专家，让性能波动彻底解决，实现了非ARM节点14万IOPS、ARM节点16.8万IOPS的稳定性能。

分布式存储有三大难，其一是数据不一致情况如何解决，由于数据打散在各个节点上，很容易出现数据不一致的情况，尤其是在高性能、大并发的数据读写情况下如何保持数据一致性。这需要华为存储在并发机制上做到精准无比。为此，华为存储研发团队研发出一套新的日志机制与工具，在海量数据中去追踪和查看IO异常，实现了上千个节点不同场景下的数据不一致情况检验。

其二是分布式存储功能与性能如何平衡，像重复数据删除功能对存储性能影响很大。为此，华为存储研发团队专门开发出一种加权算法，让重删功能实现自适应调整，为用户减去了过去需要不断手动配置的复杂性，呈现出极为简单的易用性。

最后则是如何防范数据丢失。通常，分布式存储所承载的应用场景是多样化和复杂化，这也直接促使了数据丢失潜在因素的增多。针对这种情况OceanStor分布式存储在只有64个字节的可靠性校验的空间容纳了30多个场景的预防方案，每个预防方案的“植入”需要精准到每个比特位，堪称手术刀般精准。

华为存储发展史：从筚路蓝缕到星辰大海

https://mp.ofweek.com/cloud/a545693321986

软件定义存储，看这一篇就够了！

https://www.sohu.com/a/397070625_505795

4.技术门槛

存储圈流传这样一个说法，搞存储是一个危险系数很高的职业。

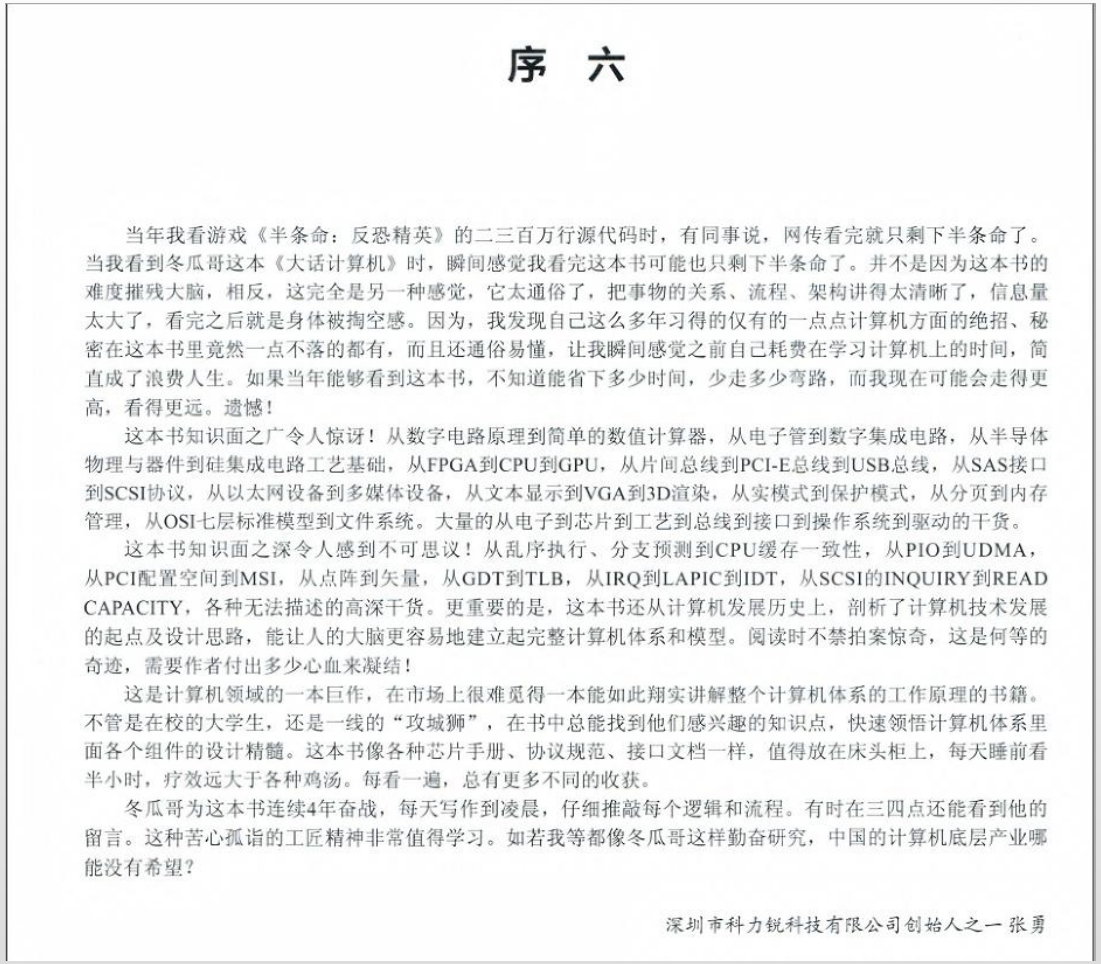
为啥呢？计算出问题通常都是局部单机故障，重启试试大概率就能解决，而存储一旦出问题就是全局系统故障，搞不好就成背锅侠。存储作为IT基础设施的底座，重要性不言而喻，谁整谁知道。

存储圈内人士总是对存储有无穷的敬畏，越是大牛越是如此。而存储圈外人士往往低估存储的门槛，认为只要投钱花时间就能搞出牛逼的存储产品。这里面直接忽略了科学理论、工程技术的系统复杂性和人的决定性因素。如果是这样，哪有什么卡脖子关键技术一说。

存储系统软件是一个复杂的系统软件工程，需要严谨的理论架构和工程化来保证数据安全性和系统稳定性，存储的底线和红线是不允许丢失数据。存储算法理论、系统架构、硬件结构、操作系统、软件工程等各个环节都非常复杂。系统复杂性决定了存储研发不可能是一件容易的事，对于存储新产品技术要在无人区不断摸索和试错，对理论创新、系统架构、工程化能力要求很高，这些都直接决定了存储的极高门槛。

LBA工具介绍

- 雏形源于SF科技前CTO、科力锐科技(http://www.clerware.com/)创始人张勇---给冬瓜哥《大话计算机》写序的前同事(当时办公位前后座，得到过他的很多帮助)，编写的1000行左右C代码的windows程序；
- 当时的程序功能还不完善、存在不少问题，使用场景受限，数据可视化的功能少，当使用工具排查存储数据问题时，总是要换算十六进制的数据等，非常不方便；
- 前同事2015年离职后代码无人维护、工具逐渐无人使用。



序 六

当年我看游戏《半条命：反恐精英》的二三百万行源代码时，有同事说，网传看完就只剩下半条命了。当我看到冬瓜哥这本《大话计算机》时，瞬间感觉我看完这本书可能也只剩下半条命了。并不是因为这本书的难度摧残大脑，相反，这完全是另一种感觉，它太通俗了，把事物的关系、流程、架构讲得太清晰了，信息量太大了，看完之后就是身体被掏空感。因为，我发现自己这么多年习得的仅有的一点点计算机方面的绝招、秘密在这本书里竟然一点不落的都有，而且还通俗易懂，让我瞬间感觉之前自己耗费在学习计算机上的时间，简直成了浪费人生。如果当年能够看到这本书，不知道能省下多少时间，少走多少弯路，而我现在可能会走得更高，看得更远。遗憾！

这本书知识面之广令人惊讶！从数字电路原理到简单的数值计算器，从电子管到数字集成电路，从半导体物理与器件到硅集成电路工艺基础，从FPGA到CPU到GPU，从片间总线到PCI-E总线到USB总线，从SAS接口到SCSI协议，从以太网设备到多媒体设备，从文本显示到VGA到3D渲染，从实模式到保护模式，从分页到内存管理，从OSI七层标准模型到文件系统。大量的从电子到芯片到工艺到总线到接口到操作系统到驱动的干货。

这本书知识面之深令人感到不可思议！从乱序执行、分支预测到CPU缓存一致性，从PIO到UDMA，从PCI配置空间到MSI，从点阵到矢量，从GDT到TLB，从IRQ到LAPIC到IDT，从SCSI的INQUIRY到READ CAPACITY，各种无法描述的高深干货。更重要的是，这本书还从计算机发展历史上，剖析了计算机技术发展的起点及设计思路，能让人的大脑更容易地建立起完整计算机体系和模型。阅读时不禁拍案惊奇，这是何等的奇迹，需要作者付出多少心血来凝结！

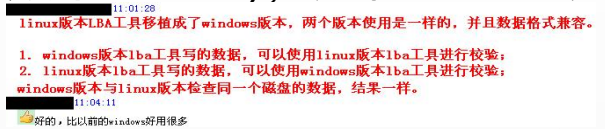
这是计算机领域的一本巨作，在市场上很难觅得一本能如此翔实讲解整个计算机体系的工作原理的书籍。不管是在校的大学生，还是一线的“攻城狮”，在书中总能找到他们感兴趣的知识点，快速领悟计算机体系里面各个组件的设计精髓。这本书像各种芯片手册、协议规范、接口文档一样，值得放在床头柜上，每天睡前看半小时，疗效远大于各种鸡汤。每看一遍，总有更多不同的收获。

冬瓜哥为这本书连续4年奋战，每天写作到凌晨，仔细推敲每个逻辑和流程。有时在三四点还能看到他的留言。这种苦心孤诣的工匠精神非常值得学习。如若我等都像冬瓜哥这样勤奋研究，中国的计算机底层产业哪能没有希望？

深圳市科力锐科技有限公司创始人之一 张勇

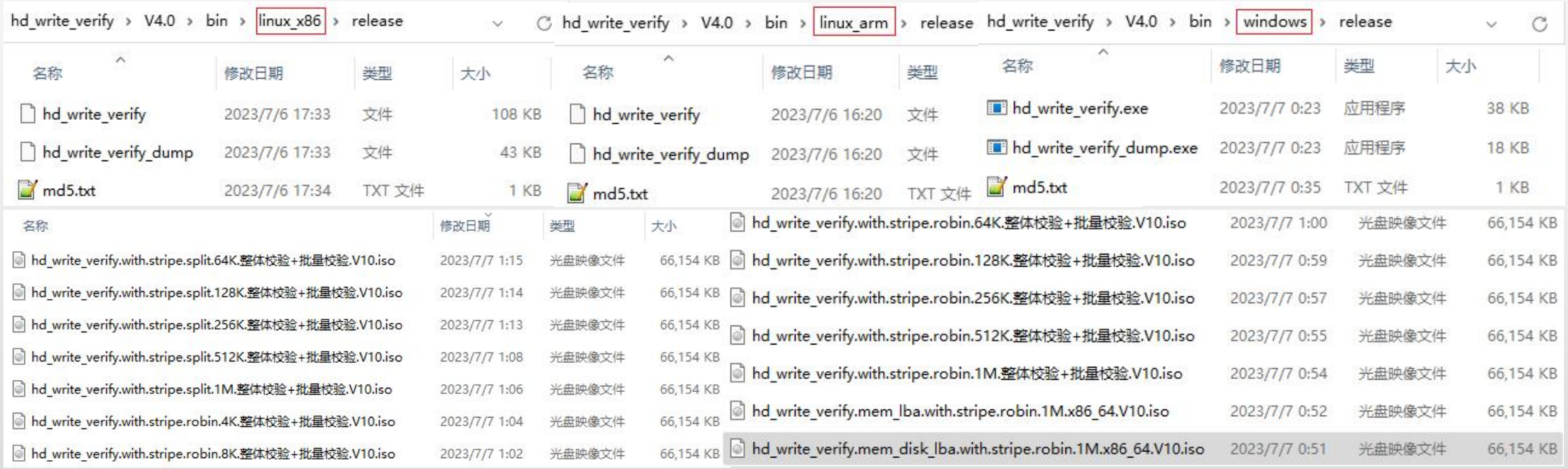
LBA工具介绍

- 借鉴于前同事编写的程序的实现原理，创新实现LBA工具：HD_WRITE_VERIFY & HD_WRITE_VERIFY_DUMP，全部代码由本人重新编写实现(先完成Linux版本的代码，然后移植完成Windows版本的代码)，并制作成自动化测试系统的系列ISO，完全拥有相关工具和自动化测试系统的知识产权；

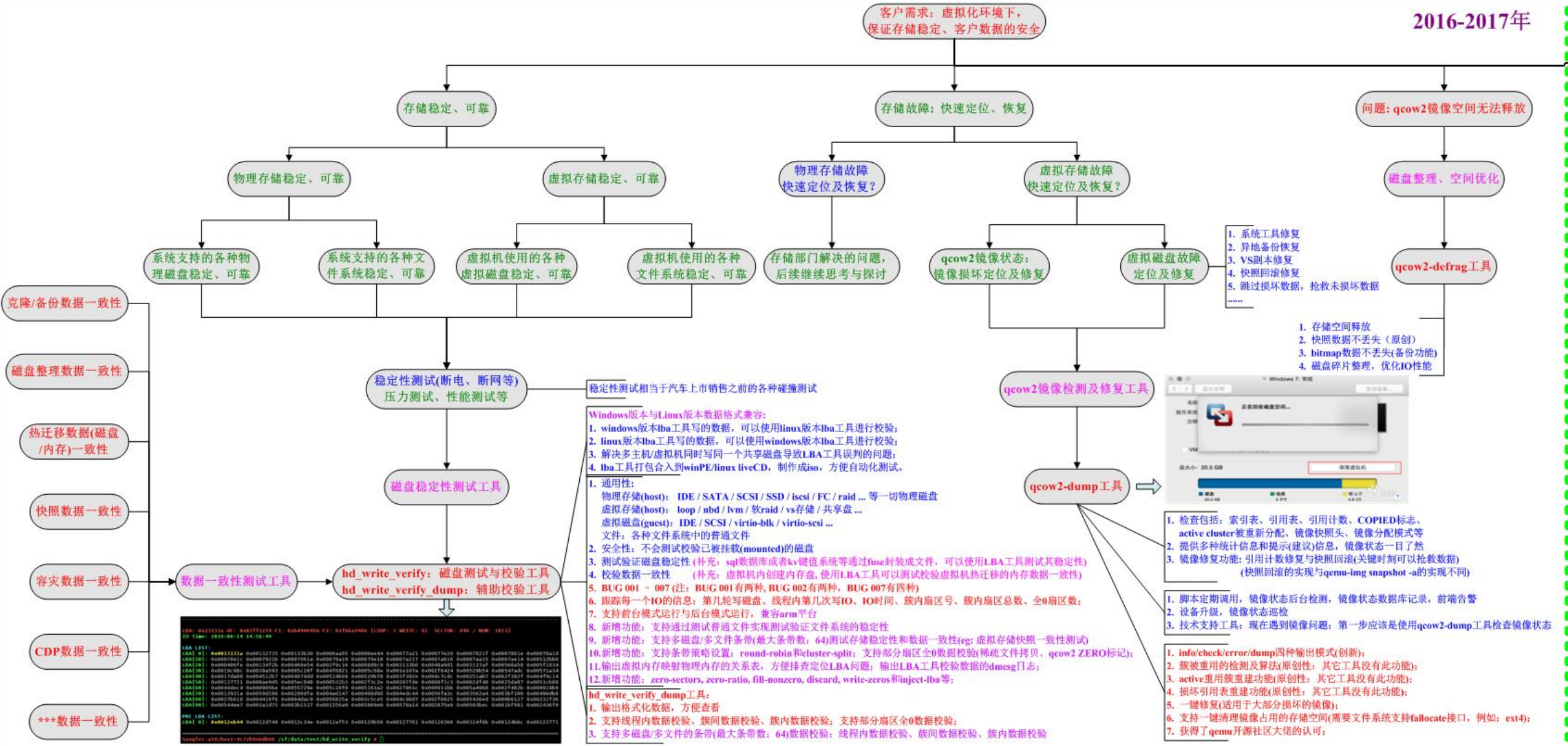


- LBA工具从2016年的V0.01版本升级到现在的V17.0版本(代码经过十多年无数次迭代、优化)，功能非常完善和强大：将复杂的存储/文件系统稳定性和数据一致性问题降维转化为直观、易懂、数据可视化的问题(测试校验完全自动化或者手动简单敲几个命令即可---谁用谁知道，不要看广告，看疗效)，并附带多个原创高级功能。

- 目前LBA工具有Linux(x86和arm)和Windows三种版本(https://github.com/zhangyoujia/hd_write_verify)，2.2+万行C代码。



LBA工具介绍：功能简图



LBA工具介绍：版权声明与数据布局

- 设计介绍：

数据结构: 数组、队列、强双向链表(每个节点99个指针指向后向节点，10个指针指向前向节点)组成的混合体；

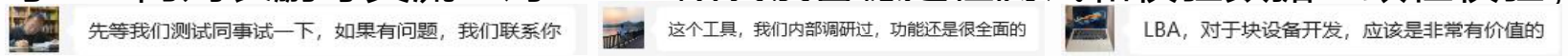
算法:　　非常复杂、精巧，部分高级功能的算法借鉴了dpdk、qemu、tgt和rsync等开源代码；

```
/*
 * HD_WRITE_VERIFY & HD_WRITE_VERIFY_DUMP v23.06
 *
 * Copyright (c) 2016-2023 YOUPLUS.
 *
 * Author: YOUPLUS <zhang_youjia@126.com>
 */

/*
 *  -----------------       -----------------       -----------------       -----------------
 *  | HF            |       | HF            |       | HF            |       | HF            |
 *  -----------------       -----------------       -----------------       -----------------
 *  | MAGIC         |       | MAGIC         |       | MAGIC         |       | MAGIC         |
 *  -----------------       -----------------       -----------------       -----------------
 *  | PRE_LBA[10]   |       | PRE_LBA[10]   |       | PRE_LBA[10]   |       | PRE_LBA[10]   |
 *  -----------------       -----------------       -----------------       -----------------
 *  | LBA_LIST[100] |       | LBA_LIST[100] |       | LBA_LIST[100] |       | LBA_LIST[100] |
 *  -----------------       -----------------       -----------------       -----------------
 *  | TIME          |       | TIME          |       | TIME          |       | TIME          |
 *  -----------------       -----------------       -----------------       -----------------
 *  | RANDOM        |       | RANDOM        |       | RANDOM        |       | RANDOM        |
 *  -----------------       -----------------       -----------------       -----------------
 *  | NUM           |       | NUM           |       | NUM           |       | NUM           |
 *  -----------------       -----------------       -----------------       -----------------
 *  | INDEX         |       | INDEX         |       | INDEX         |       | INDEX         |
 *  -----------------       -----------------       -----------------       -----------------
 *
 *  |<---sector---->|       |<---sector---->|   ...   |<---sector---->|       |<---sector---->|
 *  |<-----------------------------------------cluster----------------------------------------->|
 *
 *  |-------------------------------------------------------------------------------------------|
 *  |   cluster     |   cluster     |   cluster ... |   cluster     |   cluster     |   cluster     |
 *  |-------------------------------------------------------------------------------------------|
 *  |<-----------------------------------------disk------------------------------------------->|
 */
```

```
DATA LAYOUT:
=============================================

offsetof HF:                    0    [4   bytes]
offsetof MAGIC:                 4    [4   bytes]
offsetof PRE_LBA[10]:           8    [40  bytes]
offsetof LBA_LIST[100]:         48   [400 bytes]
offsetof TIME:                  448  [56  bytes]
offsetof RANDOM:                504  [4   bytes]
offsetof NUM:                   508  [2   bytes]
offsetof INDEX:                 510  [2   bytes]

=============================================
```

LBA工具介绍：技术交流、测试标准

- 与Intel的刘长鹏等交流：对SPDK做块设备稳定性测试和校验数据一致性校验，主要使用fio等工具的verify功能。



- 与华为、华三、腾讯、阿里、字节跳动、天翼等的研发技术交流，测试存储稳定性和校验数据一致性使用的也是一般的工具与方法。

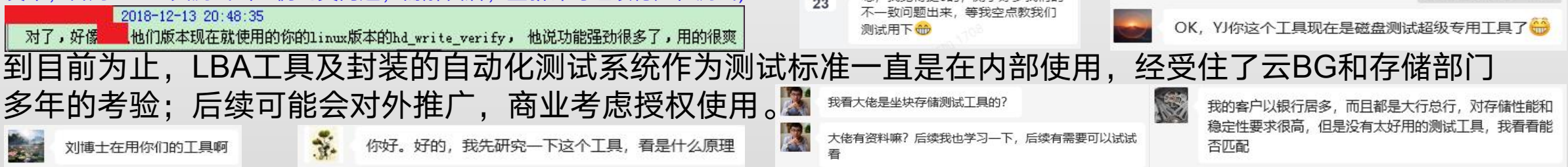
- 参与云产品送检公安部认证，测试存储稳定性和校验数据一致性也没有标准工具，测试过程中推荐使用了LBA工具。

- 存储部门目前测试存储稳定性使用的是fio/vdbench的verify功能，反馈使用时有限制，使用fio/vdbench等工具测试出来的自研存储数据不一致问题不多，但使用LBA工具却测试出了很多数据不一致的问题(后续举例说明)。

- 在SF科技，存储性能测试主要使用fio、iometer、vdbench等工具，并作为标准。

- 但存储稳定性、可靠性测试和数据一致性校验，LBA工具及封装成的自动化测试系统是标准。

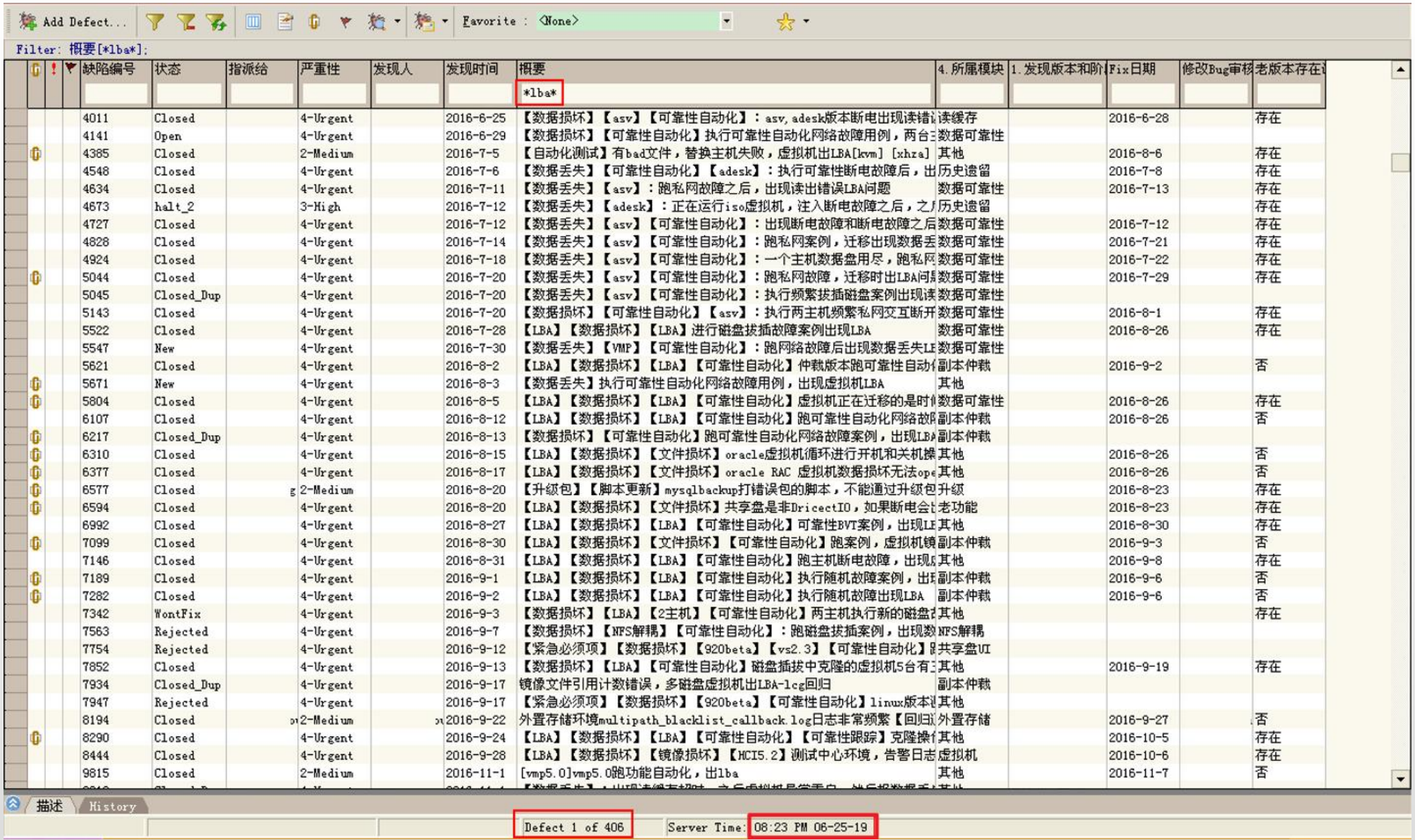
(一般三套物理集群中，不间断的跑断电、断网、拨物理盘、在IO路径关键位置故障注入的测试，必须至少稳定运行两周不出数据损坏、数据丢失等问题，产品才能发布，否则LBA工具测试出一例此类问题，需解决后，重新计时继续稳定性测试)



- 到目前为止，LBA工具及封装的自动化测试系统作为测试标准一直是在内部使用，经受住了云BG和存储部门多年的考验；后续可能会对外推广，商业考虑授权使用。

LBA工具介绍：工具的作用与价值

- SF科技分布式存储早期版本存在各种数据丢失、数据错误的问题，经过几年的更新迭代，才趋于稳定。



2016-2019三年，LBA工具测试发现的SF科技分布式存储数据不一致问题，问题单记录：400+(简单过滤，数据有遗漏，不完整)

节省了多少研发成本？

如左图数据丢失、数据错误的问题，其中有很多典型案例，可惜没有保存当时排查、解决问题的相关资料，只有后面贴出的几张截图。

LBA工具介绍：产品发布标准

- SF科技产品发布标准 -- 在断电、断网、故障注入的环境至少稳定运行两周，不出数据问题

主题 [redacted] 版本稳定性运行日报（7.1）（稳定运行6天；无新增AB类问题）

欢迎页面 | 张友加53628 - 收件箱 | [redacted] A类问题进展跟踪【0621】

新建便笺 | 答复 | 答复所有人 | 转发 | 删除 | 后续 | 文件夹 | 拷贝到新建 | 交谈 | 显示 | 工具

[redacted]
2019-06-21 09:35

收件人 [redacted]
抄送
密送
主题 [redacted]A类问题进展跟踪【0621】

79776	**本地磁盘发生簇重用** 在本地盘创建了两个虚拟机跑iso工具	[redacted]	[redacted]	重现中	//6.13 1、分析磁盘元数据，元数据特征、bitmap泄漏检查 ||对bitmap进行了调试，未发现异常点 //6.14 1、继续分析磁盘元数据 2、开始重现，使用loop挂载一个文件，多进程并发，[redacted] 重现。重现方法和脚本[redacted]确认。 //6.19 一个vs存储，一个本地存储，每个存储上创建2个raw文[redacted] 备挂载点目录下又创建2个qcow2文件，因此每个存储上[redacted] qcow2文件），大概15-20min跑一轮。目前未重现出来 //6.21 仍未重现出来，目前无其他重现方案。问题分析人力投[redacted] 镜像损坏问题。
80711	**虚拟机bitmap损坏** 虚拟机仅执行开关机操作。	[redacted]		分析中	//6.19 检查bitmap簇是否被重新分配了 //6.21 未发现簇重用，需要继续分析。
80427	**4主机集群，将1台物理主机连续强制关机了2次，虚拟机镜像损坏了** 虚拟机4月份创建了（通过克隆创建），虚拟机执行了备份、迁移、电源先关操作。主机执行过强制宕机、关机操作。	[redacted]		分析中	//6.19 L2表对应的簇重用了，与快照头对应的簇重用。 需要确认镜像损坏的时间点，目前根据虚拟机的镜像文[redacted] 和簇重用信息，查看6月6日的虚拟机相关日志。 //6.21 1、关机克隆vs存储的镜像是两条路径 2、增加overlap检查 3、分析一下管理面关机克隆的流程，判断开关机条件是[redacted]

LBA工具磁盘稳定性自动化测试：（winPE的iso运行后自动测试）

管理员: X:\windows\system32\cmd.exe

管理员: X:\windows\system32\cmd.exe - hd_write_verify.exe -c -D -F -S 1024 -V all -T 10 \\.\physicaldrive0

```
X:\>hd_write_verify.exe -c -D -F -S 1024 -V all -T 10 \\.\physicaldrive0
Disk size: 10737418240 / 10240M / 10G
硬盘: \\.\physicaldrive0 | 线程数: 10 | 扇区总数: 20971520 | 簇总数: 20480 | 簇内扇区数: 1024 | DIRECT IO & 刷新硬盘缓存 | 校验模式: 3 | 通知: 0

Current Time: 2019-01-04 11:03:47
Thread 8: Start check disk ...
Thread 9: Start check disk ...
Thread 1: Start check disk ...
Thread 0: Start check disk ...
Thread 2: Start check disk ...
Thread 3: Start check disk ...
Thread 4: Start check disk ...
Thread 5: Start check disk ...
Thread 6: Start check disk ...
Thread 7: Start check disk ...

Current Time: 2019-01-04 11:03:47
WARNING! Thread 6 DIFFER FLAGS
LBA_0: 0x405000 - LBA_1: 0x405400 flags is different | read[5]: 524288

LBA_0 IO Time: 2019-01-04 10:43:59
LBA_1 IO Time: 2019-01-04 10:51:13

Current Time: 2019-01-04 11:03:47
BUG 004: Thread: 6 DATA LOST | read[3]: 524288
PREV LBA: 0x405000 | LOST LBA: 0x405400 | NEXT LBA: 0x405c00
PREV IO Time: 2019-01-04 10:43:59
LOST IO Time: 2019-01-04 10:51:13
```

第6轮测试中的第5和第6个写IO丢失

```
this will take a long time
don't break when repair option opened, otherwise may damage the file system
please wait...
inodepno=143363, pno=165302 not mark on bitmap, name:vm-disk-1.qcow2
inodepno=143363, pno=165303 not mark on bitmap, name:vm-disk-1.qcow2
file[vm-disk-1.qcow2] blocks error:inodepno=143363, blocks=21847, used=21847, errs=0, unmark=2
file[vm-disk-3.qcow2] blocks error:inodepno=339971, blocks=87015, used=88108, errs=0, unmark=0
inodepno=348163, pno=493038 used 2, name:vm-disk-4.qcow2
inodepno=348163, pno=493039 used 2, name:vm-disk-4.qcow2

BitmaFree=1688596,      CountFree=1688594
FileUsed=215575,        Total=1904640
RGFUsed=465,    ErrorCount=60
RescuedBlock=54
```

[redacted] 2019-6-6: 见附件 4.png

一个数据块，被physicaldrive3和physicaldrive4同时使用了，导致的lba

文件系统出问题：出现数据块重用的错误，即：一个数据块被两个或者多个虚拟机的镜像引用，导致两个虚拟机写数据时，被重用的数据块出现数据被相互覆盖的情况，LBA工具测试发现数据不一致。

LBA工具介绍：工具的作用与价值

- Windows Server 2016系统上安装的iscsi target server出现数据不一致问题

案例一：重庆XXXX学院

案例二：广西XX投资集团

案例三：意大利FastXXX

```
BUG 004: [Thread: 2] DATA LOST | read[4641]
PREV LBA: 0x60a85d0 | LOST LBA: 0x7904050 | NEXT LBA: 0x9d008b0

PREV LBA: 0x60a85d0 | DATA | LBA: 0x60a85d0 HF: 0x00101272 F1: 0xb8e2a4a6 F2: 0xde427884 [LOOP: 1 WRITE: 4639 SECTOR: 0 / NUM: 16]
PREV IO Time: 2019-10-11 20:23:15

LOST LBA: 0x7904050 | DATA | LBA: 0x78fc800 HF: 0x00801277 F1: 0x3848b187 F2: 0x945d2e0c [LOOP: 1 WRITE: 3901 SECTOR: 80 / NUM: 128]
LOST IO Time: 2019-10-10 15:48:24

NEXT LBA: 0x9d008b0 | DATA | LBA: 0x9d008b0 HF: 0x00101272 F1: 0xb8e2a4a6 F2: 0xde427884 [LOOP: 1 WRITE: 4641 SECTOR: 0 / NUM: 16]
NEXT IO Time: 2019-10-11 20:23:15

PRE LBA:  0x01f62600 0x059e6000 0x03120c80 0x01d89400 0x03686880 0x089ecb00 0x00006400 0x041c8b80 0x081d3c80 0x01f53e00
```

(1) [Thread: 2] DATA LOST表示数据丢失
(2) 2号线程的第4640次写的数据丢失，数据的偏移：第0x7904050扇区
(3) 第0x7904050扇区中的当前数据是一份旧数据，时间：2019-10-10 15:48:24

Windows 2016 iscsi target 问题，目前整体情况总结如下：

1、研发内部搭建了 windows 2016 iscsi target 环境，已将问题复现出来，经过调试分析，基本确定是 windows 2016 iscsi target 存储存在的一个潜在 bug。此 bug 的触发条件是：

(1) 主机 A 和 B 共享此存储；

(2) 主机 A 不断从存储读数据，且读请求块大小超过 256K；

(3) 主机 B 通过 SCSI RESERVE/RELEASE 指令对存储进行加锁/解锁操作。以上条件同时出现，主机 A 会有一定概率读到错误的数据。

2、此问题我们已经找到必现方法，在 centos7.5 环境，简单几步操作可以复现出来。从 bug 的表征看，猜测是 windows 存储端的分片处理逻辑与加锁解锁逻辑之间配合有问题，导致读取到的数据内容错误。我们验证了其它物理 iscsi 存储和 vs 虚拟 iscsi 存储，没有此问题。

3、此存储 bug 触发的必要条件之一是：读请求的块大小超过 256K。因此，可以通过下面的方法，将存储的请求块大小限制为不超过 256K，可以临时避开这个 bug。

```
echo  256  >  /sys/block/dm-X/queue/max_sectors_kb
```

1. 问题描述

虚拟机从 iscsi 存储迁移到虚拟存储后，数据库日志提示有坏块，但虚拟机运行正常，检查虚拟机磁盘镜像也正常，问题分析如下。

2. 原因及过程分析

2.1 问题现象

一台跑 oracle 数据库的虚拟机，在开机热迁移之后出现坏块，影响业务运行；虚拟机首次从虚拟存储迁移到 iscsi 正常，数据库也没有报错；通过构造环境，对热迁移过程进行分析，可以重现 iscsi 上热迁移虚拟机到虚拟存储报错的问题。

2.2 故障分析与重现

1、数据库报错日志分析

```
Incident details in: /home/oracle/app/oracle/diag/rdbms/easdb/easdb/incident/incdir_36804/easdb_ora_31240_i36804.t
Wed Sep 04 08:02:27 2019
Trace dumping is performing id=[cdmp_20190904080227]
Errors in file /home/oracle/app/oracle/diag/rdbms/easdb/easdb/incident/incdir_36803/easdb_ora_31240_i36803.trc:
ORA-00354: 损坏重做日志块标头
ORA-00353: 日志损坏接近块 465920 更改 487331436 时间 09/03/2019 23:13:01
ORA-00312: 联机日志 4 线程 1: '/home/oracle/app/oracle/oradata/easdb/redo04.log'
ORA-01578: ORACLE 数据块损坏 (文件号 6, 块号 2398786)
ORA-01110: 数据文件 6: '/home/oracle/oradata/easdb/EAS_D_GXJT_STANDARD2.ORA'
Wed Sep 04 08:02:27 2019
Sweep [inc][36804]: completed
Checker run found 1 new persistent data failures
Errors in file /home/oracle/app/oracle/diag/rdbms/easdb/easdb/trace/easdb_ora_31240.trc  (incident=36805):
ORA-00353: 日志损坏接近块 434176 更改 487387330 时间 09/03/2019 23:13:40
ORA-00334: 归档日志: '/home/oracle/app/oracle/oradata/easdb/redo05.log'
ORA-01578: ORACLE 数据块损坏 (文件号 6, 块号 2398786)
ORA-01110: 数据文件 6: '/home/oracle/oradata/easdb/EAS_D_GXJT_STANDARD2.ORA'
Incident details in: /home/oracle/app/oracle/diag/rdbms/easdb/easdb/incident/incdir_36805/easdb_ora_31240_i36805.t
Wed Sep 04 08:02:30 2019
```

在迁移数据库虚拟机后，可以明显看到数据库日志中有坏块提示；此时业务无法正常访问数据库。同时，分析迁移后的虚拟机还是正常运行状态，那么能否有方法通过重现来

3、结论

通过以上的测试对比发现，测试时的 IO 块大小为 512K 和 1M，AC 虚拟机内部发现异常是的 IO 块大小为 256K，测试时无论是使用 nbd 的方式模拟虚拟机读写大块 IO，还是测试裸设备直接读写大块的 IO，都出现了读写的数据不一致，可以确定是存储的问题，大概率和块 size、边界对齐这些条件有关系。有可能是外置存储的 cache 的按条带对齐的处理有问题，非条带对齐写入后从 cache 读出数据是旧的。

把 lba 工具改造了一下，基本实锤了，就是存储那边返回的数据内容不对，触发因素就是 scsi 保留锁指令影响了读操作返回结果，这种属于存储侧的严重 bug，vmware 那边的情况只是问题还没暴露，按照我们对接 vs 的经验，存储层面的问题，需要专业工具配合各种场景构造，才有可能撞出来。
用其它厂商的 iscsi 进行测试没有问题，用 windows 的 iscsi 测试几秒钟就跑出来了，基本可以确认是 windows iscsi 的 bug。客户纠结 vmware 没问题，只是问题没有暴露或没有被发现，换一个角度去看，windows iscsi 的 bug，vmware 没有发现，而我们发现了，刚好体现了我们测试工具的专业性以及备份数据检查的严格性。

LBA工具介绍：工具的作用与价值

- arm平台：测试iscsi裸盘没问题，测试iscsi格式化后的ocfs2出现数据丢失

  mkfs.ocfs2 -b 4K -C 64K device；格式化时指定cluster-size<=64K进行规避

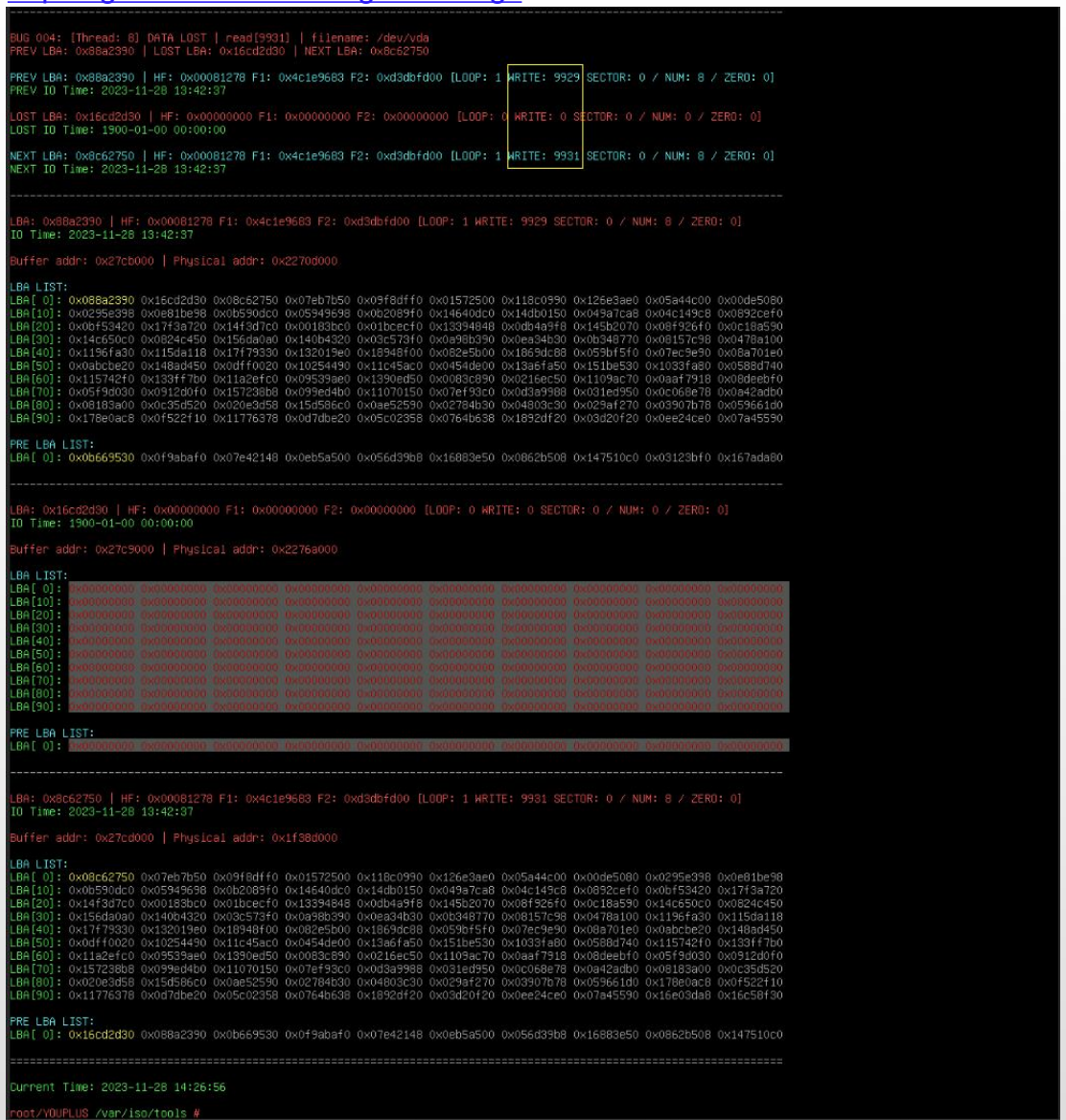
```
[root@node-1 test12345]# uname -a
Linux 5.15.67-12.cl9.aarch64 #1 SMP Mon Sep 18 20:45:26 CST 2023 aarch64 aarch64 aarch64 GNU/Linux
[root@node-1 test12345]#
[root@node-1 ~]# /root/hd_write_verify -c -D -S 1024 -V all -T 10 /dev/mapper/3600507670881083f4800000000011f46
--------------------------------------------------------------------------------

Device Topology:
        Disk: /dev/mapper/3600507670881083f4800000000011f46
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 32768
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 42949672960 / 40960M / 40.00G

--------------------------------------------------------------------------------

Disk: /dev/mapper/3600507670881083f4800000000011f46 | Thread: 10 | Total Sectors: 83886080 | Total Clusters: 81920 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush
| Verify: 6 | Notify: 0

Current Time: 2023-11-27 16:12:34
Thread 3 [tid: 279134]: Starting check disk ...
Thread 8 [tid: 279139]: Starting check disk ...
Thread 7 [tid: 279138]: Starting check disk ...
Thread 1 [tid: 279132]: Starting check disk ...
Thread 5 [tid: 279136]: Starting check disk ...
Thread 2 [tid: 279133]: Starting check disk ...
Thread 6 [tid: 279137]: Starting check disk ...
Thread 4 [tid: 279135]: Starting check disk ...
Thread 0 [tid: 279131]: Starting check disk ...
Thread 9 [tid: 279140]: Starting check disk ...

--------------------------------------------------------------------------------

Starting write disk:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-11-27 16:12:34
5933, 4033 | 5975, 4069 | 5954, 4041 | 5952, 4048 | 5960, 4044 | 5978, 4056 | 5978, 4057 | 5963, 4051 | 5969, 4052 | 5956, 4044 |

Current Time: 2023-11-27 16:15:27
Thread 9 [tid: 279140]: Starting check disk ...
Thread 8 [tid: 279139]: Starting check disk ...
Thread 3 [tid: 279134]: Starting check disk ...
Thread 1 [tid: 279132]: Starting check disk ...
Thread 5 [tid: 279136]: Starting check disk ...
Thread 2 [tid: 279133]: Starting check disk ...
Thread 7 [tid: 279138]: Starting check disk ...
Thread 6 [tid: 279137]: Starting check disk ...
Thread 0 [tid: 279131]: Starting check disk ...
Thread 4 [tid: 279135]: Starting check disk ...
```

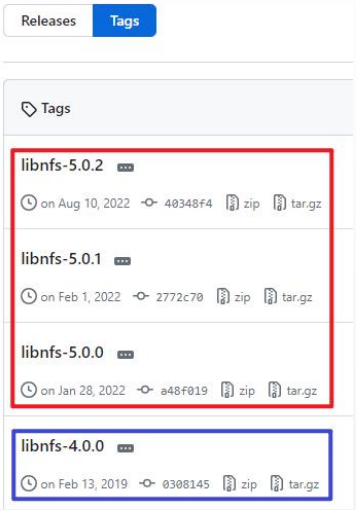
```
[root@node-1 ~]# mount | grep 3600507670881083f4800000000011f46
/dev/mapper/3600507670881083f4800000000011f46 on /mnt/share-storage/test12345 type ocfs2 (rw,noatime,nodiratime,_netdev,heartbeat=local,nointr,data=ordered,errors=re
mount-ro,atime_quantum=60,cluster_stack=o2cb,coherency=full,user_xattr,acl,_netdev)
[root@node-1 ~]#
[root@node-1 test12345]# truncate --size 20G lba.raw
[root@node-1 test12345]#
[root@node-1 test12345]#
[root@node-1 test12345]# /root/hd_write_verify -c -D -S 1024 -V all -T 10 lba.raw
--------------------------------------------------------------------------------

File Information:
        File:      lba.raw
        Size:      21474836480
        Blocks:    0
        IO Block:  1048576
        Device:    64804
        Inode:     1035777
        Links:     1
        File Size: 21474836480 / 20480M / 20.00G

--------------------------------------------------------------------------------

File: lba.raw | Thread: 10 | Total Sectors: 41943040 | Total Clusters: 40960 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush | Verify: 6 | Notify: 0

Current Time: 2023-11-27 16:20:31
Thread 2 [tid: 342913]: Starting check file ...
Thread 7 [tid: 342918]: Starting check file ...
Thread 5 [tid: 342916]: Starting check file ...
Thread 1 [tid: 342912]: Starting check file ...
Thread 0 [tid: 342911]: Starting check file ...
Thread 3 [tid: 342914]: Starting check file ...
Thread 9 [tid: 342920]: Starting check file ...
Thread 6 [tid: 342917]: Starting check file ...
Thread 8 [tid: 342919]: Starting check file ...
Thread 4 [tid: 342915]: Starting check file ...

--------------------------------------------------------------------------------

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-11-27 16:20:31

--------------------------------------------------------------------------------

Current Time: 2023-11-27 16:20:32
BUG 002[3]: [Thread: 2] BATCH VERIFY SECTOR[64] DIFFER: 698 | CORRECT LBA[0]: 0x1530000 | ERROR LBA[64]: 0x1530040 | filename: lba.raw
LBA: 0x1530040 | thread_num: 10, bitmap: 0 0 1 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x1ed9400 | HF: 0x04001272 F1: 0x4aa3a49e F2: 0x9e36d54c [LOOP: 1 WRITE: 28 SECTOR: 0 / NUM: 1024 / ZERO: 262]
LAST WRITE IO Time: 2023-11-27 16:20:32

Buffer addr: 0xfffe2c690000 | Physical addr: 0x17b920000

LBA LIST:
LBA[ 0]: 0x01ed9400 0x0001d000 0x01bab000 0x022c8000 0x011e3000 0x013b0000 0x01a85000 0x00217400 0x00c00000 0x00bd7000
LBA[10]: 0x01be8000 0x00f20400 0x018da000 0x00f86000 0x00568000 0x00810000 0x00635c00 0x01f13400 0x015d1000 0x00cd0800
LBA[20]: 0x0031dc00 0x026fb000 0x017ccc00 0x01676800 0x001fe000 0x001aac00 0x02247800 0x018fb000 0x00199c00 0x02248000
LBA[30]: 0x018a5800 0x0080f800 0x014c7800 0x026f9400 0x014a4400 0x00a55000 0x0076ec00 0x000ff400 0x00e16000 0x02250800
LBA[40]: 0x00460000 0x00a3f800 0x00ef8000 0x00744800 0x0228c800 0x027e5000 0x02751000 0x000d8000 0x02009c00 0x00658800
LBA[50]: 0x00b9f000 0x023f5000 0x01cf7000 0x01923400 0x00679800 0x0002d400 0x009ee000 0x020b4800 0x00704000 0x01732000
LBA[60]: 0x01c70800 0x00e64000 0x00366400 0x01924400 0x0197a800 0x00cf2000 0x0016c000 0x00143800 0x020e8000 0x01280000
LBA[70]: 0x01901000 0x01d84400 0x0079b800 0x00cb4000 0x017bc000 0x01a0a000 0x00316000 0x00b6d800 0x002d0000 0x01ac3400
LBA[80]: 0x013a1800 0x00944000 0x01091800 0x02447c00 0x00fa8800 0x00b7c800 0x02511800 0x01d50000 0x00162000 0x01981800
LBA[90]: 0x01b4d800 0x01e2e000 0x00c97800 0x02590800 0x00160000 0x007c8800 0x01f00000 0x0192b800 0x00eb8000 0x0259c000

PRE LBA LIST:
LBA[ 0]: 0x005d5400 0x00f6f800 0x011f8000 0x022b3c00 0x01fdf000 0x015cb000 0x00670800 0x018eb000 0x01a23000 0x01fd3000

--------------------------------------------------------------------------------

BATCH VERIFY:
LBA: 0x1530000 | HF: 0x04001272 F1: 0x4aa3a49e F2: 0x9e36d54c [LOOP: 1 WRITE: 3 SECTOR: 0 / NUM: 1024 / ZERO: 262]
BATCH VERIFY IO Time: 2023-11-27 16:20:31

Buffer addr: 0xfffde94e0000 | Physical addr: 0x21bfb0000

LBA LIST:
LBA[ 0]: 0x01530000 0x02430400 0x02483400 0x01540000 0x0231e800 0x01475400 0x00af2000 0x00e99000 0x0122b800 0x00eb4000
LBA[10]: 0x002fd400 0x0019a000 0x00628000 0x01ee3800 0x00208c00 0x01fd3000 0x01a23000 0x018eb000 0x00670800 0x015cb000
LBA[20]: 0x01fdf000 0x022b3c00 0x011f8000 0x00f6f800 0x005d5400 0x01ed9400 0x0001d000 0x01bab000 0x022c8000 0x011e3000
LBA[30]: 0x013b0000 0x01a85000 0x00217400 0x00c00000 0x00bd7000 0x01be8000 0x00f20400 0x018da000 0x00f86000 0x00568000
LBA[40]: 0x00810000 0x00635c00 0x01f13400 0x015d1000 0x00cd0800 0x0031dc00 0x026fb000 0x017ccc00 0x01676800 0x001fe000
LBA[50]: 0x001aac00 0x02247800 0x018fb000 0x00199c00 0x02248000 0x018a5800 0x0080f800 0x014c7800 0x026f9400 0x014a4400
LBA[60]: 0x00a55000 0x0076ec00 0x000ff400 0x00e16000 0x02250800 0x00460000 0x00a3f800 0x00ef8000 0x00744800 0x0228c800
LBA[70]: 0x027e5000 0x02751000 0x000d8000 0x02009c00 0x00658800 0x00b9f000 0x023f5000 0x01cf7000 0x01923400 0x00679800
LBA[80]: 0x0002d400 0x009ee000 0x020b4800 0x00704000 0x01732000 0x01c70800 0x00e64000 0x00366400 0x01924400 0x0197a800
LBA[90]: 0x00cf2000 0x0016c000 0x00143800 0x020e8000 0x01280000 0x01901000 0x01d84400 0x0079b800 0x00cb4000 0x017bc000

PRE LBA LIST:
LBA[ 0]: 0x002f2c00 0x00000800 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

--------------------------------------------------------------------------------

DETECT READ ERROR:
LBA: 0x1530040 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
DETECT ERROR IO Time: 1900-01-00 00:00:00

Buffer addr: 0xfffde94e8000 | Physical addr: 0x21bfb8000

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

--------------------------------------------------------------------------------

Current Time: 2023-11-27 16:20:32

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具介绍：工具的作用与价值

- 文件存储(nas)：虚拟机磁盘使用nfs上的qcow2镜像，测试出数据丢失

https://github.com/sahlberg/libnfs/tags

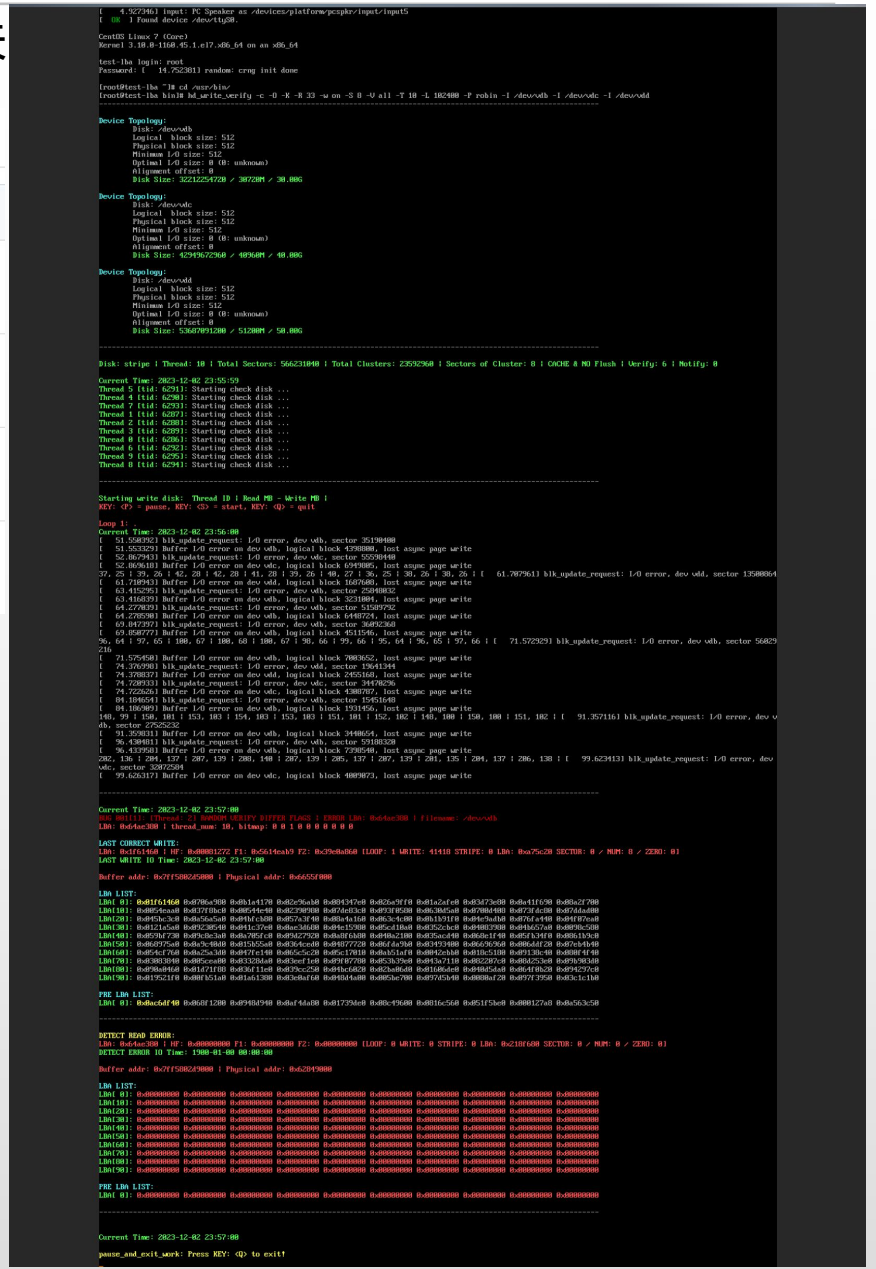




问题原因：libnfs库导致

(1)使用libnfs-5.0.x库，虚拟机内跑LBA工具测试会出现数据丢失和EIO；

(2)使用libnfs-4.0.0库，虚拟机内跑LBA工具测试正常；



LBA工具介绍：工具的作用与价值

- longhorn存储：对源虚拟机快照克隆，校验目的磁盘发现数据丢失

```
PREV LBA IO Time: 2024-01-18 04:45:47

NEXT LBA: 0x2b19800 | HF: 0x08001270 F1: 0x561d61d1 F2: 0xa455fea2 [LOOP: 51 WRITE: 1842 SECTOR: 0 / NUM: 2048 / ZERO: 104]
NEXT LBA IO Time: 2024-01-17 20:01:05

------------------------------------------------------------------------------------------------------------------------

Current Time: 2024-01-18 07:04:58
[BUG 004: [Thread: 0] DATA LOST] read[4004] | filename: /dev/vda
PREV LBA: 0x2119800 - NEXT LBA: 0x4dcc800 flags is different | read[4004]

PREV LBA: 0x2119800 | HF: 0x08001275 F1: 0x561d61d1 F2: 0xedbf13ac [LOOP: 52 WRITE: 4003 SECTOR: 0 / NUM: 2048 / ZERO: 268]
PREV LBA IO Time: 2024-01-18 04:45:42

NEXT LBA: 0x4dcc800 | HF: 0x08001278 F1: 0x561d61d1 F2: 0x281275e8 [LOOP: 51 WRITE: 634 SECTOR: 0 / NUM: 2048 / ZERO: 434]
NEXT LBA IO Time: 2024-01-17 17:43:29

------------------------------------------------------------------------------------------------------------------------

Current Time: 2024-01-18 07:05:09
[BUG 004: [Thread: 0] DATA LOST] read[4012] | filename: /dev/vda
PREV LBA: 0x3d11800 - NEXT LBA: 0x1fb0800 flags is different | read[4012]

PREV LBA: 0x3d11800 | HF: 0x08001279 F1: 0x561d61d1 F2: 0x86e9d138 [LOOP: 52 WRITE: 4011 SECTOR: 0 / NUM: 2048 / ZERO: 88]
PREV LBA IO Time: 2024-01-18 04:45:45

NEXT LBA: 0x1fb0800 | HF: 0x08001276 F1: 0x561d61d1 F2: 0x0ac3f680 [LOOP: 51 WRITE: 2013 SECTOR: 0 / NUM: 2048 / ZERO: 16]
NEXT LBA IO Time: 2024-01-17 20:23:02

------------------------------------------------------------------------------------------------------------------------

Current Time: 2024-01-18 07:05:14
[BUG 004: [Thread: 0] DATA LOST] read[3974] | filename: /dev/vda
PREV LBA: 0x1d7800 - NEXT LBA: 0x288b000 flags is different | read[3974]

PREV LBA: 0x1d7800 | HF: 0x08001273 F1: 0x561d61d1 F2: 0xd68dd248 [LOOP: 52 WRITE: 3973 SECTOR: 0 / NUM: 2048 / ZERO: 356]
PREV LBA IO Time: 2024-01-18 04:45:47

NEXT LBA: 0x288b000 | HF: 0x08001278 F1: 0x561d61d1 F2: 0x281275e8 [LOOP: 51 WRITE: 3173 SECTOR: 0 / NUM: 2048 / ZERO: 434]
NEXT LBA IO Time: 2024-01-17 22:05:28

------------------------------------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------------------------------------

Starting write disk:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 53: ..................................................
Current Time: 2024-01-18 07:05:30
5839, 4030 | 5921, 4068 | 5875, 4064 | 5964, 4126 | 5920, 4087 | 5942, 4085 | 6039, 4144 | 5978, 4136 | 5917, 4079 | 5996, 4135 |

Current Time: 2024-01-18 09:28:10
Thread 9 [tid: 1236]: Starting check disk ...
Thread 1 [tid: 1228]: Starting check disk ...
Thread 5 [tid: 1232]: Starting check disk ...
Thread 4 [tid: 1231]: Starting check disk ...
Thread 6 [tid: 1233]: Starting check disk ...
Thread 8 [tid: 1235]: Starting check disk ...
Thread 2 [tid: 1229]: Starting check disk ...
Thread 0 [tid: 1227]: Starting check disk ...
Thread 3 [tid: 1230]: Starting check disk ...
Thread 7 [tid: 1234]: Starting check disk ...
```

源虚拟机的磁盘，LBA工具已经完第53轮测试并正在进行全盘数据校验，前面52轮测试的数据校验都没有问题。

LBA工具10个线程向源磁盘写入数据，校验克隆后的磁盘数据发现每个线程写入的数据都出现丢失，手动dump校验0号线程写入的数据，如右图：第50轮测试数据，第19个同步direct IO的数据已丢失。

```
Welcome to the YOUPLUS's LBA TESTING SYSTEM

https://github.com/zhangyoujia/

YOUPLUS login: root (automatic login)
Last login: Wed Jan 17 19:19:08 GMT-8 2024 on tty2
root/YOUPLUS /var/iso/tools # hd_write_verify_dump -c -D -T 0 /dev/vda

Device Topology:
        Disk: /dev/vda
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 42949672960 / 40960M / 40.00G

        Sector_Per_Cluster: 2048

Current Time: 2024-01-18 10:17:09

[BUG 004: [Thread: 0] DATA LOST] read[20] | filename: /dev/vda
PREV LBA: 0x2422000 | LOST LBA: 0x3156000 | NEXT LBA: 0x14cd000

PREV LBA: 0x2422000 | HF: 0x08001270 F1: 0x561d61d1 F2: 0x7ba3b97c [LOOP: 50 WRITE: 18 SECTOR: 0 / NUM: 2048 / ZERO: 40]
PREV IO Time: 2024-01-17 08:57:09

LOST LBA: 0x3156000 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
LOST IO Time: 1900-01-00 00:00:00

NEXT LBA: 0x14cd000 | HF: 0x08001270 F1: 0x561d61d1 F2: 0x7ba3b97c [LOOP: 50 WRITE: 20 SECTOR: 0 / NUM: 2048 / ZERO: 40]
NEXT IO Time: 2024-01-17 08:57:15

------------------------------------------------------------------------------------------------------------------------

LBA: 0x2422000 | HF: 0x08001270 F1: 0x561d61d1 F2: 0x7ba3b97c [LOOP: 50 WRITE: 18 SECTOR: 0 / NUM: 2048 / ZERO: 40]
IO Time: 2024-01-17 08:57:09

Buffer addr: 0x7fdd9879b000 | Physical addr: 0x67035000

LBA LIST:
LBA[ 0]: 0x02422000 0x03156000 0x014cd000 0x037f7800 0x03bf1000 0x01902800 0x021e5800 0x012b6000 0x02eb3000 0x00061800
LBA[10]: 0x01fef000 0x025ad800 0x025cd800 0x01cfe800 0x00d2c800 0x01e69000 0x01966800 0x04d05800 0x03764000 0x047bd800
LBA[20]: 0x02940000 0x031fb800 0x001cc000 0x023b4800 0x00d1b000 0x01ecd000 0x007ad000 0x00d98000 0x00192800 0x00ab4000
LBA[30]: 0x03f98000 0x01d42800 0x04b96000 0x0357f000 0x00bb3000 0x00133800 0x00c4e000 0x04c54000 0x0405e000 0x02718000
LBA[40]: 0x03c64800 0x0280f000 0x0195f000 0x02492800 0x00717000 0x016b2000 0x03d54800 0x031cb800 0x01f72000 0x0450b000
LBA[50]: 0x04556000 0x01059800 0x0182b800 0x04e45800 0x03129800 0x04579000 0x00f87000 0x035f2000 0x01b99000 0x04f6a000
LBA[60]: 0x02e1a800 0x02274000 0x03f66000 0x00c76000 0x0280c800 0x03e28800 0x033f9000 0x0197d000 0x00104000 0x01086000
LBA[70]: 0x04855800 0x00929000 0x020af800 0x021f2000 0x00023000 0x04404800 0x01ad3800 0x02668800 0x02c92800 0x01766000
LBA[80]: 0x005e1000 0x02c56000 0x0130b000 0x008a4800 0x019ab000 0x02e6e000 0x000d2000 0x00e48000 0x03763000 0x011c2000
LBA[90]: 0x0185c000 0x031d7000 0x033b3000 0x045dc800 0x0014f000 0x04ff0000 0x04270800 0x031e0000 0x00246800 0x0076c800

PRE LBA LIST:
LBA[ 0]: 0x00159000 0x01af4000 0x016ff800 0x01975800 0x04709000 0x00df2000 0x02e8b000 0x01842000 0x02850000 0x02f2b000

------------------------------------------------------------------------------------------------------------------------

LBA: 0x3156000 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
IO Time: 1900-01-00 00:00:00

Buffer addr: 0x7fdd9889d000 | Physical addr: 0x7fcd8000

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

------------------------------------------------------------------------------------------------------------------------

LBA: 0x14cd000 | HF: 0x08001270 F1: 0x561d61d1 F2: 0x7ba3b97c [LOOP: 50 WRITE: 20 SECTOR: 0 / NUM: 2048 / ZERO: 40]
IO Time: 2024-01-17 08:57:15

Buffer addr: 0x7fdd98639000 | Physical addr: 0x7fc50000

LBA LIST:
LBA[ 0]: 0x014cd000 0x037f7800 0x03bf1000 0x01902800 0x021e5800 0x012b6000 0x02eb3000 0x00061800 0x01fef000 0x025ad800
LBA[10]: 0x025cd800 0x01cfe800 0x00d2c800 0x01e69000 0x01966800 0x04d05800 0x03764000 0x047bd800 0x02940000 0x031fb800
LBA[20]: 0x001cc000 0x023b4800 0x00d1b000 0x01ecd000 0x007ad000 0x00d98000 0x00192800 0x00ab4000 0x03f98000 0x01d42800
LBA[30]: 0x04b96000 0x0357f000 0x00bb3000 0x00133800 0x00c4e000 0x04c54000 0x0405e000 0x02718000 0x03c64800 0x0280f000
LBA[40]: 0x0195f000 0x02492800 0x00717000 0x016b2000 0x03d54800 0x031cb800 0x01f72000 0x04508000 0x04556000 0x01059800
LBA[50]: 0x0182b800 0x04e45800 0x03129800 0x04579000 0x00f87000 0x035f2000 0x01b99000 0x04f6a000 0x02e1a800 0x02274000
LBA[60]: 0x03f66000 0x00c76000 0x0280c800 0x03e28800 0x033f9000 0x0197d000 0x00104000 0x01086000 0x04855800 0x00929000
LBA[70]: 0x020af800 0x021f2000 0x00023000 0x04404800 0x01ad3800 0x02668800 0x02c92800 0x01766000 0x005e1000 0x02c56000
LBA[80]: 0x0130b000 0x008a4800 0x019ab000 0x02e6e000 0x000d2000 0x00e48000 0x03763000 0x011c2000 0x0185c000 0x031d7000
LBA[90]: 0x033b3000 0x045dc800 0x0014f000 0x04ff0000 0x04270800 0x031e0000 0x00246800 0x0076c800 0x048d2000 0x01be4000

PRE LBA LIST:
LBA[ 0]: 0x03156000 0x02422000 0x00159000 0x01af4000 0x016ff800 0x01975800 0x04709000 0x00df2000 0x02e8b000 0x01842000

------------------------------------------------------------------------------------------------------------------------

Current Time: 2024-01-18 10:17:11

root/YOUPLUS /var/iso/tools #
```

LBA工具介绍：工具的作用与价值

- longhorn V1存储：LBA工具测试三副本存储出现数据丢失(单副本没有问题)

LBA工具簇间数据校验：

三副本存储裸盘测试数据丢失情况：
1. 十个线程测试，每个线程写了1.9万次左右IO(75%的1K，25%的512字节)，只有一号线程第18955次IO的1K数据丢失了，数据的LBA偏移地址：0x2ebf8；

2. 使用LBA工具dump读取LBA偏移地址：0x2ebf8的数据为全0；

3. 复现几率高，可以多次测试重现；

4. **原因：** longhorn存储原生缺陷，虚拟机开机时，通过clone方式构建一个单副本的volume，然后再rebuild成一个三副本的volume；在rebuild的过程中，如果存在非4K对齐写IO，会走入read modify write逻辑，此时由于没有互斥该操作，可能导致数据错误；

5. **修复方案：** 使用互斥锁保护非4K对齐写IO的read modify write操作；

```
[root@vm1 ~]# hd_write_verify_dump -c -D -L 0x2ebf8 -S 2 /dev/sdb
--------------------------------------------------------------------------------

Device Topology:
       Disk: /dev/sdb
       Logical  block size: 512
       Physical block size: 512
       Minimum I/O size: 512
       Optimal I/O size: 0 (0: unknown)
       Alignment offset: 0
       Disk Size: 209715200 / 200M / 0.20G

       Sector_Per_Cluster: 2

Current Time: 2024-03-15 15:11:02

================================================================================

LBA: 0x2ebf8 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
IO Time: 1900-01-00 00:00:00

Buffer addr: 0x195f000 | Physical addr: 0x2e70f1000

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

--------------------------------------------------------------------------------

LBA: 0x2ebf9 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
IO Time: 1900-01-00 00:00:00

Buffer addr: 0x195f200 | Physical addr: 0x2e70f1200

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

================================================================================

Current Time: 2024-03-15 15:11:02

[root@vm1 ~]#
```

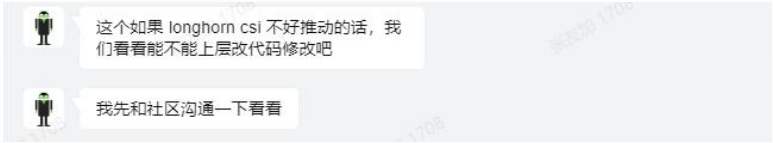
```
[root@vm1 ~]# hd_write_verify_dump -c -D -L 0x5f31c -C 0 /dev/sdb
--------------------------------------------------------------------------------

Device Topology:
       Disk: /dev/sdb
       Logical  block size: 512
       Physical block size: 512
       Minimum I/O size: 512
       Optimal I/O size: 0 (0: unknown)
       Alignment offset: 0
       Disk Size: 209715200 / 200M / 0.20G

       Sector_Per_Cluster: 2

Current Time: 2024-03-15 15:17:58

================================================================================

BUG 004: [Thread: 1] DATA LOST | read[3] | filename: /dev/sdb
PREV LBA: 0x5f31c | LOST LBA: 0x2ebf8 | NEXT LBA: 0x17c76

PREV LBA: 0x5f31c | HF: 0x00021271 F1: 0xa18795fe F2: 0xa1660818 [LOOP: 1 WRITE: 18954 SECTOR: 0 / NUM: 2 / ZERO: 0]
PREV IO Time: 2024-03-15 14:54:16

LOST LBA: 0x2ebf8 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
LOST IO Time: 1900-01-00 00:00:00

NEXT LBA: 0x17c76 | HF: 0x00021271 F1: 0xa18795fe F2: 0xa1660818 [LOOP: 1 WRITE: 18956 SECTOR: 0 / NUM: 2 / ZERO: 0]
NEXT IO Time: 2024-03-15 14:54:16

--------------------------------------------------------------------------------

LBA: 0x5f31c | HF: 0x00021271 F1: 0xa18795fe F2: 0xa1660818 [LOOP: 1 WRITE: 18954 SECTOR: 0 / NUM: 2 / ZERO: 0]
IO Time: 2024-03-15 14:54:16

Buffer addr: 0x205c000 | Physical addr: 0x2e42e1000

LBA LIST:
LBA[ 0]: 0x0005f31c 0x0002ebf8 0x00017c76 0x00036c24 0x00005fea 0x00005ba6 0x00032c14 0x00025dd6 0x00028cee 0x0005acb2
LBA[10]: 0x0003fdb6 0x0003bdca 0x000062fa 0x0001fcde 0x000548ae 0x0001e6e2 0x0004c498 0x0004052c 0x0002b5a4 0x00003afa
LBA[20]: 0x0005beee 0x0002c126 0x0004eff4 0x0003e720 0x0005e8ee 0x00022a46 0x0005b00a 0x00018148 0x000040d4 0x00015e9a
LBA[30]: 0x0003b5b0 0x0004fb4a 0x00029576 0x00032c16 0x000638f2 0x0004ec9c 0x00041abe 0x0001bc86 0x0000d476 0x0003c6a4
LBA[40]: 0x0001fd46 0x00045c9e 0x00003738 0x0000807a 0x0004cd6e 0x0002405a 0x000436a6 0x0003d116 0x00051876 0x0005b31c
LBA[50]: 0x0001936e 0x0002e2a0 0x0001a7ea 0x0000537c 0x0003e47a 0x0002fe16 0x0001d072 0x000026aa 0x00042bea 0x0001fcf2
LBA[60]: 0x0004666e 0x0004ee20 0x0004cc8c 0x0004f47a 0x0005c2cc 0x000491cc 0x0004223a 0x00014d4c 0x000234b6 0x00036872
LBA[70]: 0x00017910 0x00017e5e 0x00054776 0x00021ef6 0x0001a0ea 0x00008eb4 0x0004a17c 0x0000a72e 0x0003d118 0x00012816
LBA[80]: 0x0000fd4c 0x00023366 0x00026eee 0x0002dcb6 0x000124a4 0x0004c278 0x0003d3fa 0x0005fce4 0x00016ce8 0x0001458e
LBA[90]: 0x000239ea 0x0001a0ee 0x00011c78 0x000130ae 0x0002188e 0x000084fe 0x0004cbee 0x000130ee 0x000432be 0x00057bf6

PRE LBA LIST:
LBA[ 0]: 0x0001c0fa 0x00020cfa 0x00008b5e 0x00017e5a 0x0006083c 0x00020fce 0x00020cf8 0x0002566e 0x0001be64 0x00054d7e

--------------------------------------------------------------------------------

LBA: 0x2ebf8 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
IO Time: 1900-01-00 00:00:00

Buffer addr: 0x205b000 | Physical addr: 0x2d69c1000

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

--------------------------------------------------------------------------------

LBA: 0x17c76 | HF: 0x00021271 F1: 0xa18795fe F2: 0xa1660818 [LOOP: 1 WRITE: 18956 SECTOR: 0 / NUM: 2 / ZERO: 0]
IO Time: 2024-03-15 14:54:16

Buffer addr: 0x205d000 | Physical addr: 0x2bbdd0000

LBA LIST:
LBA[ 0]: 0x00017c76 0x00036c24 0x00005fea 0x00005ba6 0x00032c14 0x00025dd6 0x00028cee 0x0005acb2 0x0003fdb6 0x0003bdca
LBA[10]: 0x000062fa 0x0001fcde 0x000548ae 0x0001e6e2 0x0004c498 0x0004052c 0x0002b5a4 0x00003afa 0x0005beee 0x0002c126
LBA[20]: 0x0004eff4 0x0003e720 0x0005e8ee 0x00022a46 0x0005b00a 0x00018148 0x000040d4 0x00015e9a 0x0003b5b0 0x0004fb4a
LBA[30]: 0x00029576 0x00032c16 0x000638f2 0x0004ec9c 0x00041abe 0x0001bc86 0x0000d476 0x0003c6a4 0x0001fd46 0x00045c9e
LBA[40]: 0x00003738 0x0000807a 0x0004cd6e 0x0002405a 0x000436a6 0x0003d116 0x00051876 0x0005b31c 0x0001936e 0x0002e2a0
LBA[50]: 0x0001a7ea 0x0000537c 0x0003e47a 0x0002fe16 0x0001d072 0x000026aa 0x00042bea 0x0001fcf2 0x0004666e 0x0004ee20
LBA[60]: 0x0004cc8c 0x0004f47a 0x0005c2cc 0x000491cc 0x0004223a 0x00014d4c 0x000234b6 0x00036872 0x00017910 0x00017e5e
LBA[70]: 0x00054776 0x00021ef6 0x0001a0ea 0x00008eb4 0x0004a17c 0x0000a72e 0x0003d118 0x00012816 0x0000fd4c 0x00023366
LBA[80]: 0x00026eee 0x0002dcb6 0x000124a4 0x0004c278 0x0003d3fa 0x0005fce4 0x00016ce8 0x0001458e 0x000239ea 0x0001a0ee
LBA[90]: 0x00011c78 0x000130ae 0x0002188e 0x000084fe 0x0004cbee 0x000130ee 0x000432be 0x00057bf6 0x00013bc2 0x0003d92a

PRE LBA LIST:
LBA[ 0]: 0x0002ebf8 0x0005f31c 0x0001c0fa 0x00020cfa 0x00008b5e 0x00017e5a 0x0006083c 0x00020fce 0x00020cf8 0x0002566e

================================================================================

Current Time: 2024-03-15 15:17:58
```

LBA工具介绍：工具的作用与价值

- K8S容器内运行虚拟机(使用longhorn存储)，关电源时导致数据丢失问题

测试出LBA问题的环境：

1. k8s容器内运行虚拟机；



lba_test.gif

2. 虚拟机使用longhorn存储，配置：系统盘 + n个数据盘；

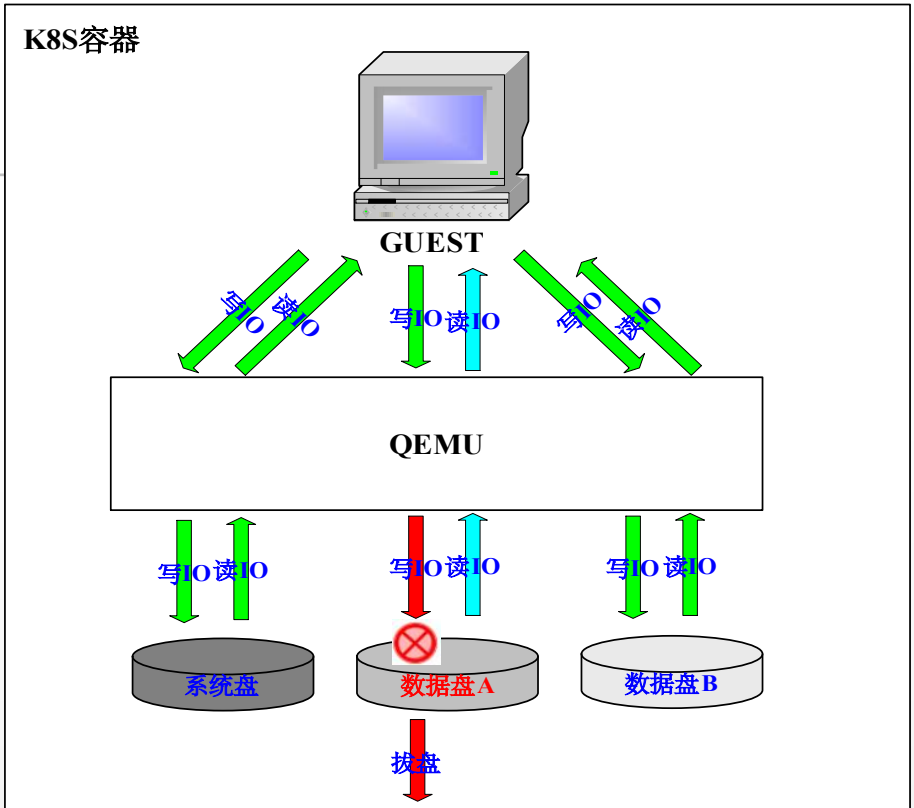
3. 虚拟机运行时，操作关电源；qemu进程还未退出，k8s容器的控制面实现有问题：先detach了数据盘；

4. 此时/sys/block/sdX/size为0，读IO不会失败，会返回全0数据，而写IO会失败，错误码：28 （如下图）
   ENOSPC 28 No space left on device（如右下图：使用nbd1磁盘进行模拟）

5. 假设：在当前虚拟机内，有某个应用正在从数据盘A中拷贝文件到数据盘B中，虚拟机关电源时，可能导致从数据盘A中读取文件的数据为全0，写到数据盘B中的文件是损坏的；

6. **结论：**修改longhorn csi控制面的实现，确保qemu进程未退出之前，不能detach存储。

```
b'   IO iothread1-1241919 [069] d...1 883723.410675: bpf_trace_printk: rw complete req_addr:0x3c024610 ret:0'
b'   IO iothread1-1241919 [069] d...1 883723.410681: bpf_trace_printk: req complete req_addr:0x3c024610 status:0'
b'   IO iothread3-1241921 [004] d...1 883723.410846: bpf_trace_printk: rw complete req_addr:0x34017150 ret:-28'
b'       qemu-kvm-1241916 [064] d...1 883724.411100: bpf_trace_printk: read req_addr:0x34006910 sector_num:0x2ef3600 sector_size:256'
b'   IO iothread3-1241921 [004] d...1 883724.411173: bpf_trace_printk: rw complete req_addr:0x34008d40 ret:-28'
b'   IO iothread3-1241921 [004] d...1 883724.411276: bpf_trace_printk: rw complete req_addr:0x34017150 ret:-28'
b'   IO iothread3-1241921 [004] d...1 883724.411332: bpf_trace_printk: rw complete req_addr:0x34006910 ret:0'
b'   IO iothread3-1241921 [004] d...1 883724.411336: bpf_trace_printk: req complete req_addr:0x34006910 status:0'
b'   IO iothread3-1241921 [004] d...1 883724.411355: bpf_trace_printk: rw complete req_addr:0x34006b10 ret:-28'
b'   IO iothread3-1241921 [004] d...1 883724.411394: bpf_trace_printk: rw complete req_addr:0x34018a80 ret:-28'
b'
b'   PRE LBA LIST:
b'   LBA[ 0]: 0x05de6d00 0x0564c500 0x06e4ec00 0x093b2800 0x07dba200 0x015c2500 0x0120e400 0x04fc6000 0x0873c100 0x05c3fc00
b'
b'   ----------------------------------------------------------------------------------------------------------------
b'
b'   DETECT READ ERROR:
b'   LBA: 0x5de6d00 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 STRIPE: 1 LBA: 0x2ef3600 SECTOR: 0 / NUM: 0 / ZERO: 0]
b'   DETECT ERROR IO Time: 1900-01-00 00:00:00
b'
b'   Buffer addr: 0x7ff49a605000 | Physical addr: 0x5e3d3000
b'
b'   LBA LIST:
b'   LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
b'   LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
b'   LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
b'   LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
b'   LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
```



longhorn存储： iscsi

虚拟机关电源时， qemu进程还未退出， 数据盘先被 detach了， 导致LBA问题。

```
[root@compute467 ~]# cat /sys/block/nbd1/size
0
[root@compute467 ~]#
[root@compute467 ~]# hexdump -C -n 512 /dev/nbd1
[root@compute467 ~]#
[root@compute467 ~]# echo $?
0
[root@compute467 ~]#
[root@compute467 ~]# dd if=/dev/zero of=/dev/nbd1 bs=512 count=1
dd: 写入 '/dev/nbd1' 出错：设备上没有空间
记录了1+0 的读入
记录了0+0 的写出
0字节已复制，0.000214475 s，0.0 kB/s
[root@compute467 ~]#
[root@compute467 ~]# echo $?
1
[root@compute467 ~]#
```

LBA工具介绍：工具的作用与价值

- longhorn v1.9存储(spdk engine)：LBA工具测试发现数据丢失问题(非必现)

测试环境一：

1. 鲲鹏ARM服务器，OpenEuler 22.03版本(5.10.0内核)，Longhorn v1.9版本(spdk engine)；

2. Longhorn存储200G的NVMeOF盘，使用LBA工具连续测试了两天，共跑了200多轮，没测试出问题；

```
Start writing disk: Thread ID | Read: MB - Write: MB | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 110: ..........
Current Time: 2025-06-25 09:08:17

14653, 9821 | 14569, 9785 | 14501, 9722 | 14645, 9868 | 14731, 9892 | 14594, 9792 | 14744, 9884 | 14780, 9920 | 14548, 9756 | 14904, 9993 | 625.61, 431.39 |
```

上图是连续测试了一天时的截图，bsrange: 1-2048，性能在1.1GB/s - 1.3GB/s之间，性能稳定，波动小

测试环境二：

1. 鲲鹏ARM服务器，OpenEuler 24.03版本(6.6.0内核)，Longhorn v1.9版本(spdk engine)；

2. Longhorn存储200G的NVMeOF盘，使用LBA工具连续测试出几次数据不一致问题(非必现，如右图)；

```
Start writing disk: Thread ID | Read: MB - Write: MB | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2025-06-26 17:43:43

33041, 22146 | 34081, 22869 | 15224, 10205 | 37798, 25345 | 20558, 13783 | 18820, 12618 | 26867, 18023 | 35817, 24013 | 23384, 15671 | 18269, 12246 | 587.25, 379.36 |
```

连续测试出几次数据丢失问题，bsrange: 1-2048，性能在400MB/s - 1.1GB/s之间，性能不稳定，波动大

测试NVMeOF盘，簇数据：1M，第0-63扇区(32K)的数据是正确的，第64扇区开始的8扇区(4K)数据出错了(右图鲜红色部分)，第72-2047扇区的数据又全部是正确的。

```
[root@node220 ~]# hd_write_verify_dump -c -D -T 0 /dev/dm-11
--------------------------------------------------------------------------------

Device Topology:
        Disk: /dev/dm-11
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 131072
        Max Sectors KB:   128
        Disk Size: 214748364800 / 204800M / 200.00G

        Sector_Per_Cluster: 2048
        Verify_Thread_NUM:  10

Current Time: 2025-06-26 13:59:17

================================================================================

WARNING: [Thread: 00] WRITE SECTORS INCONSISTENT | read[1] | filename: /dev/dm-11

CORRECT LBA[0]: 0x0 | HF: 0x80051a10 F1: 0xc6647923 F2: 0xd5406f00 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 2048 / ZERO: 598]
CORRECT IO Time: 2025-06-26 13:54:21

ERROR LBA[64]: 0x40 | HF: 0x46b1a000 F1: 0x00000000 F2: 0xcd725966 [LOOP: 7603751 WRITE: 3560166821 SECTOR: 2811 / NUM: 4881 / ZERO: 333810297]
INCONSISTENT IO Time: 1447998616-181777393-1477011813 181228399:-319075540:423481672

--------------------------------------------------------------------------------

BUG 007[1] LAST IO?: [Thread: 00] VERIFY SECTOR[64] DIFFER: 8 | read[1] | filename: /dev/dm-11

hd_write_verify_dump -c -D -L 0x00000000 disk/file
hd_write_verify_dump -c -D -L 0x14279800 disk/file

CORRECT LBA[0]: 0x0 | HF: 0x80051a10 F1: 0xc6647923 F2: 0xd5406f00 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 2048 / ZERO: 598]
CORRECT IO Time: 2025-06-26 13:54:21

ERROR LBA[64]: 0x40 | HF: 0x46b1a000 F1: 0x00000000 F2: 0xcd725966 [LOOP: 7603751 WRITE: 3560166821 SECTOR: 2811 / NUM: 4881 / ZERO: 333810297]
INCONSISTENT IO Time: 1447998616-181777393-1477011813 181228399:-319075540:423481672

CORRECT LBA[72]: 0x48 | HF: 0x80051a10 F1: 0xc6647923 F2: 0xd5406f00 [LOOP: 1 WRITE: 1 SECTOR: 72 / NUM: 2048 / ZERO: 598]
CORRECT IO Time: 2025-06-26 13:54:21

--------------------------------------------------------------------------------

LBA: 0x3f | HF: 0x80051a10 F1: 0xc6647923 F2: 0xd5406f00 [LOOP: 1 WRITE: 1 SECTOR: 63 / NUM: 2048 / ZERO: 598]
IO Time: 2025-06-26 13:54:21

Buffer addr: 0xffffb4147e00 | Physical addr: 0x20540d1d6e00

LBA LIST:
LBA[ 0]: 0x00000000 0x14279800 0x08d6c800 0x03ffd000 0x0b010000 0x0a21e000 0x074de000 0x16daa800 0x07cba000 0x019e4000
LBA[10]: 0x086e8800 0x0d254000 0x16560000 0x0bf5b000 0x18f46800 0x041a4800 0x0fce2800 0x126c5000 0x0da43000 0x142e6800
LBA[20]: 0x13034800 0x15b5b800 0x1609e000 0x0e368000 0x009cf800 0x00180800 0x0f0be800 0x0e250000 0x0fe42000 0x1097c000
LBA[30]: 0x04ec5000 0x0c737000 0x16eaa000 0x0037c800 0x0a560000 0x006c2800 0x06fa1000 0x0ffaa800 0x04db4800 0x11f44000
LBA[40]: 0x13ea8000 0x0a947000 0x0c2eb000 0x02315000 0x11fd2000 0x14b62000 0x14193000 0x13180000 0x09020000 0x09630000
LBA[50]: 0x15839800 0x15ef9000 0x18dff000 0x0410f800 0x188e2000 0x0a287800 0x0abd5000 0x1288f000 0x0ffac000 0x15369800
LBA[60]: 0x00d6a000 0x0df69800 0x15294800 0x0637f000 0x02ef8000 0x00508800 0x1263f800 0x1465c000 0x053ec000 0x11bf5000
LBA[70]: 0x009f9800 0x08865000 0x020ac000 0x13cca800 0x0ac2c000 0x172b2000 0x0dbcf000 0x0aaa9000 0x0058d800 0x0579a000
LBA[80]: 0x0bad8000 0x0c862800 0x109e3800 0x15970000 0x04853000 0x0fb73000 0x0e9d9800 0x13c60000 0x0360b000 0x16f62800
LBA[90]: 0x0c1d7000 0x140d8000 0x07565000 0x0743f000 0x0c3b0800 0x021b2800 0x15ad6000 0x13b71000 0x133a9000 0x01c00800

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

--------------------------------------------------------------------------------

LBA: 0x40 | HF: 0x46b1a000 F1: 0x00000000 F2: 0xcd725966 [LOOP: 7603751 WRITE: 3560166821 SECTOR: 2811 / NUM: 4881 / ZERO: 333810297]
IO Time: 1447998616-181777393-1477011813 181228399:-319075540:423481672

Buffer addr: 0xffffb4148000 | Physical addr: 0x20549c274000

LBA LIST:
LBA[ 0]: 0x00001b03 0x00000000 0x128e29fe 0x00000000 0xf52b1a77 0x0e4b88af 0x7169e34e 0x04f27680 0x1c863c69 0x0100161b
LBA[10]: 0x455c478d 0x15b17480 0xe4e508f1 0x1e110be1 0xb024211e 0x14282f09 0x26758423 0x035229cf 0x22bd3064 0x1bb7b1ff
LBA[20]: 0xd2aca000 0x00000010 0x567ab4c2 0x15a1c691 0xd06e5698 0x08398d94 0x6ec1cad3 0x1264c97b 0x880eb95a 0x1093e1ce
LBA[30]: 0x7450d72b 0x14213802 0xe2f69ae5 0x19efeccf 0xaeec535c 0x13d90c1e 0xac2f8a6b 0x1cefca0d 0xf05071dd 0x08c42320
LBA[40]: 0x05638e29 0x195e8c3d 0xd907f1c5 0x09ec4ed5 0xc2507e38 0x08f59d0e 0x992e0fc7 0x181f5279 0x2b4241f8 0x02781db8
LBA[50]: 0xe46c483f 0x013aee15 0x9a835000 0x00000009 0x8e8a4120 0x178d813a 0x31414824 0x123464e2 0xa2162904 0x0fd173fb
LBA[60]: 0x5fc0c520 0x16185d74 0x296818a4 0x0fe8dbce 0xb0db0314 0x079206c5 0x35910062 0x1e1b646b 0xb6812c0c 0x038bd1c4
LBA[70]: 0x00ca2551 0x10e91ef8 0xed59c4b0 0x0de82379 0xdb531890 0x0dd00dc2 0xf1f6312 0x18529d0e 0x285aec62 0x1200eaf2
LBA[80]: 0xceda5d8c 0x12f50930 0xe0154bb1 0x199a39b8 0x00000000 0x00000000 0x7a36452e 0x17e17e70 0xecafc9a5 0x1485f2b2
LBA[90]: 0x79437914 0x007a47fe 0x129e0f22 0x169c0c5e 0x8ac2cde4 0x0612783c 0x2a5659bc 0x1ce009a6 0x986ecb37 0x13ecec01

PRE LBA LIST:
LBA[ 0]: 0x07672ce5 0x8ef4ef78 0x126b01da 0x68ab7bc1 0x0ecac512 0xad8fde0d 0x1e02e161 0x2e32f06d 0x0b1d0837 0x7f4b836d

--------------------------------------------------------------------------------

LBA: 0x48 | HF: 0x80051a10 F1: 0xc6647923 F2: 0xd5406f00 [LOOP: 1 WRITE: 1 SECTOR: 72 / NUM: 2048 / ZERO: 598]
IO Time: 2025-06-26 13:54:21

Buffer addr: 0xffffb4149000 | Physical addr: 0x2054a3ac1000

LBA LIST:
LBA[ 0]: 0x00000000 0x14279800 0x08d6c800 0x03ffd000 0x0b010000 0x0a21e000 0x074de000 0x16daa800 0x07cba000 0x019e4000
LBA[10]: 0x086e8800 0x0d254000 0x16560000 0x0bf5b000 0x18f46800 0x041a4800 0x0fce2800 0x126c5000 0x0da43000 0x142e6800
LBA[20]: 0x13034800 0x15b5b800 0x1609e000 0x0e368000 0x009cf800 0x00180800 0x0f0be800 0x0e250000 0x0fe42000 0x1097c000
LBA[30]: 0x04ec5000 0x0c737000 0x16eaa000 0x0037c800 0x0a560000 0x006c2800 0x06fa1000 0x0ffaa800 0x04db4800 0x11f44000
LBA[40]: 0x13ea8000 0x0a947000 0x0c2eb000 0x02315000 0x11fd2000 0x14b62000 0x14193000 0x13180000 0x09020000 0x09630800
LBA[50]: 0x15839800 0x15ef9000 0x18dff000 0x0410f800 0x188e2000 0x0a287800 0x0abd5000 0x1288f000 0x0ffac000 0x15369800
LBA[60]: 0x00d6a000 0x0df69800 0x15294800 0x0637f000 0x02ef8000 0x00508800 0x1263f800 0x1465c000 0x053ec000 0x11bf5000
LBA[70]: 0x009f9800 0x08865000 0x020ac000 0x13cca800 0x0ac2c000 0x172b2000 0x0dbcf000 0x0aaa9000 0x0058d800 0x0579a000
LBA[80]: 0x0bad8000 0x0c862800 0x109e3800 0x15970000 0x04853000 0x0fb73000 0x0e9d9800 0x13c60000 0x0360b000 0x16f62800
LBA[90]: 0x0c1d7000 0x140d8000 0x07565000 0x0743f000 0x0c3b0800 0x021b2800 0x15ad6000 0x13b71000 0x133a9000 0x01c00800

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

================================================================================

Current Time: 2025-06-26 13:59:17
```

LBA工具介绍：工具的作用与价值

- qemu：qcow2代码存在原生缺陷可能导致数据丢失问题

测试复现问题：

1. 虚拟磁盘使用raw格式镜像，虚拟机内跑LBA工具测试过程中，kill -9 pid杀qemu进程(模拟虚拟机关闭电源---部分关闭电源的实现案，当关闭电源超时后会kill杀qemu进程)，未出现LBA问题(数据丢失)。

2. 虚拟磁盘使用qcow2格式镜像(preallocation=metadata/full分配模式)，虚拟机内跑LBA工具测试过程中，kill -9 pid杀qemu进程(模拟虚拟机关闭电源---部分关闭电源的实现方案，当关闭电源超时后会kill杀qemu进程)，未出现LBA问题(数据丢失)。

3. 虚拟磁盘使用qcow2格式镜像(preallocation=off分配模式)，虚拟机内跑LBA工具测试过程中，kill -9 pid杀qemu进程(模拟虚拟机关闭电源---部分关闭电源的实现方案，当关闭电源超时后会kill杀qemu进程)，出现LBA问题(数据丢失)。

问题原因：

qemu的qcow2代码存在原生缺陷，preallocation=off分配模式qcow2格式镜像并且已随机写入部分数据后，当存在如下情况：

1. qcow2镜像中的三个簇，第一簇对应的L2表项已经创建分配，第二簇对应的L2表项未创建分配，第三簇对应的L2表项已经创建分配；

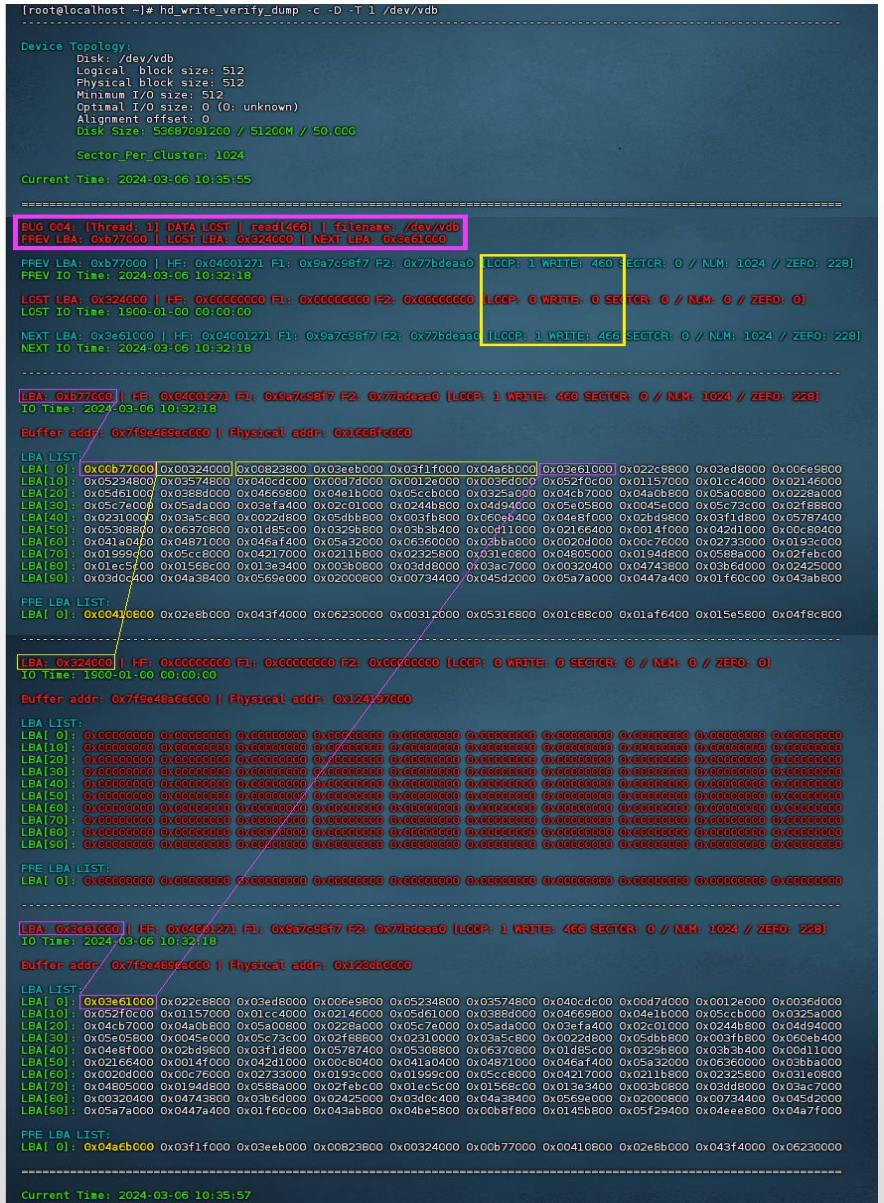
2. 先后向这三簇写入数据，第一簇直接覆盖写数据(不会更新L2表项)；第二簇先在qcow2 cache中创建L2表项，然后写数据；第三簇直接覆盖写数据(不会更新L2表项)；

3. kill -9 pid杀qemu进程时，第二簇在qcow2 cache中创建L2表项还未被刷新到存储中，导致中间IO的数据丢失。

4. qemu所有版本都存在这个问题：测试了qemu-7.0和最新版本qemu-8.2，都可以必现问题。

LBA工具介绍：工具的作用与价值

- qemu：qcow2代码存在原生缺陷可能导致数据丢失问题patch



LBA工具簇间数据校验：
第461-465次IO数据丢失

```
2599 static coroutine_fn int qcow2_co_pwritev_part(
2600         BlockDriverState *bs, int64_t offset, int64_t bytes,
2601         QEMUIOVector *qiov, size_t qiov_offset, BdrvRequestFlags flags)
2602 {
2603     BDRVQcow2State *s = bs->opaque;
2604     int offset_in_cluster;
2605     int ret;
2606     unsigned int cur_bytes; /* number of sectors in current iteration */
2607     uint64_t host_offset;
2608     QCowL2Meta *l2meta = NULL;
2609     AioTaskPool *aio = NULL;
2610     bool flush = false;
2611
2612     trace_qcow2_writev_start_req(qemu_coroutine_self(), offset, bytes);
2613
2614     while (bytes != 0 && aio_task_pool_status(aio) == 0) {
2615
2616         l2meta = NULL;
2617
2618         trace_qcow2_writev_start_part(qemu_coroutine_self());
2619         offset_in_cluster = offset_into_cluster(s, offset);
2620         cur_bytes = MIN(bytes, INT_MAX);
2621         if (bs->encrypted) {
2622             cur_bytes = MIN(cur_bytes,
2623                             QCOW_MAX_CRYPT_CLUSTERS * s->cluster_size
2624                             - offset_in_cluster);
2625         }
2626
2627         qemu_co_mutex_lock(&s->lock);
2628
2629         ret = qcow2_alloc_host_offset(bs, offset, &cur_bytes,
2630                                       &host_offset, &l2meta);
2631         if (ret < 0) {
2632             goto out_locked;
2633         }
2634
2635         if (l2meta /* && running in qemu */) {
2636             flush = true;
2637         }
2638
2639         ret = qcow2_pre_write_overlap_check(bs, 0, host_offset,
2640                                             cur_bytes, true);
2641         if (ret < 0) {
2642             goto out_locked;
2643         }
2644
2645         qemu_co_mutex_unlock(&s->lock);
2646
2647         if (!aio && cur_bytes != bytes) {
2648             aio = aio_task_pool_new(QCOW2_MAX_WORKERS);
2649         }
2650         ret = qcow2_add_task(bs, aio, qcow2_co_pwritev_task_entry, 0,
2651                              host_offset, offset,
2652                              cur_bytes, qiov, qiov_offset, l2meta);
2653         l2meta = NULL; /* l2meta is consumed by qcow2_co_pwritev_task() */
2654         if (ret < 0) {
2655             goto fail_nometa;
2656         }
2657
2658         bytes -= cur_bytes;
2659         offset += cur_bytes;
2660         qiov_offset += cur_bytes;
2661         trace_qcow2_writev_done_part(qemu_coroutine_self(), cur_bytes);
2662     }
2663     ret = 0;
2664
2665     qemu_co_mutex_lock(&s->lock);
2666
2667     //刷新L2表元数据
2668     if (flush) {
2669         ret = qcow2_cache_flush(bs, s->l2_table_cache);
2670     }
2671
2672 out_locked:
2673     qcow2_handle_l2meta(bs, &l2meta, false);
2674
2675     qemu_co_mutex_unlock(&s->lock);
2676
2677 fail_nometa:
2678     if (aio) {
2679         aio_task_pool_wait_all(aio);
2680         if (ret == 0) {
2681             ret = aio_task_pool_status(aio);
2682         }
2683         g_free(aio);
2684     }
2685
2686     trace_qcow2_writev_done_req(qemu_coroutine_self(), ret);
2687
2688     return ret;
2689 }
```

LBA工具介绍：工具的作用与价值

- 自研存储研发过程中，测试出很多数据一致性问题(部分典型问题，总结wiki如下)

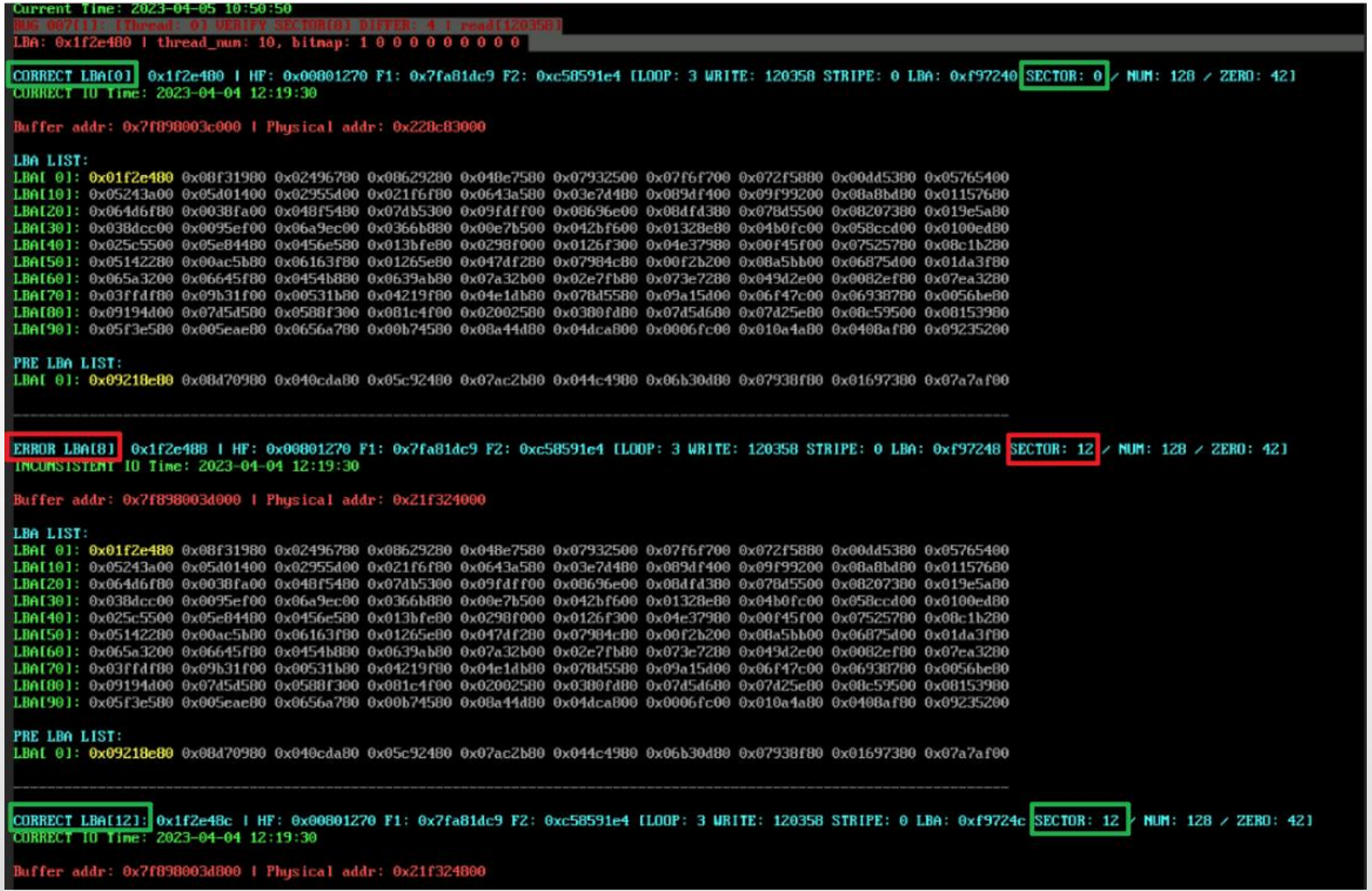
【疑难问题排查】

- 【自研存储】虚拟机内跑LBA工具测试出存储EIO问题
- 【自研存储】虚拟机内跑LBA工具测试出存储出错问题
- 【自研存储】超整合节点重启后回滚快照虚拟机故障
- 【自研存储-全闪】业务中连续替换data-pool的nvme磁盘4次虚拟机数据不一
- 【自研存储-全闪】虚拟机跑LBA工具测试出存储数据不一致性问题
- 【自研存储-全闪】虚拟机跑LBA工具测试出数据不一致问题2--数据丢失
- 【自研存储-全闪】虚拟机跑LBA工具测试出数据不一致问题--数据丢失
- 【自研存储-混闪】三节点之一kill存储引擎虚拟机跑LBA工具测试出数据不一
- 【自研存储-混闪】依次替换booster池中的nvme盘两次虚机出现数据读写不
- 【自研存储-混闪】修改IO及ULT超时阈值后虚机跑LBA工具出现数据不一致问
- 【自研存储-混闪】反复拔插数据盘虚拟机跑LBA工具测试出数据不一致问题
- 【自研存储-混闪】存储异常导致虚拟机文件系统损坏
- 【自研存储-混闪】环境亚健康重构结束后虚拟机磁盘出现数据丢失
- 【自研存储-混闪】虚拟机跑LBA工具主机拔插磁盘测试出数据不一致性问题
- 【自研存储-混闪】虚拟机跑LBA工具测试出单磁盘场景LBA问题
- 【自研存储-混闪】虚拟机跑LBA工具测试出多磁盘场景LBA问题
- 【自研存储-混闪】虚拟机跑LBA工具测试出多磁盘场景LBA问题--数据错位
- 【自研存储-混闪】虚拟机跑LBA工具测试出存储数据不一致性问题
- 【自研存储-混闪】虚拟机跑LBA工具测试出数据不一致问题--两个字节错误
- 【自研存储-混闪】虚拟机跑LBA工具测试出现数据丢失
- 【自研存储-混闪】连续两次删除booster中nvme磁盘虚机内出现数据不一致
- 【自研存储-混闪】连续扩盘连续删盘虚拟机跑出LBA问题

- 存储部门和测试部门每个人都应该熟练掌握LBA工具的使用，
  加强推广普及LBA工具及自动化系统测试验证存储稳定性，
  缩短开发存储产品稳定性达到可正式发布标准的研发周期、降低研发成本。

LBA工具介绍：数据不一致问题，案例一

- 【自研存储-混闪】虚拟机跑LBA工具测试出数据不一致问题--两个字节错误



LBA工具介绍：数据不一致问题，案例一

- 【自研存储-混闪】虚拟机跑LBA工具测试出数据不一致问题--两个字节错误

问题原因：

客户端io使用了相同的buffer导致数据在内存中不一致，之所以采用了相同的buffer是因为olog offset为0导致一个io中的两个recxmap到相同的地址

验证思路：

写入64K数据并且记录每个512字节的crc，批量读取64数据内的多段数据，查看数据crc是否满足要求

验证过程记录：

修改前：

```
wr idx 119 csum 8027518714553792079
wr idx 120 csum 17495080994129261748
wr idx 121 csum 10950643281404104464
wr idx 122 csum 18262799847820884056
wr idx 123 csum 3981115071714742922
wr idx 124 csum 2340204067906706629
wr idx 125 csum 11354231219956398386
wr idx 126 csum 10813301452522952677
wr idx 127 csum 13355868685102970665
rd idx 0 csum 5856497656288625140
unexcepted err ori crc 335982869464060228 new crc 5856497656288625140
rd idx 9 csum 5856497656288625140
unexcepted err ori crc 10228107122013071008 new crc 5856497656288625140
rd idx 20 csum 5856497656288625140
unexcepted err ori crc 69069208052814211 new crc 5856497656288625140
rd idx 30 csum 5856497656288625140
```

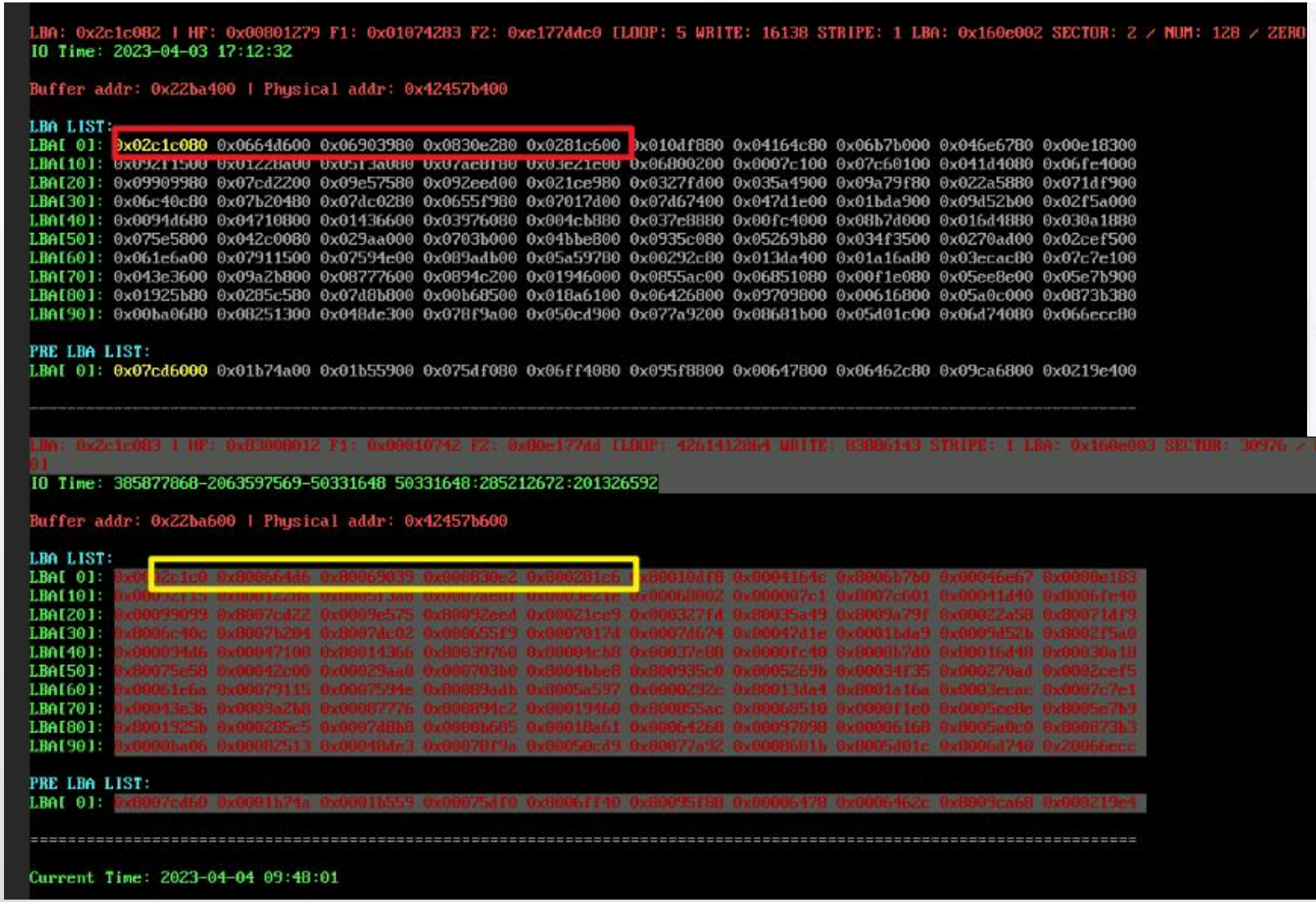
修改后：

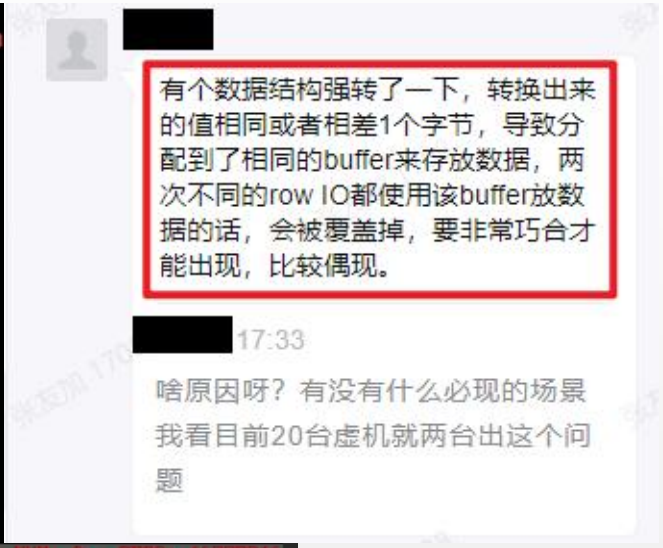
```
wr idx 119 csum 8027518714553792079
wr idx 120 csum 17495080994129261748
wr idx 121 csum 10950643281404104464
wr idx 122 csum 18262799847820884056
wr idx 123 csum 3981115071714742922
wr idx 124 csum 2340204067906706629
wr idx 125 csum 11354231219956398386
wr idx 126 csum 10813301452522952677
wr idx 127 csum 13355868685102970665
rd idx 0 csum 335982869464060228
rd idx 9 csum 10228107122013071008
rd idx 20 csum 69069208052814211
rd idx 30 csum 5856497656288625140
```

LBA工具介绍：数据不一致问题，案例二

- 【自研存储-混闪】虚拟机跑LBA工具测试出多磁盘场景LBA问题--数据错位

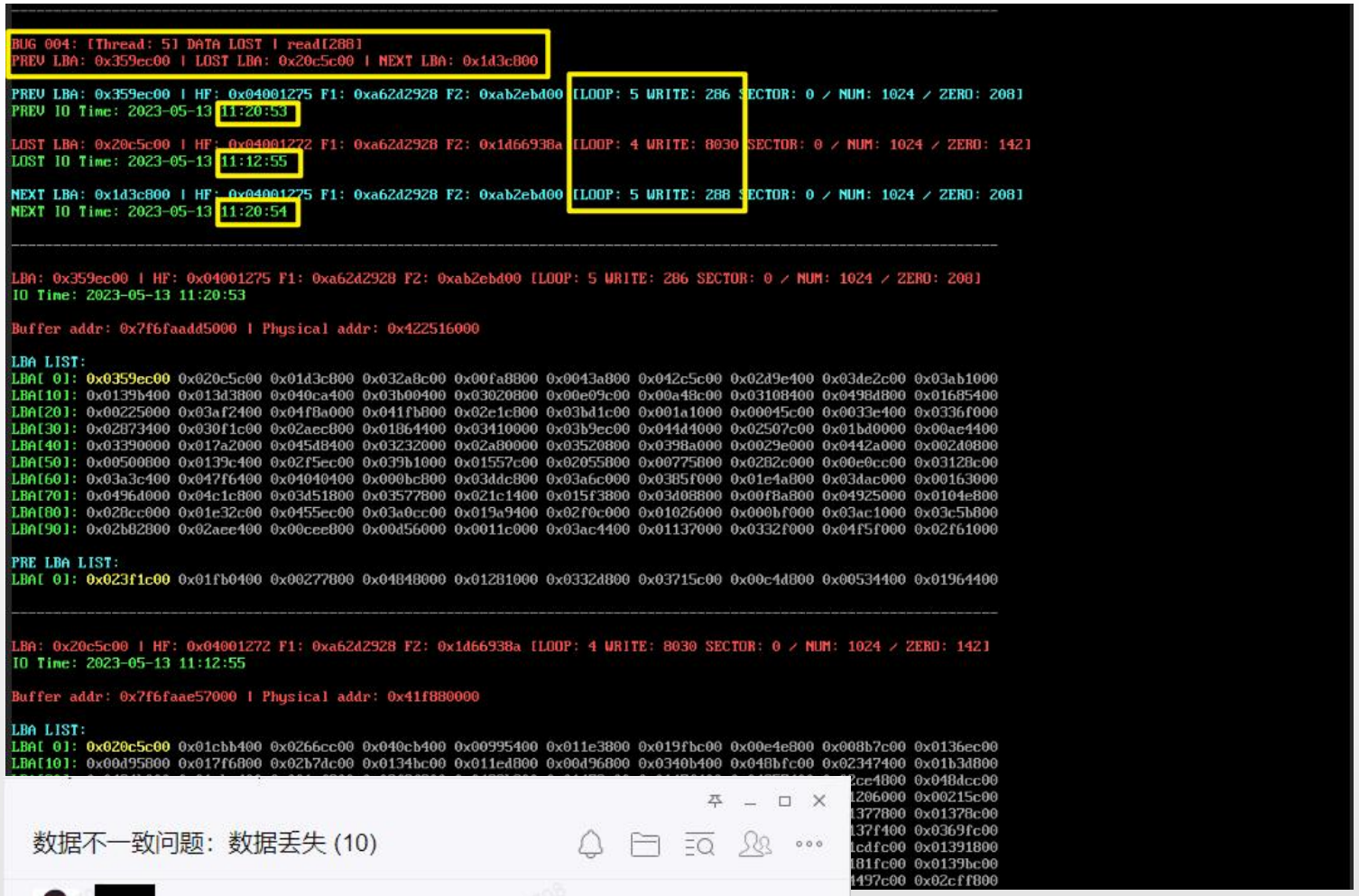
  第一次见到这种数据不一致出错的情形：数据错位一个字节！

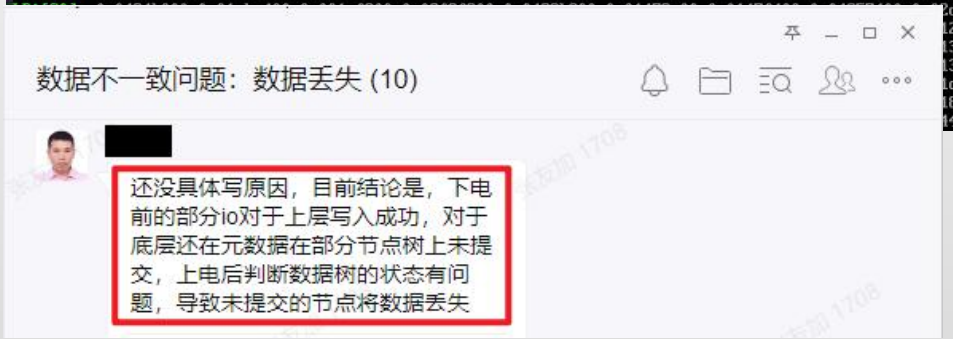


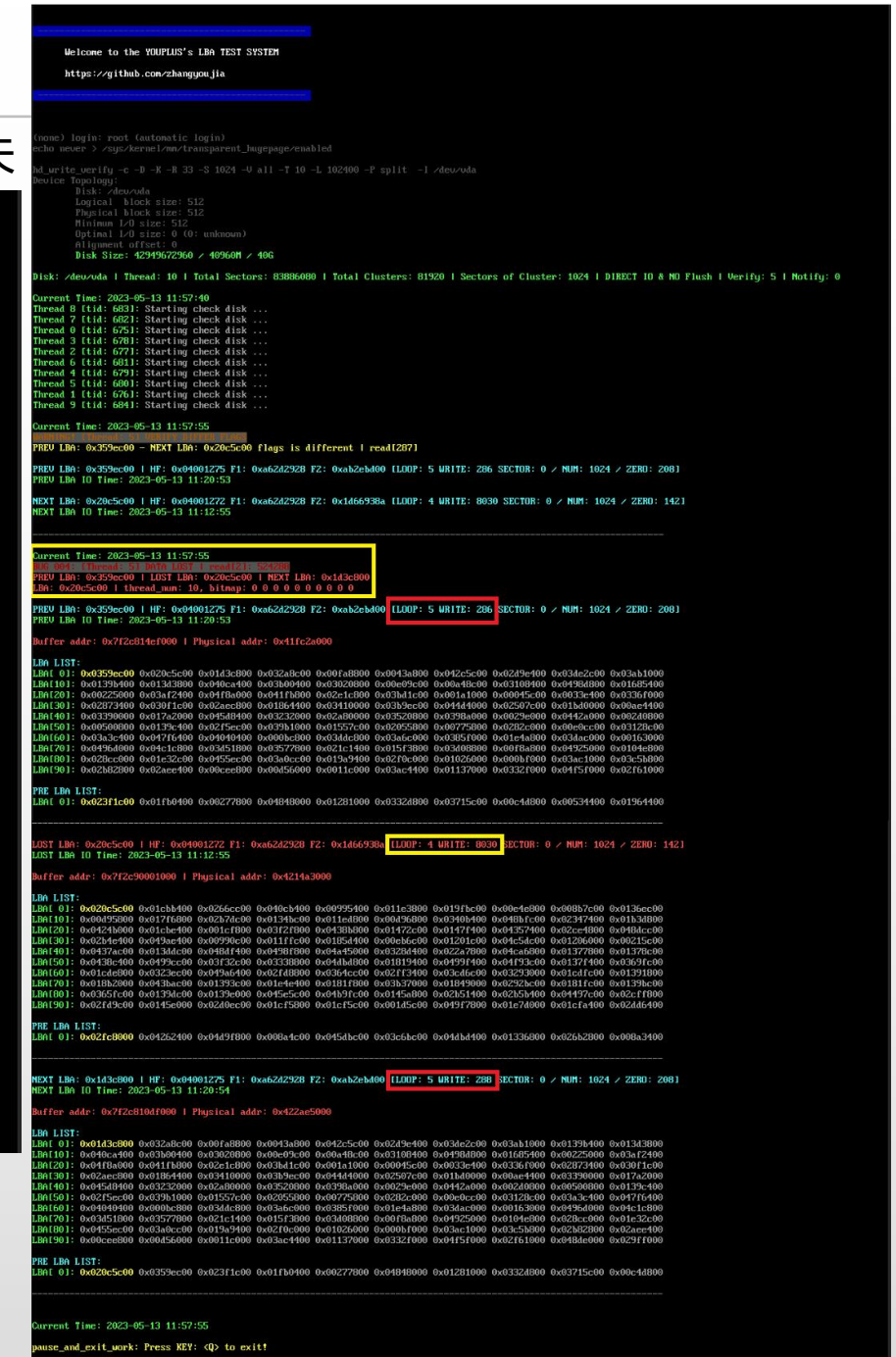


LBA工具介绍：数据不一致问题，案例三

- 【自研存储-全闪】虚拟机跑LBA工具测试出数据不一致问题--数据丢失







LBA工具介绍：数据不一致问题，案例四

- 【自研存储-全闪】虚拟机跑LBA工具测试出数据不一致问题2--数据丢失

```
[root@node81 ~]# daos obj dkey-query  -p testA -c c8e405fc-0f84-4d9c-b3b1-ea42a073b052 -i 939282005873393974.0 --dkey 1 --poolmap
/var/log/storage/pool_buf/4eaee947-2adb-4cde-9ac4-a7b27148b533_18_0
Dkey Layout: grp_nr: 1 grp_size: 4
  idx: 5 shardid: 4 rank: 3 tgt: 10 rebuild: 0 hyper: 0
  idx: 4 shardid: 3 rank: 1 tgt: 1 rebuild: 0 hyper: 0
  idx: 6 shardid: 5 rank: 5 tgt: 7 rebuild: 0 hyper: 0
  idx: 4294967295 shardid: 4294967295 rank: 4294967295 tgt: 4294967295 rebuild: 1 hyper: 0
[root@node81 ~]# daos obj dkey-query  -p testA -c c8e405fc-0f84-4d9c-b3b1-ea42a073b052 -i 939282005873393974.0 --dkey 1 --poolmap
/var/log/storage/pool_buf/4eaee947-2adb-4cde-9ac4-a7b27148b533_19_0
Dkey Layout: grp_nr: 1 grp_size: 3
  idx: 4 shardid: 4 rank: 3 tgt: 10 rebuild: 0 hyper: 0
  idx: 3 shardid: 3 rank: 2 tgt: 2 rebuild: 0 hyper: 0
  idx: 5 shardid: 5 rank: 5 tgt: 7 rebuild: 0 hyper: 0
node81 INFO 2023/05/25 20:07:19 [gid:137] daos_engine:1 Cluster map 4eaee947 version 19 reclaim 1 disable 0
root[0] v0 UP_IN 0
      protect domain[2] v1 UP_IN 0
            rack[3] v1 UP_IN 0
                  node[6] v1 UP_IN 0
                        rank[1] v1 UP_IN 0
                              target[0][1] v1 UP (fseq=11 out=14 up=6 upin=8 flags=1)
                              target[1][2] v1 UP_IN (fseq=1 out=1 up=1 upin=1 flags=0)
                        rank[2] v1 UP_IN 0
                              target[2][1] v1 UP_IN (fseq=10 out=12 up=18 upin=19 flags=0)
                              target[3][2] v1 UP_IN (fseq=1 out=1 up=1 upin=1 flags=0)
                  node[10] v1 UP_IN 0
                        rank[4] v1 UP_IN 0
                              target[4][1] v1 UP_IN (fseq=1 out=1 up=1 upin=1 flags=0)
node81 INFO 2023/05/25 20:07:19 [gid:137] daos_engine:1                              target[5][2] v1 UP_IN (fseq=1 out=1 up=1 upin=1 flags=0)
```

已有基本结论：jump算法导致的数据重构丢失，明天我找攀哥确认，环境先继续帮忙保留，明天确认后，我会通知环境释放

【问题影响】

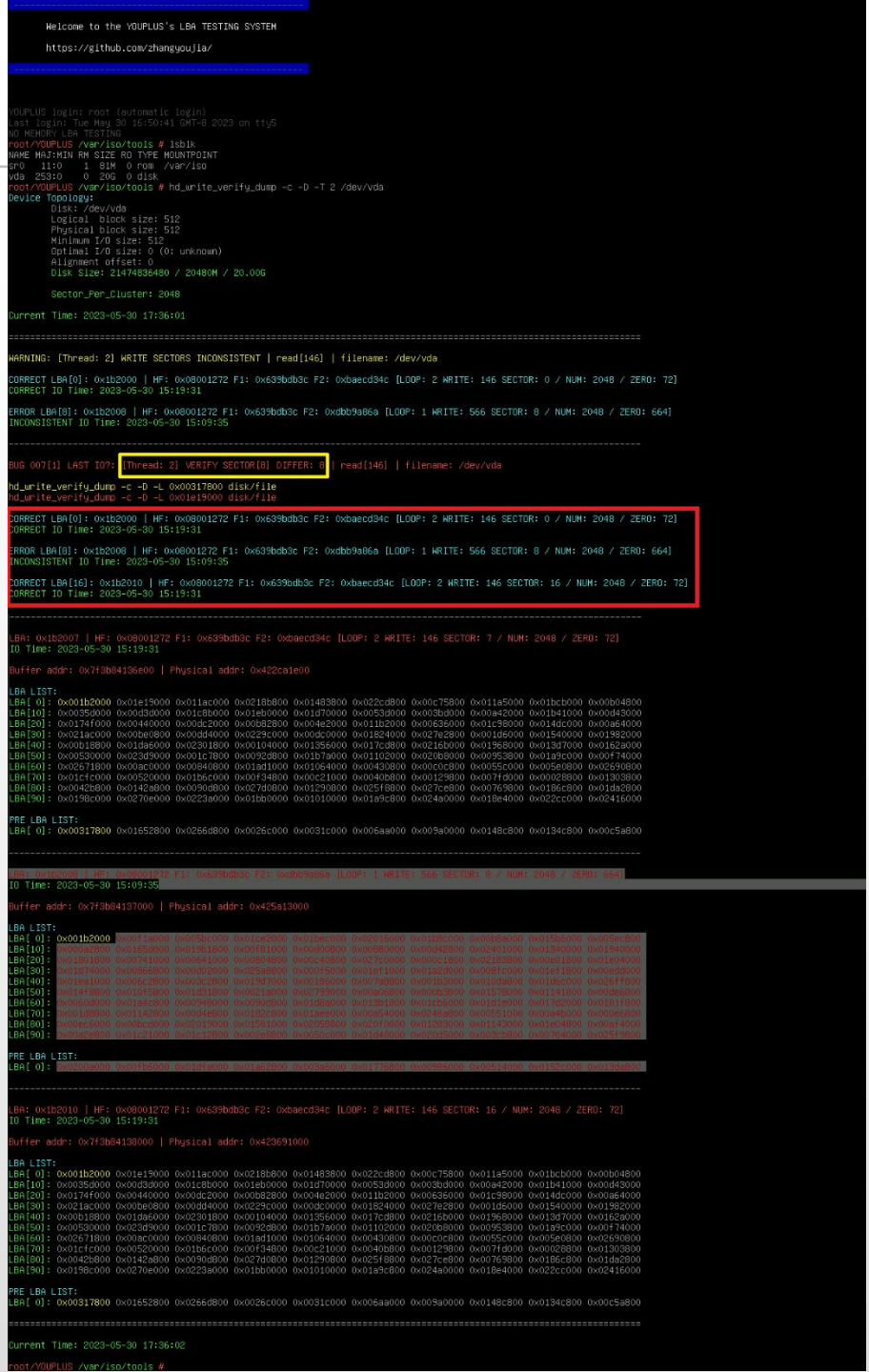
故障target后数据不一致。

【问题根因】

再允许不同group间允许重复，如果group中重复的target UP后，计算合并layout的时候只取值一次，导致选出的layout有问题。

【修改方案】

不同的group间允许多次选择，相同的group中不允许重复。



LBA工具介绍：数据不一致问题，案例五

- 【自研存储-混闪】修改IO及ULT超时阈值虚机跑LBA工具出数据不一致问题

5号线程写了212个IO，随机读取第99个IO的数据进行校验，发现簇LBA地址：0x4108800，簇内偏移1336扇区开始，连续622扇区的数据不一致(数据丢失)。

[redacted] 38分钟以前

上传了附件 139461开发自验报告模板.docx

【问题影响】

OLOG还未close就被punch，数据丢失

【问题根因】

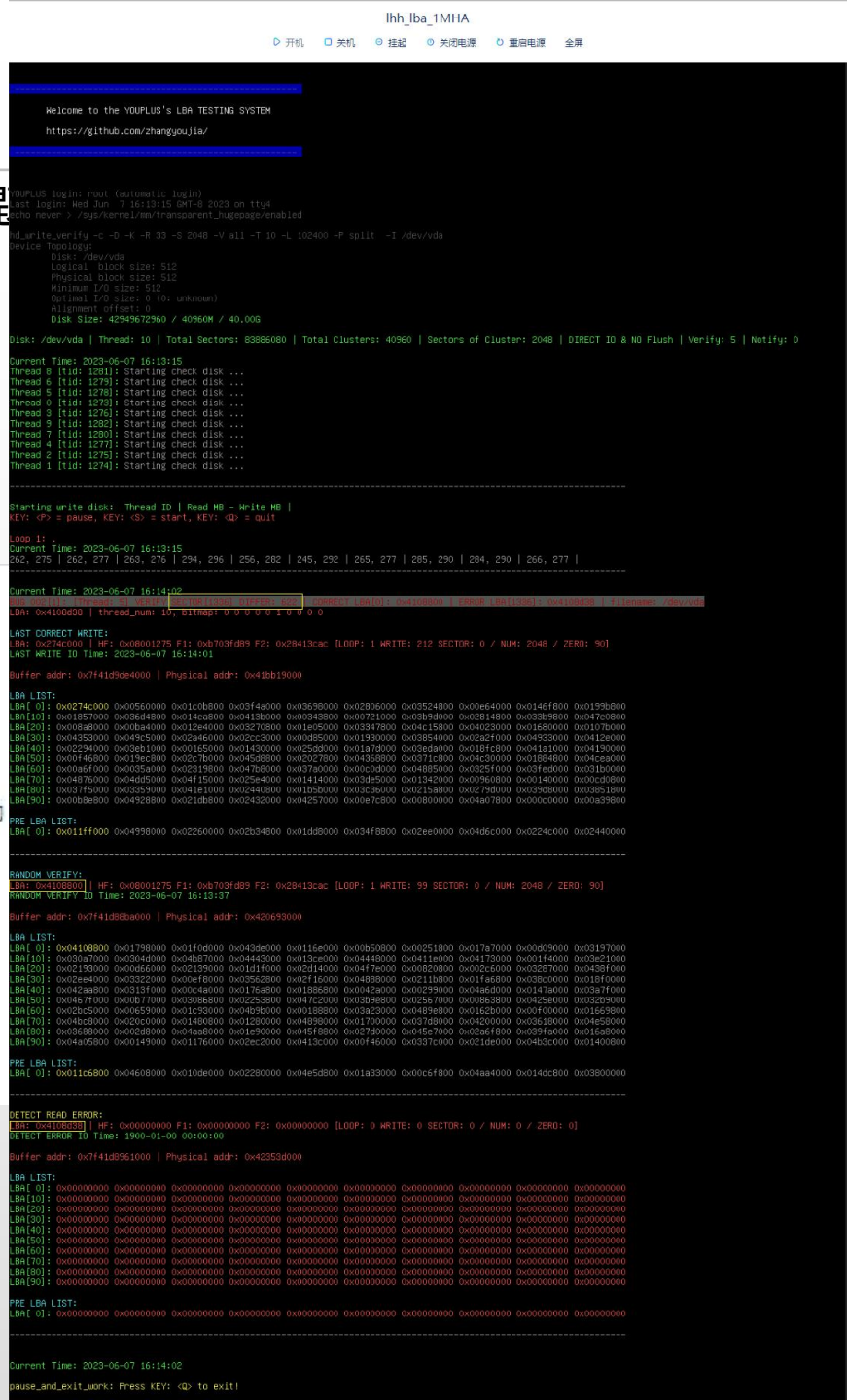
T1时刻GC兜底超过24 hours（可配置）将未close的olog的有效数据搬移后将olog加入到punch队列，T2时刻olog又发生写入，一段时间后T3时刻olog被真正punch，T2~T3时刻的数据被punch，导致后续读到0，丢数据

【修改方案】

OLOG 聚合写入流程超过12hours后将olog强制关闭，防止触发24hours超时punch机制

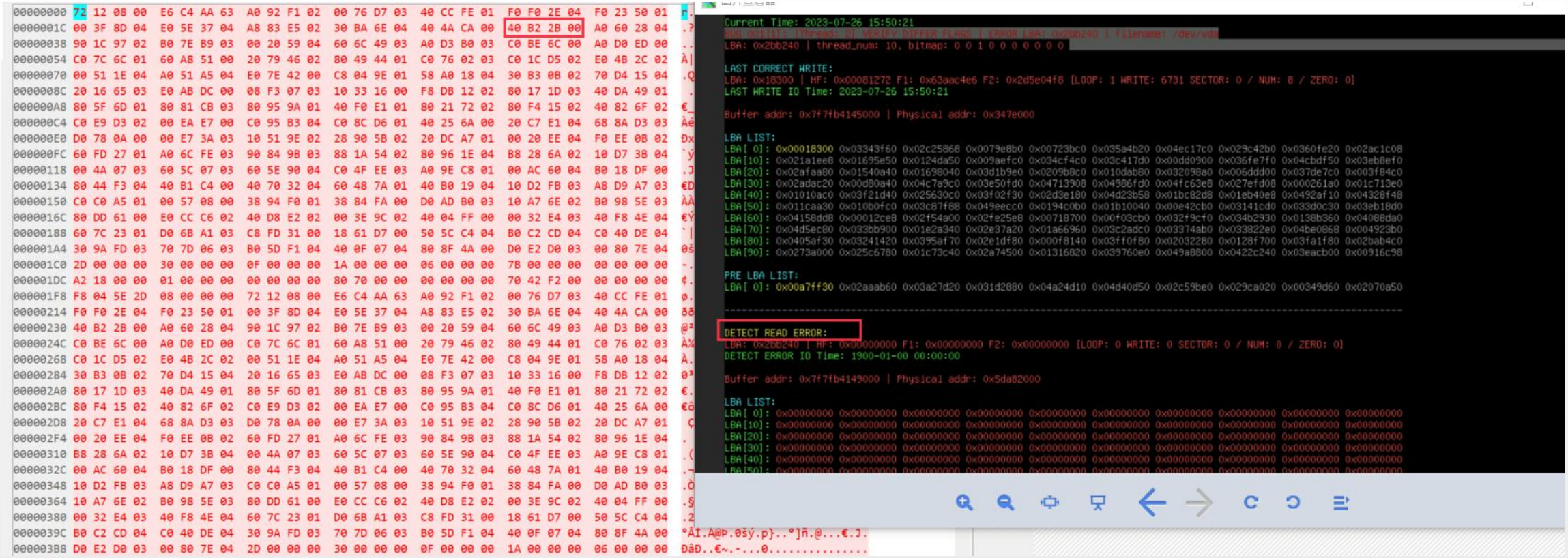
【问题存在版本】

row场景下20230611前的主线版本



LBA工具介绍：数据不一致问题，案例六

- 【自研存储-混闪】连续扩盘连续删盘虚拟机跑出LBA问题



2号线程，随机数据校验：读取前面已经成功写入并落盘的数据来进行校验，发现数据错误--全0数据。

2023-09-06 17:19:52

【问题原因】iter next误将二级指针判断为一级，导致dtx被误认为abort，重构拉取数据时，无法拉取到被abort（实际重试是成功，只是中间有abort）的数据

【修改方案】iter next需要识别指针状态，并且在insert插入使用reuse时,需要初始化evt_node_entry_at_re

LBA工具介绍：上海XXX公司使用LBA工具

- 虚拟机中跑LBA工具的小IO测试，簇内校验发现有一扇区数据丢失

[redacted]

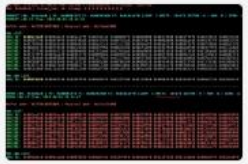
@YOUPLUS 不好意思打搅一下， 试用 LBA 工具的过程中发现一个问题， LBA 工具测试的结果会受上一次测试的影响吗？从报出的 BUG 来看， LOOP 10 读到了 LOOP 9 的数据：

[redacted]

不会受上一次测试的影响，这就是测试出了数据不一致问题：有数据丢失了

6号线程，第10轮测试，簇内校验，第35208个IO中的第1扇区的数据丢失了。

[redacted]



134174 这个簇的数据貌似被 134175 覆盖了

构造数据的时候会去读盘么？

这次的问题就比上次的问题严重了，上次是丢失了数据，这次的数据完全被写到了不应该在的位置了。

会不会是 LBA 工具写串了

134174 和 134175 两个 IO 簇的数据是一样的

不用怀疑LBA工具了，还是排查一下存储方向的原因吧，估计用fio/vdbench等工具也能测试出这个问题。

fio 没跑出来，所以有这个疑问

YOUPLUS：不用怀疑LBA工具了，还是排查一下存储方向的原因吧，估计用fio/vdbench等工具也能...

你用的是什么存储啊？要不你先在一个本地存储(ext4/xfs)上跑同样的虚拟机测试对比一下就知道了。

这两个IO的时间就不太正常

我这边应该是存储层面的问题

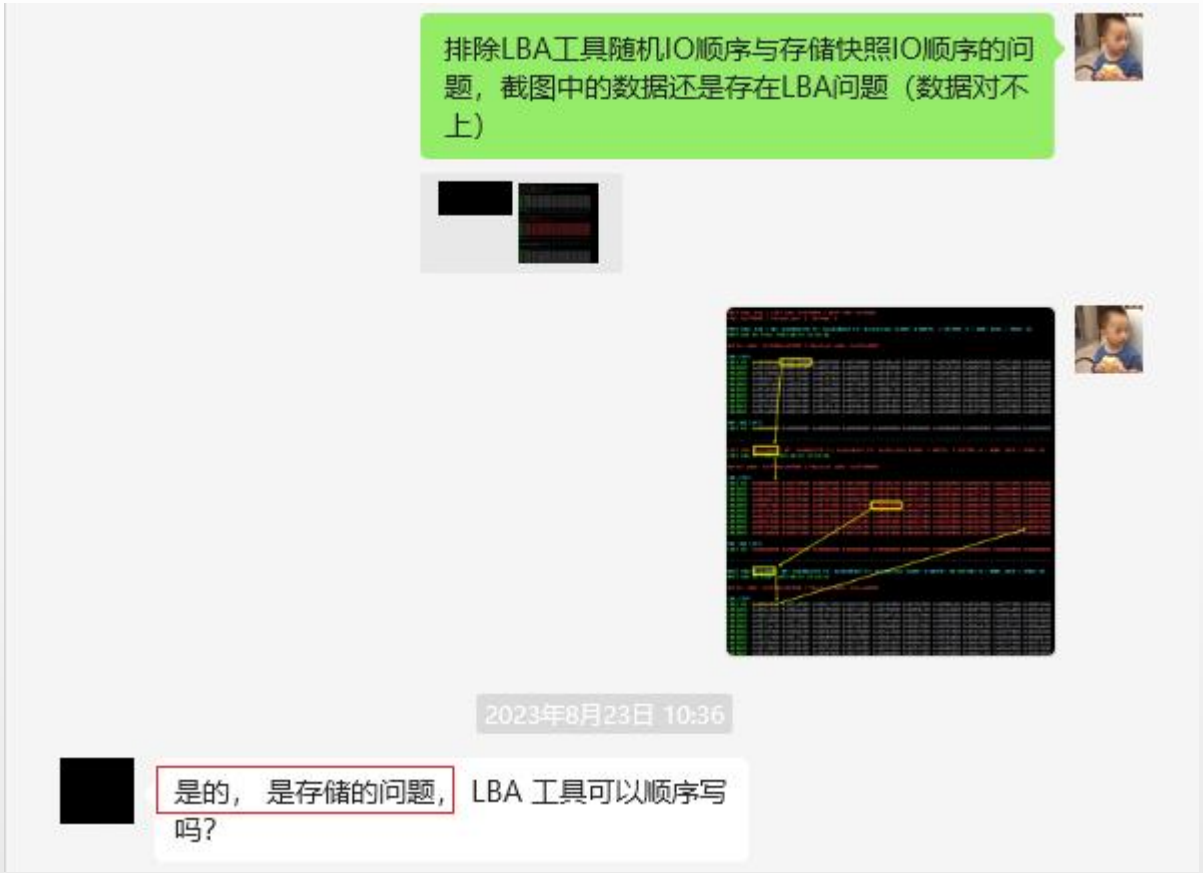
正在看存储日志

👌

所以LBA工具比fio/vdbench的verify功能强大多了，轻易就测试出来了LBA问题，但fio工具却测试不出来。

LBA工具介绍：北京XXX公司使用LBA工具

- 对全闪NVMeOF存储卷打快照，LBA工具测试出快照中的数据不一致



另外：

- 北京XXX公司、XXX软件公司等在使用LBA工具；
- 目前已有七十多个公司在使用LBA工具及自动化测试系统；
- 与openGuass社区总工确定合作意向；也有国外使用者；
- Intel SPDK上海团队可能在试用LBA工具，也可能没有，没有收到反馈；

LBA工具介绍：工具的作用与价值

- qemu-2.5版本热迁移代码原生BUG：排查、修复此问题代价较大 -- 两个资深的研发花了三周左右时间

1. 2020年2月份(新冠疫情爆发期间)，多台虚拟机跑LBA自动化测试系统并进行热迁移，偶现内存热迁移数据不一致问题；
2. 批量虚拟机跑热迁移(加大主机节点负载)，增加复现概率；
3. 排查手段：LBA工具 + 详细的调试日志 + mprotect抓取篡改内存现场；

```
[redacted] 03-04 15:44:17
[2020-03-04]
一、当前的情况：
1、早上热迁移winpe和linux虚拟机各出现一次lba，调试日志明确记录了这次内存数据被正确发送到目的端并保存，但是guest运行后读对应位置内存时，数据却是错误的。
2、分析了guest的内存，是1M buff中间一个4K内存页损坏，而前后的内存页数据是正确的，与调试日志对得上。
3、目前初步怀疑是目的端虚拟机启动阶段，某个流程把guest内存页4K数据破坏掉，关键阶段的时间线如下，在02:18:00.364045到02:18:09.273347这段时间，内存被破坏。
2020-03-04 02:18:00.364045  目的端qemu接收到对应内存页并保存，调试日志显示此内存页数据是正确的
2020-03-04 02:18:00.464607  内存迁移全部完成
2020-03-04 02:18:09.203232  热迁移完成，目的端虚拟机开始运行
2020-03-04 02:18:09.273347  测试工具检测数据错误，报告出现lba

二、下一步计划
1、在目的端虚拟机启动前的几个阶段，分别将guest内存文件保存下来，这样可以进一步判断内存页数据是在哪个阶段被破坏。
2、添加调试代码，调用mprotect接口将guest物理内存对应的qemu虚拟地址空间设为只读，如果有流程破坏内存，就可以被抓到。

[redacted] 03-04 18:15:11
有一种场景需要注意：
1、现在测试都是使用共享内存，ksm模块不会进行内存回收；
2、如果使用普通内存，ksm是会时进行相同页内存回收的(特别是全0页)，需要在这种场景下跑LBA工具进行热迁移测试，会不会也存在LBA的问题？

三、结论：
1、lba工具测试虚拟机热迁移，发现内存数据不一致；
2、在qemu的热迁移源端和目的端的代码中加详细的调试日志，确认所有内存数据被正确发送到目的端并保存，但是目的端guest运行后读对应位置内存时，LBA工具却发现有些内存数据是错误的；
3、在qemu的热迁移目的端的代码中调用mprotect接口将guest物理内存对应的qemu虚拟地址空间设为只读，在目的端虚拟机运行之前，部分内存数据被篡改并主动触发断言出core；
4、qemu热迁移原生bug，内存热迁移每轮迭代没有对解压缩线程做同步等待，如果解压缩线程执行速度很慢，在内存热迁移已经整体完成后，解压缩线程还在向guest写入旧数据，导致guest内存损坏。

Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `/usr/bin/kvm -id 227587371880 -chardev socket,id=qmp,path=/var/run/qemu-server/'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x00007f5ee420ee70 in inflate_fast () from /lib/x86_64-linux-gnu/libz.so.1
[Current thread is 1 (Thread 0x7f5e503fe700 (LWP 12496))]

(gdb) bt
#0  0x00007f5ee420ee70 in inflate_fast () from /lib/x86_64-linux-gnu/libz.so.1
#1  0x00007f5ee421116e in inflate () from /lib/x86_64-linux-gnu/libz.so.1
#2  0x00007f5ee42150b1 in uncompress2 () from /lib/x86_64-linux-gnu/libz.so.1
#3  0x00007f5ee4215193 in uncompress () from /lib/x86_64-linux-gnu/libz.so.1
#4  0x0000561b5b0b3058 in do_data_decompress (opaque=0x561b5d1453f0) at /home/test/qemu-2.5.1/migration/ram.c:2341
#5  0x00007f5eddc8bb50 in start_thread () from /lib/x86_64-linux-gnu/libpthread.so.0
#6  0x00007f5edd9d5a7d in clone () from /lib/x86_64-linux-gnu/libc.so.6
#7  0x0000000000000000 in ?? ()
```

LBA工具介绍：工具的作用与价值

- 簇IO随机拆分功能也测试发现了qemu-2.0.1版本virtio-blk磁盘IO合并功能存在原生bug

```
static int multiwrite_merge(BlockDriverState *bs, BlockRequest *reqs,
    int num_reqs, MultiwriteCB *mcb)
{
    int i, outidx;

    // Sort requests by start sector
    qsort(reqs, num_reqs, sizeof(*reqs), &multiwrite_req_compare);

    // Check if adjacent requests touch the same clusters. If so, combine them,
    // filling up gaps with zero sectors.
    outidx = 0;
    for (i = 1; i < num_reqs; i++) {
        int merge = 0;
        int64_t oldreq_last = reqs[outidx].sector + reqs[outidx].nb_sectors;

        // Handle exactly sequential writes and overlapping writes.
        if (reqs[i].sector <= oldreq_last) {
            merge = 1;
        }

        if (reqs[outidx].qiov->niov + reqs[i].qiov->niov + 1 > IOV_MAX) {
            merge = 0;
        }

        if (merge) {
            size_t size;
            QEMUIOVector *qiov = g_malloc0(sizeof(*qiov));
            qemu_iovec_init(qiov,
                reqs[outidx].qiov->niov + reqs[i].qiov->niov + 1);

            // Add the first request to the merged one. If the requests are
            // overlapping, drop the last sectors of the first request.
            size = (reqs[i].sector - reqs[outidx].sector) << 9;
            qemu_iovec_concat(qiov, reqs[outidx].qiov, 0, size);

            // We should need to add any zeros between the two requests
            assert (reqs[i].sector <= oldreq_last);

            // Add the second request
            qemu_iovec_concat(qiov, reqs[i].qiov, 0, reqs[i].qiov->size);

            reqs[outidx].nb_sectors = qiov->size >> 9;
            reqs[outidx].qiov = qiov;

            mcb->callbacks[i].free_qiov = reqs[outidx].qiov;
        } else {
            outidx++;
            reqs[outidx].sector     = reqs[i].sector;
            reqs[outidx].nb_sectors = reqs[i].nb_sectors;
            reqs[outidx].qiov       = reqs[i].qiov;
        }
    }

    return outidx + 1;
}
```

目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

LBA工具实现原理

- 磁盘、簇、扇区、全局bitmap(内存)

磁盘

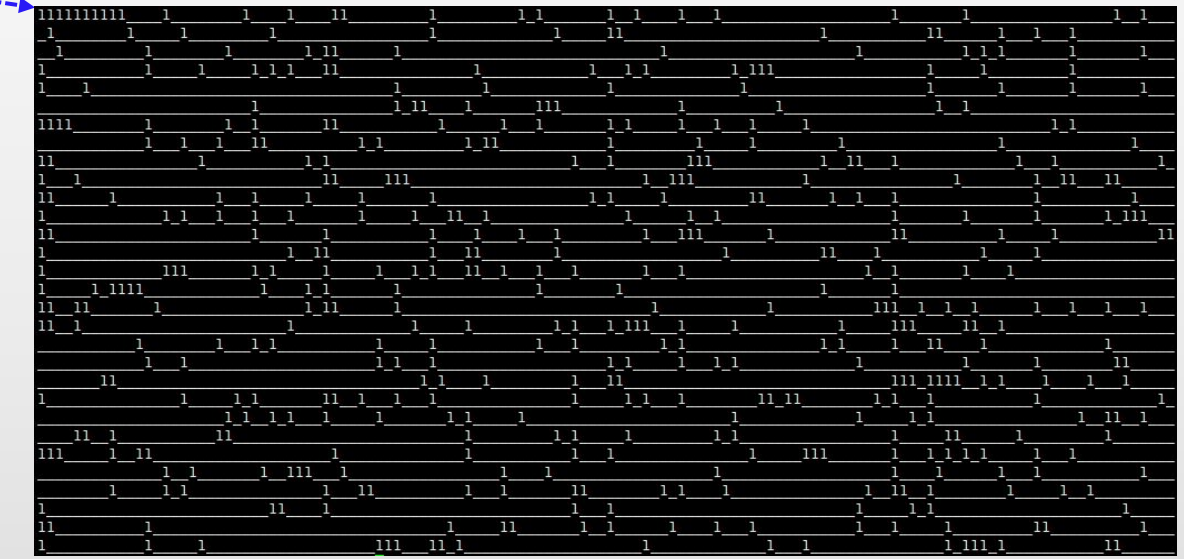
测试/校验固定起始簇

簇

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	...	M

全局bitmap(内存)

bitmapt.txt



cluster

sector | sector | sector | sector | sector | sector

扇区

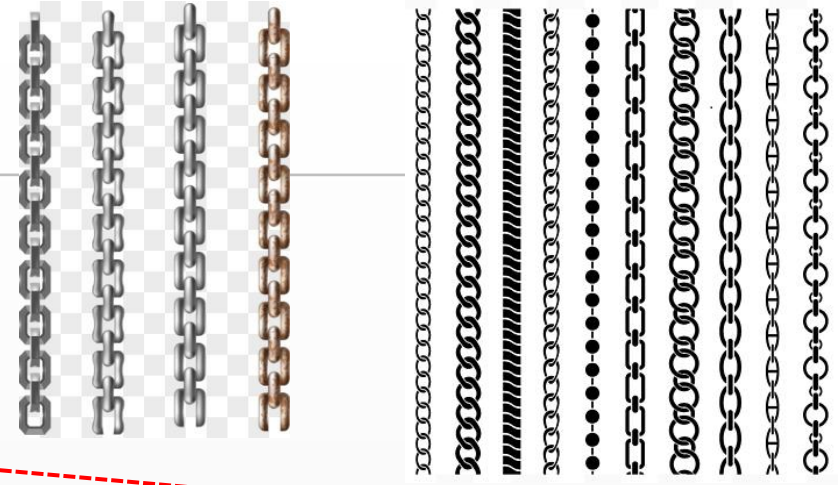
HF	HF	HF	HF	HF	HF
MAGIC	MAGIC	MAGIC	MAGIC	MAGIC	MAGIC
PRE[10]	PRE[10]	PRE[10]	PRE[10]	PRE[10]	PRE[10]
LBA[100]	LBA[100]	LBA[100]	LBA[100]	LBA[100]	LBA[100]
TIME	TIME	TIME	TIME	TIME	TIME
RANDOM	RANDOM	RANDOM	RANDOM	RANDOM	RANDOM
NUM	NUM	NUM	NUM	NUM	NUM
INDEX：0	INDEX：1	INDEX：2	INDEX：3	INDEX：...	INDEX: N-1

1. 每个磁盘/文件被划分为M个IO数据簇；
2. 每个IO数据簇包含N个扇区；
3. IO数据簇N个扇区中的数据：每个扇区除index字段表示簇中的扇区序号外，其它数据均相同

4. LBA工具测试磁盘/文件时，为每个磁盘/文件分配全局bitmap内存，并置为全0；
5. 每个IO数据簇不会被重复分配，分配新IO数据簇时，随机分配bitmap未被置1的对应数据簇；
6. 每个已分配的IO数据簇对应的bitmap被置位为1；

LBA工具实现原理

链条



- 多线程、随机线程(高级功能)

磁盘

测试/校验固定起始簇

随机IO

簇

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	...	M

多线程

- 线程0
- 线程1
- 线程2
- 线程3
- 线程4
- 线程5
- 线程6
- 线程7
- 线程8
- 线程9

随机数据校验

- 线程0的bitmap
- 线程1的bitmap
- 线程2的bitmap
- 线程3的bitmap
- 线程4的bitmap
- 线程5的bitmap
- 线程6的bitmap
- 线程7的bitmap
- 线程8的bitmap
- 线程9的bitmap

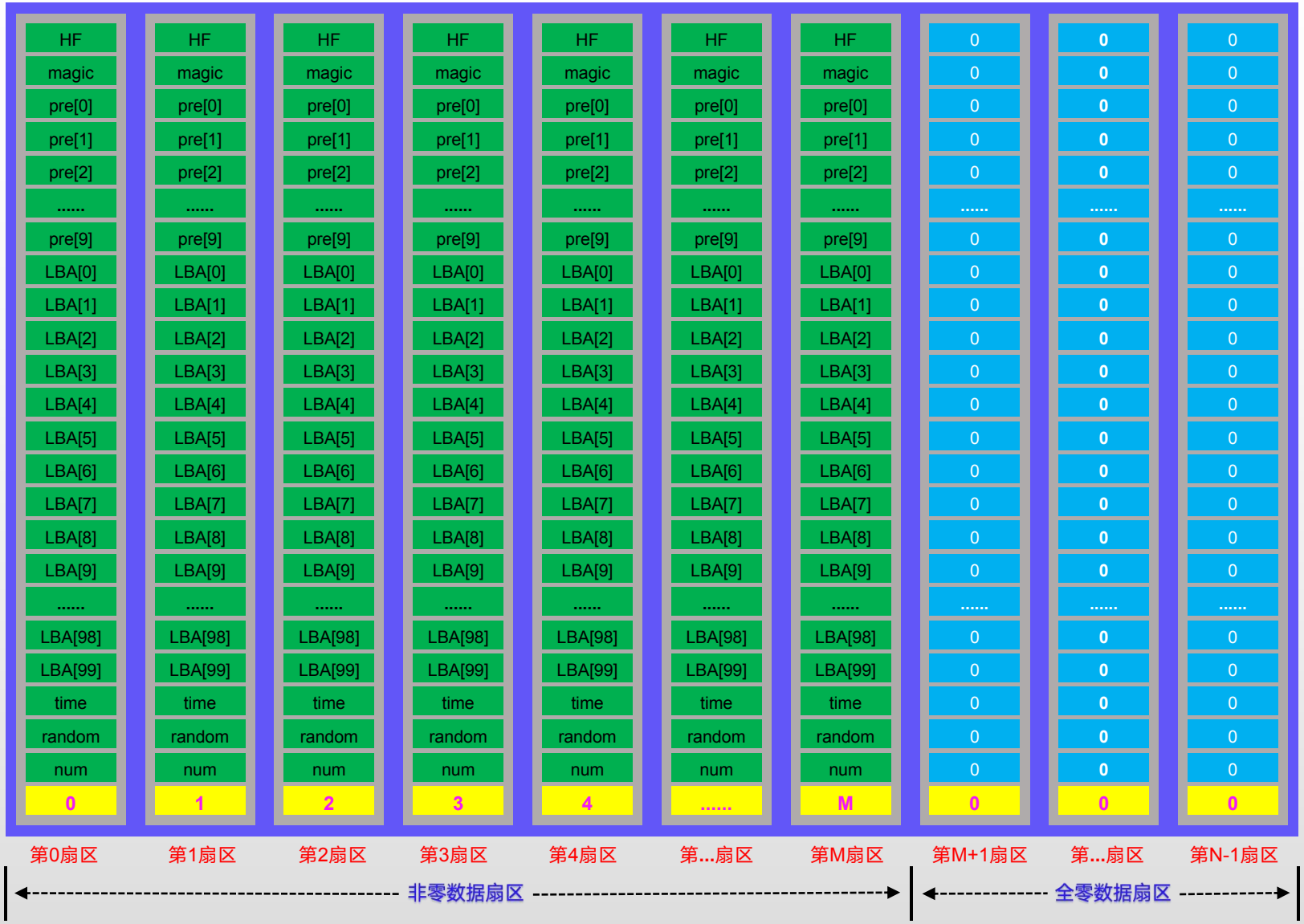
LBA工具实现原理：数据结构

- 强双向链表、数组、队列混合体：5个数组、2个链表、2个队列

```
DATA LAYOUT:
================================================

offsetof HF:                    0    [4   bytes]
offsetof MAGIC:                 4    [4   bytes]
offsetof PRE_LBA[10]:           8    [40  bytes]
offsetof LBA_LIST[100]:         48   [400 bytes]
offsetof TIME:                  448  [56  bytes]
offsetof RANDOM:                504  [4   bytes]
offsetof NUM:                   508  [2   bytes]
offsetof INDEX:                 510  [2   bytes]

================================================
```

```
[LOOP: 1 WRITE: 101 SECTOR: 0 / NUM: 1024 / ZERO: 322]
```

1. 数组1：一个IO数据簇包括N(1024)个扇区，如右图和上图
2. 数组2：包括M+1(702)个扇区的非零数据
3. 数组3：包括N-M-1(322)个扇区的全零数据
4. 数组4：每个扇区中的PRE_LBA[10]，如上图
5. 数组5：每个扇区中的LBA_LIST[100]，如上图
6. 队列1：每个扇区中的PRE_LBA[10]，如下一页
7. 队列2：每个扇区中的LBA_LIST[100]，如下一页
8. 链表1：PRE_LBA[10]前向链表，如下一页
9. 链表2：LBA_LIST[100]后向链表，如下一页
   PRE_LBA[10]和LBA_LIST[100]形成强双向链表

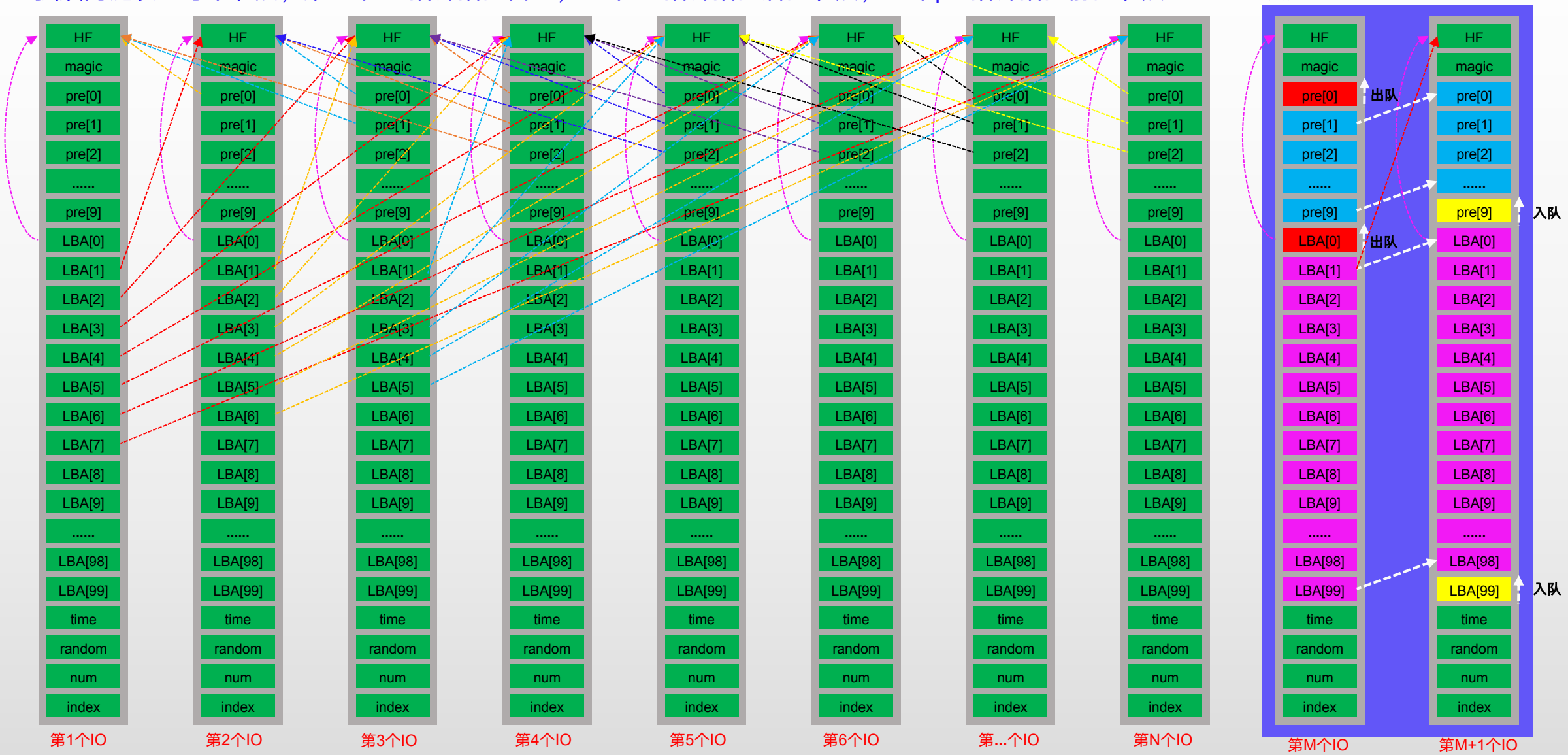


LBA工具实现原理：无状态

- 强双向链表、队列

强双向链表：每个节点，第一个lba指针指向自已，99个lba指针指向后向节点，10个pre指针指向前向节点

队列：相邻的IO簇



LBA工具实现原理

- 强双向链表、队列、数组（**为什么要设计成强双向链表的形式？**）

强双向链表也可以理解为是网状，每个数据与附近的数据之间形成强相关的关系，只要数据出错了（哪怕是数据只出错一个字节）
LBA工具很容易知道出错数据的偏移位置和大小、时间等信息，例如：渔民的渔网破了一个洞，渔民很容易知道渔网上破洞的位置和大小。

```
LBA: 0x0 | HF: 0x04001270 F1: 0x0892947a F2: 0xcab47c88 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 322]
IO Time: 2023-07-12 15:36:04

Buffer addr: 0x12e0000 | Physical addr: 0x58277a00000

                第1个IO     第2个IO     第3个IO     第4个IO     第5个IO     第6个IO     第7个IO     第8个IO     第9个IO     第10个IO
LBA LIST:
LBA[ 0]: 0x00000000 0x007ed000 0x00112800 0x00652000 0x00161000 0x0037b800 0x00460000 0x00489800 0x00159000 0x0016e800
LBA[10]: 0x00755000 0x004f9c00 0x002aec00 0x0063a400 0x00653800 0x00051000 0x00333400 0x007cf800 0x00433000 0x00554000
LBA[20]: 0x007fe800 0x00247000 0x0077c800 0x002e0000 0x002d4c00 0x0027f000 0x0002b800 0x0021c400 0x0015c000 0x00600000
LBA[30]: 0x0032bc00 0x0022bc00 0x00463400 0x006fe000 0x007cd400 0x0047b000 0x005a6800 0x00727000 0x0063d800 0x0028e800
LBA[40]: 0x000a8800 0x006bb800 0x00127000 0x001b3400 0x00020000 0x002a9800 0x00522000 0x00365000 0x000e4000 0x00204400
LBA[50]: 0x001bd000 0x001e3800 0x007c7000 0x002b3c00 0x004b8400 0x001cb800 0x00611c00 0x002edc00 0x00377800 0x00174800
LBA[60]: 0x005be800 0x00053c00 0x00095000 0x00461000 0x00511000 0x00708000 0x002f3000 0x00740800 0x007d1000 0x002e2000
LBA[70]: 0x00307400 0x005a9000 0x00481800 0x0041cc00 0x004eb800 0x003fa800 0x00782400 0x007c2400 0x00228800 0x004e2c00
LBA[80]: 0x0067d800 0x0011dc00 0x00436400 0x00420000 0x00650400 0x00684400 0x00767c00 0x006f2000 0x00186400 0x002bd400
LBA[90]: 0x004d2800 0x007a4c00 0x00284000 0x0073f000 0x00720000 0x006f9c00 0x004d9000 0x0008b400 0x005dc800 0x0070a800
                第91个IO    第92个IO    第93个IO    第94个IO    第95个IO    第96个IO    第97个IO    第98个IO    第99个IO    第100个IO
PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
```

```
LBA: 0x7ed000 | HF: 0x04001270 F1: 0x0892947a F2: 0xcab47c88 [LOOP: 1 WRITE: 2 SECTOR: 0 / NUM: 1024 / ZERO: 322]
IO Time: 2023-07-12 15:36:04

Buffer addr: 0xffff86170000 | Physical addr: 0x500911c0000

                第2个IO     第3个IO     第4个IO     第5个IO     第6个IO     第7个IO     第8个IO     第9个IO     第10个IO    第11个IO
LBA LIST:
LBA[ 0]: 0x007ed000 0x00112800 0x00652000 0x00161000 0x0037b800 0x00460000 0x00489800 0x00159000 0x0016e800 0x00755000
LBA[10]: 0x004f9c00 0x002aec00 0x0063a400 0x00653800 0x00051000 0x00333400 0x007cf800 0x00433000 0x00554000 0x007fe800
LBA[20]: 0x00247000 0x0077c800 0x002e0000 0x002d4c00 0x0027f000 0x0002b800 0x0021c400 0x0015c000 0x00600000 0x0032bc00
LBA[30]: 0x0022bc00 0x00463400 0x006fe000 0x007cd400 0x0047b000 0x005a6800 0x00727000 0x0063d800 0x0028e800 0x000a8800
LBA[40]: 0x006bb800 0x00127000 0x001b3400 0x00020000 0x002a9800 0x00522000 0x00365000 0x000e4000 0x00204400 0x001bd000
LBA[50]: 0x001e3800 0x007c7000 0x002b3c00 0x004b8400 0x001cb800 0x00611c00 0x002edc00 0x00377800 0x00174800 0x005be800
LBA[60]: 0x00053c00 0x00095000 0x00461000 0x00511000 0x00708000 0x002f3000 0x00740800 0x007d1000 0x002e2000 0x00307400
LBA[70]: 0x005a9000 0x00481800 0x0041cc00 0x004eb800 0x003fa800 0x00782400 0x007c2400 0x00228800 0x004e2c00 0x0067d800
LBA[80]: 0x0011dc00 0x00436400 0x00420000 0x00650400 0x00684400 0x00767c00 0x006f2000 0x00186400 0x002bd400 0x004d2800
LBA[90]: 0x007a4c00 0x00284000 0x0073f000 0x00720000 0x006f9c00 0x004d9000 0x0008b400 0x005dc800 0x0070a800 0x00204800
                第92个IO    第93个IO    第94个IO    第95个IO    第96个IO    第97个IO    第98个IO    第99个IO    第100个IO   第101个IO
PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
                第1个IO
```

dump第1个IO和第2个IO的数据

```
LBA: 0x112800 | HF: 0x04001270 F1: 0x0892947a F2: 0xcab47c88 [LOOP: 1 WRITE: 3 SECTOR: 0 / NUM: 1024 / ZERO: 322]
IO Time: 2023-07-12 15:36:04

Buffer addr: 0xffff95640000 | Physical addr: 0x5814e6b0000

                第3个IO     第4个IO     第5个IO     第6个IO     第7个IO     第8个IO     第9个IO     第10个IO    第11个IO    第12个IO
LBA LIST:
LBA[ 0]: 0x00112800 0x00652000 0x00161000 0x0037b800 0x00460000 0x00489800 0x00159000 0x0016e800 0x00755000 0x004f9c00
LBA[10]: 0x002aec00 0x0063a400 0x00653800 0x00051000 0x00333400 0x007cf800 0x00433000 0x00554000 0x007fe800 0x00247000
LBA[20]: 0x0077c800 0x002e0000 0x002d4c00 0x0027f000 0x0002b800 0x0021c400 0x0015c000 0x00600000 0x0032bc00 0x0022bc00
LBA[30]: 0x00463400 0x006fe000 0x007cd400 0x0047b000 0x005a6800 0x00727000 0x0063d800 0x0028e800 0x000a8800 0x006bb800
LBA[40]: 0x00127000 0x001b3400 0x00020000 0x002a9800 0x00522000 0x00365000 0x000e4000 0x00204400 0x001bd000 0x001e3800
LBA[50]: 0x007c7000 0x002b3c00 0x004b8400 0x001cb800 0x00611c00 0x002edc00 0x00377800 0x00174800 0x005be800 0x00053c00
LBA[60]: 0x00095000 0x00461000 0x00511000 0x00708000 0x002f3000 0x00740800 0x007d1000 0x002e2000 0x00307400 0x005a9000
LBA[70]: 0x00481800 0x0041cc00 0x004eb800 0x003fa800 0x00782400 0x007c2400 0x00228800 0x004e2c00 0x0067d800 0x0011dc00
LBA[80]: 0x00436400 0x00420000 0x00650400 0x00684400 0x00767c00 0x006f2000 0x00186400 0x002bd400 0x004d2800 0x007a4c00
LBA[90]: 0x00284000 0x0073f000 0x00720000 0x006f9c00 0x004d9000 0x0008b400 0x005dc800 0x0070a800 0x00204800 0x006bf800
                第93个IO    第94个IO    第95个IO    第96个IO    第97个IO    第98个IO    第99个IO    第100个IO   第101个IO   第102个IO
PRE LBA LIST:
LBA[ 0]: 0x007ed000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
                第2个IO     第1个IO
```

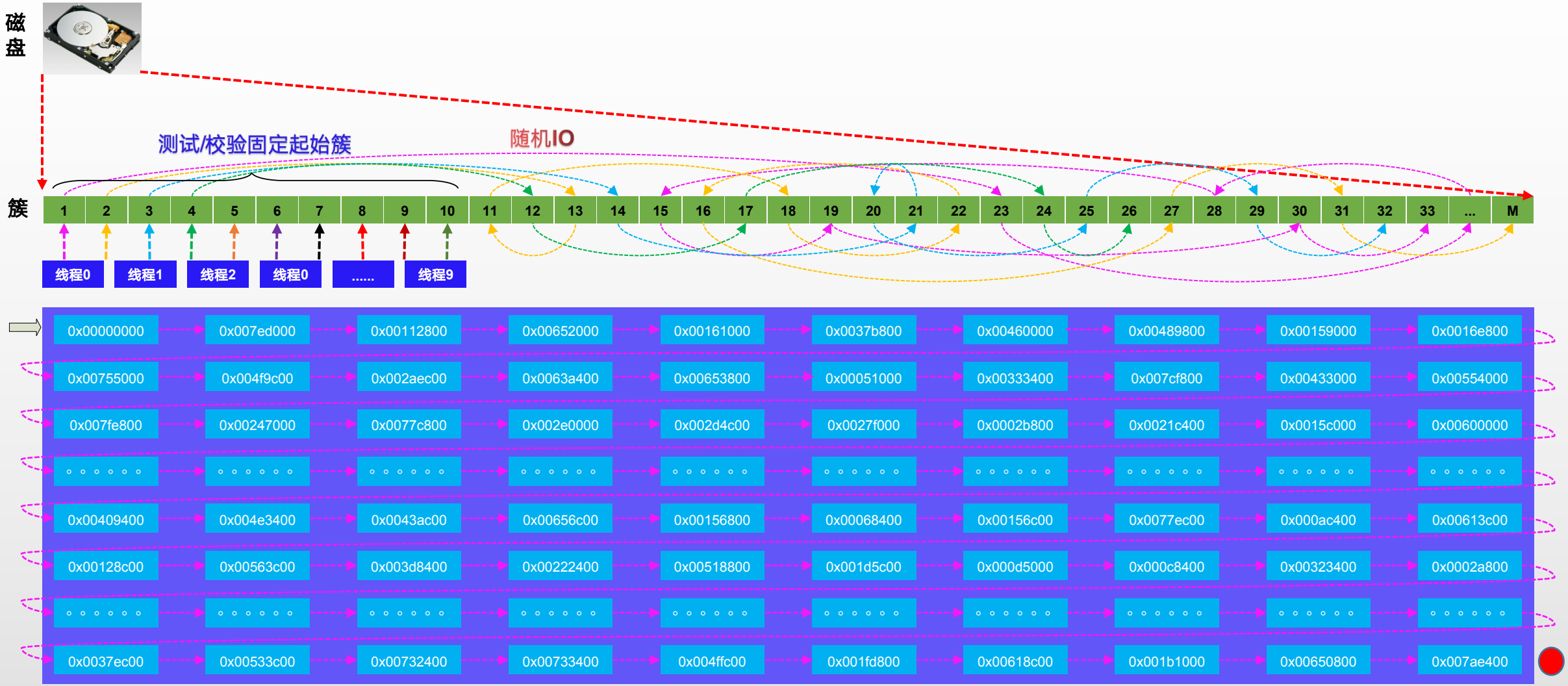
dump第3个IO的数据

```
LBA: 0x204800 | HF: 0x04001270 F1: 0x0892947a F2: 0xcab47c88 [LOOP: 1 WRITE: 101 SECTOR: 0 / NUM: 1024 / ZERO: 322]
IO Time: 2023-07-12 15:36:13

Buffer addr: 0xffffaf2d0000 | Physical addr: 0x58398310000

                第101个IO   第102个IO   第103个IO   第104个IO   第105个IO   第106个IO   第107个IO   第108个IO   第109个IO   第110个IO
LBA LIST:
LBA[ 0]: 0x00204800 0x006bf800 0x0076d400 0x001d7800 0x00309000 0x000f8000 0x004aac00 0x00422400 0x0010a400 0x0029e800
LBA[10]: 0x002a6c00 0x00312400 0x0032d000 0x0015fc00 0x00565400 0x000e1c00 0x000ec400 0x00716000 0x004e1c00 0x002b0400
LBA[20]: 0x000f3c00 0x0026a800 0x006fb000 0x0061b000 0x00301c00 0x004ee000 0x0010a800 0x0070e000 0x0015b000 0x006fe400
LBA[30]: 0x00393000 0x0013a400 0x001a3400 0x0016f400 0x005cac00 0x00200000 0x0038a400 0x000de000 0x005a0400 0x004b0800
LBA[40]: 0x00376400 0x00488800 0x00570000 0x00004c00 0x001cc400 0x006f8400 0x00360c00 0x00024c00 0x002fb800 0x00778400
LBA[50]: 0x006d8000 0x0044e400 0x00641000 0x00268800 0x0070b000 0x0003c000 0x004f6400 0x006fdc00 0x00791800 0x005f6400
LBA[60]: 0x00217400 0x0007d400 0x00661000 0x00180000 0x00656400 0x00780800 0x000c6000 0x00094400 0x006a0000 0x001c5800
LBA[70]: 0x00538400 0x00228c00 0x00731400 0x00096c00 0x001f8c00 0x006c3400 0x00252c00 0x00046400 0x006c3c00 0x0036b000
LBA[80]: 0x004c2000 0x00141000 0x00087c00 0x0017f800 0x00554400 0x007b5400 0x007e7000 0x00517c00 0x002af800 0x006fa000
LBA[90]: 0x007e0000 0x003d6000 0x006cf000 0x0043b800 0x001ef000 0x00006800 0x0076e000 0x0014a800 0x0017a800 0x00451800
                第191个IO   第192个IO   第193个IO   第194个IO   第195个IO   第196个IO   第197个IO   第198个IO   第199个IO   第200个IO
PRE LBA LIST:
LBA[ 0]: 0x0070a800 0x005dc800 0x0008b400 0x004d9000 0x006f9c00 0x00720000 0x0073f000 0x00284000 0x007a4c00 0x004d2800
                第100个IO   第99个IO    第98个IO    第97个IO    第96个IO    第95个IO    第94个IO    第93个IO    第92个IO    第91个IO
```

dump第101个IO的数据

LBA工具实现原理：数据校验

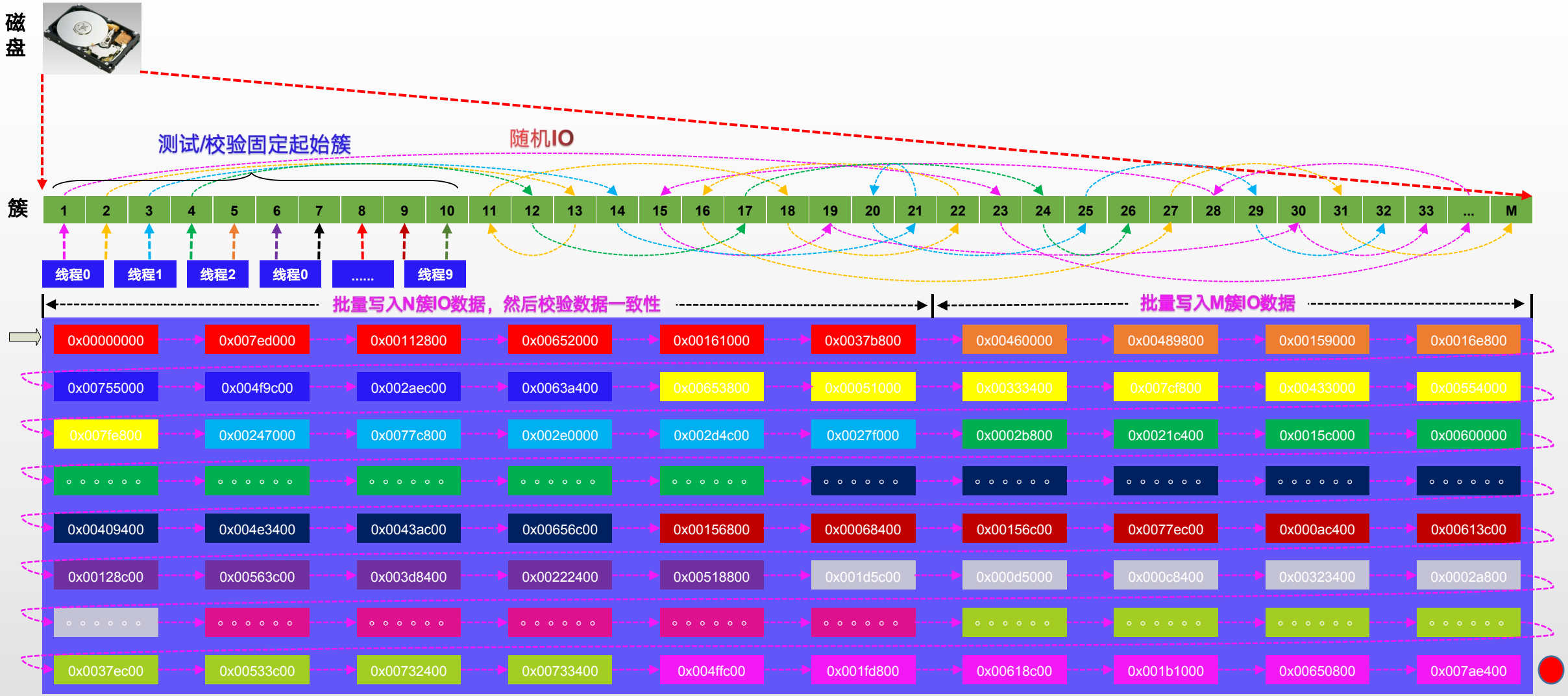
- 全盘数据校验 （**checktime & runtime**：线程数据校验、簇间数据校验、簇内数据校验）



1. runtime的全盘数据校验，即：10个线程向磁盘写入IO数据簇，当磁盘中的数据写满后，10个线程随即对全盘的所有写入数据进行一次完整校验(簇间数据校验、簇内数据校验)；
2. checktime的全盘数据校验，三个重要命令：**hd_write_verify -c -D -T 1 disk/file**　　**hd_write_verify -c -D -T 10 disk/file**　　**hd_write_verify_dump -c -D -T 0-9 disk/file**
3. 每次对磁盘写入数据测试完一轮后，可以进行三种数据校验：全盘数据校验、批量数据校验、随机数据校验，即：强双向链表、数组、队列混合体数据出现脱钩断链、鸡蛋的裂缝，都会被检测出来。如前面列举的案例：数据错位一个字节，或者出错两个字节，LBA工具都能校验发现。

LBA工具实现原理：数据校验

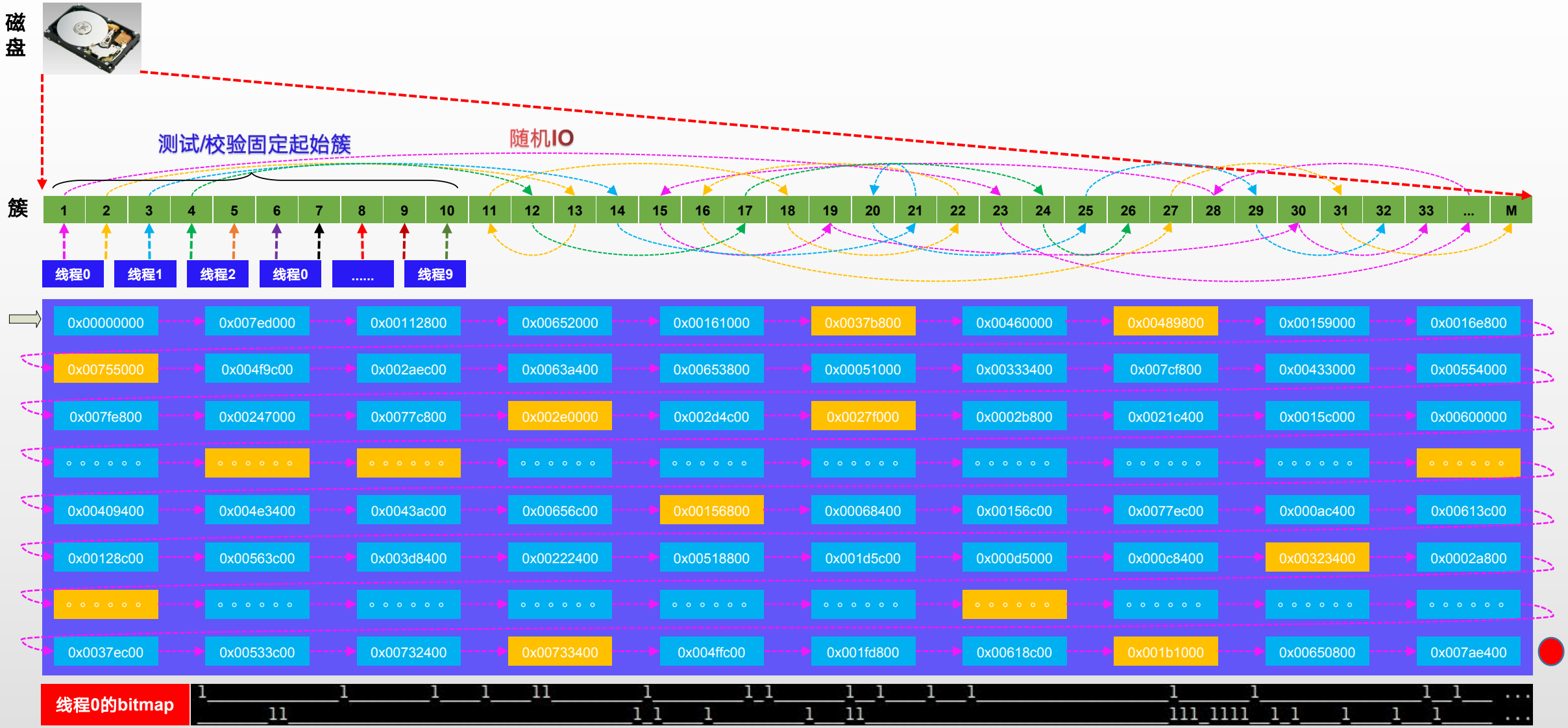
- 批量数据校验（**runtime**：线程数据校验、簇间数据校验、簇内数据校验）



1. 批量数据校验属于runtime的数据校验，即：一边写入测试数据，一边校验前面写入数据的一致性；
2. 10个线程，每个线程分批向磁盘写入数据，每批写入磁盘的IO数据簇个数是随机的：**i, j, k, m, n, l, x, y, z, ...**；
3. 在写下一批IO数据簇之前，先校验上一批数据的一致性：簇间数据校验和簇内数据校验，当一个线程的所有数据写完成后，即这个线程的线程数据校验也完成了

LBA工具实现原理：数据校验

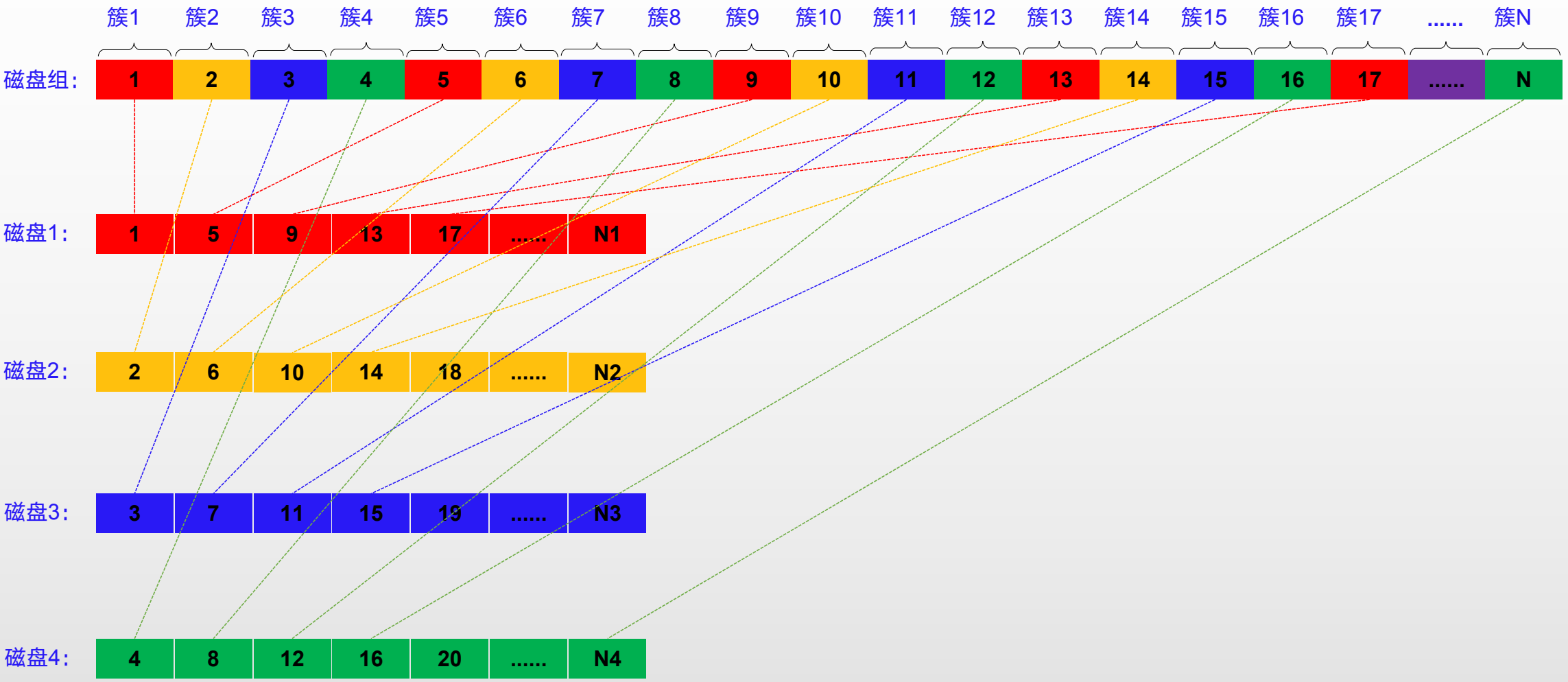
- 随机数据校验（**runtime**：簇间数据校验、簇内数据校验）



1. 每个线程写入磁盘中的IO数据簇，当每个IO数据簇被分配时，对应的线程bitmap被置位为1；
2. 每个线程定期随机查找被置位为1的线程bitmap，读取对应IO簇的数据进行校验；

LBA工具实现原理：磁盘组数据一致性(一致性备份、快照等)

- 条带策略：round-robin (高级功能) --- 最多支持同时测试64个磁盘/文件

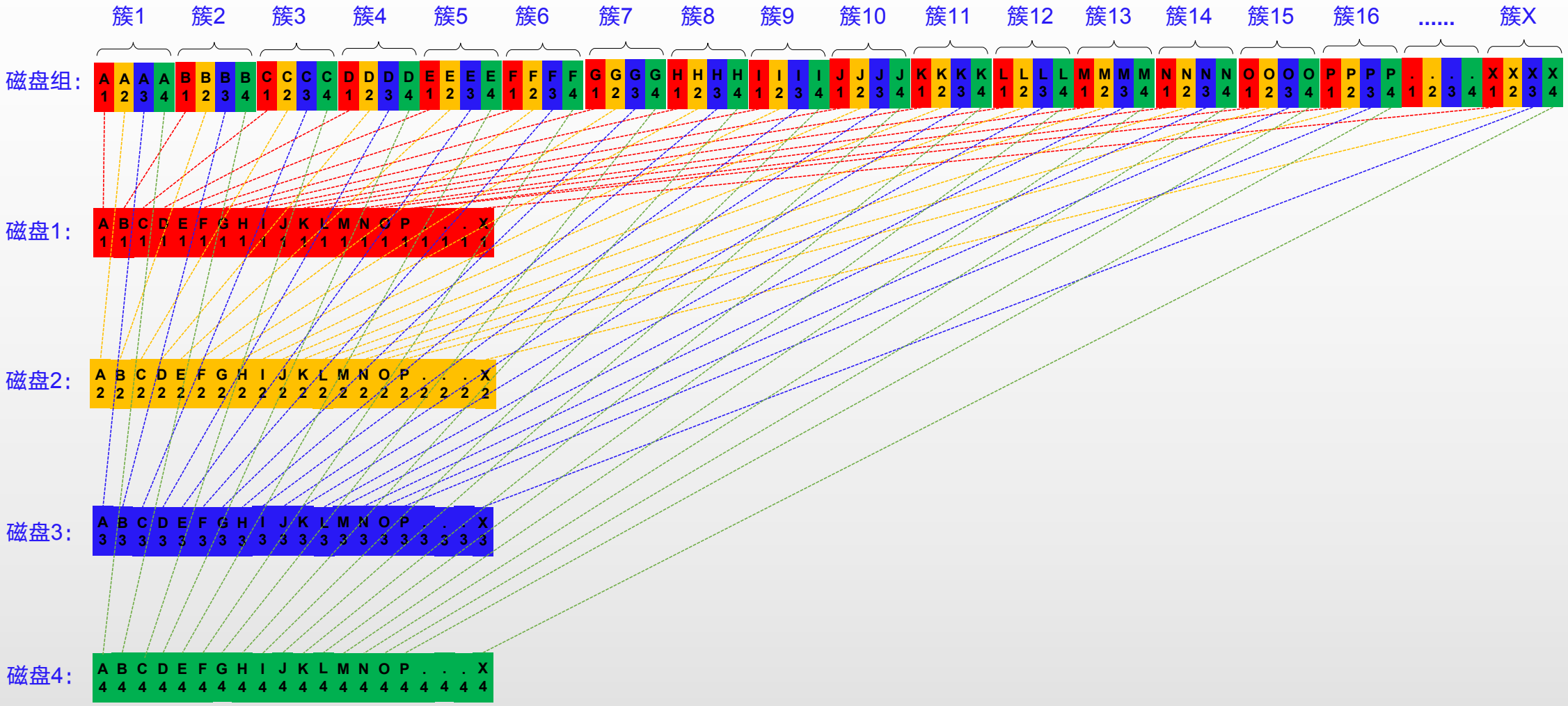


```
[root@localhost lba]# hd_write_verify -c -D -S 1024 -V all -T 10 -P robin --stripe lba1.raw --stripe lba2.raw --stripe lba3.raw --stripe lba4.raw
File Information:
        File:       lba1.raw
        Size:       1073741824
        Blocks:     0
        IO Block:   65536
        Device:     2051
        Inode:      1677992490
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:       lba2.raw
        Size:       1073741824
        Blocks:     0
        IO Block:   65536
        Device:     2051
        Inode:      1677992491
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:       lba3.raw
        Size:       1073741824
        Blocks:     0
        IO Block:   65536
        Device:     2051
        Inode:      1677992492
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:       lba4.raw
        Size:       1073741824
        Blocks:     0
        IO Block:   65536
        Device:     2051
        Inode:      1677992493
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File: stripe | Thread: 10 | Total Sectors: 33554432 | Total Clusters: 8192 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush | Verify: 5 | Notify: 0

Current Time: 2023-06-12 20:12:30
Thread 5 [tid: 1496736]: Starting check file ...
Thread 9 [tid: 1496740]: Starting check file ...
Thread 6 [tid: 1496737]: Starting check file ...
Thread 8 [tid: 1496739]: Starting check file ...
Thread 0 [tid: 1496731]: Starting check file ...
Thread 3 [tid: 1496734]: Starting check file ...
Thread 7 [tid: 1496738]: Starting check file ...
Thread 4 [tid: 1496735]: Starting check file ...
Thread 2 [tid: 1496733]: Starting check file ...
Thread 1 [tid: 1496732]: Starting check file ...

--------------------------------------------------------------------------------------------------------------

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-12 20:12:30
68, 101 | 89, 109 | 68, 100 | 92, 111 | 59, 106 | 62, 113 | 90, 117 | 89, 109 | 86, 107 | 84, 106 |
Pause

Start
122, 133 | 140, 150 | 125, 137 | 137, 147 | 122, 140 | 133, 141 | 125, 149 | 119, 144 | 139, 143 | 129, 136 |
Quit

Current Time: 2023-06-12 20:13:02
[root@localhost lba]#
```

```
hd_write_verify_dump -c -D -T 0 -I lba1.raw -I lba2.raw -I lba3.raw -I lba4.raw
--------------------------------------------------------------------------------

File Information:
        File:       lba1.raw
        Size:       1073741824
        Blocks:     1994752
        IO Block:   4096
        Device:     2064
        Inode:      163831
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:       lba2.raw
        Size:       1073741824
        Blocks:     1734656
        IO Block:   4096
        Device:     2064
        Inode:      166369
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:       lba3.raw
        Size:       1073741824
        Blocks:     1763328
        IO Block:   4096
        Device:     2064
        Inode:      167951
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:       lba4.raw
        Size:       1073741824
        Blocks:     1392640
        IO Block:   4096
        Device:     2064
        Inode:      172036
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

        Sector_Per_Cluster: 1024

Current Time: 2024-03-21 10:39:52

================================================================================================================

BUG 007[1] LAST IO?: [Thread: 0] VERIFY SECTOR[755] DIFFER: 1 | read[419] | filename: lba4.raw

hd_write_verify_dump -c -D -L 0x002ba400 disk/file
hd_write_verify_dump -c -D -L 0x00379400 disk/file

CORRECT LBA[0]: 0x731c00 | HF: 0x04001270 F1: 0x641f6aec F2: 0x6f4ee3c0 [LOOP: 1 WRITE: 419 ROBIN-STRIPE: 3 LBA: 0x1cc400 SECTOR: 0 / NUM: 1024 / ZERO: 0]
CORRECT IO Time: 2022-12-20 16:41:24

ERROR LBA[755]: 0x731ef3 | HF: 0x04001270 F1: 0x641f6aec F2: 0x6f4ee3c0 [LOOP: 1 WRITE: 419 ROBIN-STRIPE: 3 LBA: 0x1cc6f3 SECTOR: 755 / NUM: 1024 / ZERO: 0]
INCONSISTENT IO Time: 2022-12-20 16:41:24

CORRECT LBA[756]: 0x731ef4 | HF: 0x04001270 F1: 0x641f6aec F2: 0x6f4ee3c0 [LOOP: 1 WRITE: 419 ROBIN-STRIPE: 3 LBA: 0x1cc6f4 SECTOR: 756 / NUM: 1024 / ZERO: 0]
CORRECT IO Time: 2022-12-20 16:41:24

--------------------------------------------------------------------------------

LBA: 0x731ef2 | HF: 0x04001270 F1: 0x641f6aec F2: 0x6f4ee3c0 [LOOP: 1 WRITE: 419 ROBIN-STRIPE: 3 LBA: 0x1cc6f2 SECTOR: 754 / NUM: 1024 / ZERO: 0]
IO Time: 2022-12-20 16:41:24

Buffer addr: 0x7fdc09788400 | Physical addr: 0x5df745400

LBA LIST:
LBA[ 0]: 0x00731c00 0x00379400 0x000ee800 0x005a3400 0x00595000 0x005cfc00 0x003ab400 0x00405c00 0x0075d800 0x000e7c00
LBA[10]: 0x0031f400 0x00514800 0x0026e800 0x00083400 0x0015b400 0x003fac00 0x004d7400 0x00137400 0x0044cc00 0x005a5c00
LBA[20]: 0x004fa400 0x0019d400 0x00295800 0x0057dc00 0x00524800 0x0013a800 0x00195000 0x005c8800 0x0012cc00 0x003ac800
LBA[30]: 0x00194400 0x002cec00 0x00131800 0x0000f800 0x0076b000 0x003ed400 0x0050d400 0x00493400 0x0004a800 0x00510c00
LBA[40]: 0x003abc00 0x00064c00 0x00214400 0x0002ac00 0x003e2c00 0x00181400 0x0003d000 0x0011a400 0x004ad800 0x00717c00
LBA[50]: 0x002e3800 0x00731400 0x0063f400 0x005ff400 0x000c4800 0x00129800 0x004de000 0x00587400 0x00272400 0x005ff800
LBA[60]: 0x002e5400 0x0039c800 0x002b1800 0x00129c00 0x0065a000 0x0053a400 0x006eb000 0x004d9400 0x00767000 0x00728c00
LBA[70]: 0x000b4c00 0x0008b800 0x006ae800 0x0075dc00 0x00435800 0x002c5800 0x0066fc00 0x00048800 0x00419400 0x007d0400
LBA[80]: 0x007c4c00 0x001a8800 0x004e2000 0x0037fc00 0x0072f800 0x0015bc00 0x00523800 0x00058c00 0x00211c00 0x00323000
LBA[90]: 0x0067c800 0x002ffc00 0x0061d400 0x00597800 0x004d6800 0x0061dc00 0x00455800 0x001d6400 0x00433c00 0x0023dc00

PRE LBA LIST:
LBA[ 0]: 0x002ba400 0x00124800 0x000ed000 0x000ca800 0x00616000 0x0055e800 0x00302800 0x00388c00 0x004c6c00 0x00305000

--------------------------------------------------------------------------------

LBA: 0x731ef3 | HF: 0x04001270 F1: 0x641f6aec F2: 0x6f4ee3c0 [LOOP: 1 WRITE: 419 ROBIN-STRIPE: 3 LBA: 0x1cc6f3 SECTOR: 755 / NUM: 1024 / ZERO: 0]
IO Time: 2022-12-20 16:41:24

Buffer addr: 0x7fdc09788600 | Physical addr: 0x5df745600

LBA LIST:
LBA[ 0]: 0x00731c00 0x00379400 0x000ee800 0x005a3400 0x00595000 0x005cfc00 0x003ab400 0x00405c00 0x0075d800 0x000e7c00
LBA[10]: 0x0031f400 0x00514800 0x0026e800 0x00083400 0x0015b400 0x003fac00 0x004d7400 0x00137400 0x0044cc00 0x005a5c00
LBA[20]: 0x004fa400 0x0019d400 0x00295800 0x0057dc00 0x00524800 0x0013a800 0x00195000 0x005c8800 0x0012cc00 0x003ac800
LBA[30]: 0x00194400 0x002cec00 0x00131800 0x0000f800 0x0076b000 0x003ed400 0x0050d400 0x00493400 0x0004a800 0x00510c00
LBA[40]: 0x003abc00 0x00064c00 0x00214400 0x0002ac00 0x003e2c00 0x00181400 0x0003d000 0x0011a400 0x004ad800 0x00717c00
LBA[50]: 0x002e3800 0x00731400 0x0063f400 0x005ff400 0x000c4800 0x00129800 0x004de000 0x00587400 0x00272400 0x005ff800
LBA[60]: 0x002e5400 0x0039c800 0x002b1800 0x00129c00 0x0065a000 0x0053a400 0x006eb000 0x004d9400 0x00767000 0x00728c00
LBA[70]: 0x000b4c00 0x0008b800 0x006ae800 0x0075dc00 0x00435800 0x002c5800 0x0066fc00 0x00048800 0x00419400 0x007d0400
LBA[80]: 0x007c4c00 0x001a8800 0x004e2000 0x0037fc00 0x0072f800 0x0015bc00 0x00523800 0x00058c00 0x00211c00 0x00323000
LBA[90]: 0x0067c800 0x002ffc00 0x0061d400 0x00597800 0x004d6800 0x0061dc00 0x00455800 0x001d6400 0x00433c00 0x0023dc00

PRE LBA LIST:
LBA[ 0]: 0x002ba400 0x00124800 0x000ed000 0x000ca800 0x00616000 0x0055e800 0x00302800 0x00388c00 0x004c6c00 0x00305000

--------------------------------------------------------------------------------

LBA: 0x731ef4 | HF: 0x04001270 F1: 0x641f6aec F2: 0x6f4ee3c0 [LOOP: 1 WRITE: 419 ROBIN-STRIPE: 3 LBA: 0x1cc6f4 SECTOR: 756 / NUM: 1024 / ZERO: 0]
IO Time: 2022-12-20 16:41:24

Buffer addr: 0x7fdc09788800 | Physical addr: 0x5df745800

LBA LIST:
LBA[ 0]: 0x00731c00 0x00379400 0x000ee800 0x005a3400 0x00595000 0x005cfc00 0x003ab400 0x00405c00 0x0075d800 0x000e7c00
LBA[10]: 0x0031f400 0x00514800 0x0026e800 0x00083400 0x0015b400 0x003fac00 0x004d7400 0x00137400 0x0044cc00 0x005a5c00
LBA[20]: 0x004fa400 0x0019d400 0x00295800 0x0057dc00 0x00524800 0x0013a800 0x00195000 0x005c8800 0x0012cc00 0x003ac800
LBA[30]: 0x00194400 0x002cec00 0x00131800 0x0000f800 0x0076b000 0x003ed400 0x0050d400 0x00493400 0x0004a800 0x00510c00
LBA[40]: 0x003abc00 0x00064c00 0x00214400 0x0002ac00 0x003e2c00 0x00181400 0x0003d000 0x0011a400 0x004ad800 0x00717c00
LBA[50]: 0x002e3800 0x00731400 0x0063f400 0x005ff400 0x000c4800 0x00129800 0x004de000 0x00587400 0x00272400 0x005ff800
LBA[60]: 0x002e5400 0x0039c800 0x002b1800 0x00129c00 0x0065a000 0x0053a400 0x006eb000 0x004d9400 0x00767000 0x00728c00
LBA[70]: 0x000b4c00 0x0008b800 0x006ae800 0x0075dc00 0x00435800 0x002c5800 0x0066fc00 0x00048800 0x00419400 0x007d0400
LBA[80]: 0x007c4c00 0x001a8800 0x004e2000 0x0037fc00 0x0072f800 0x0015bc00 0x00523800 0x00058c00 0x00211c00 0x00323000
LBA[90]: 0x0067c800 0x002ffc00 0x0061d400 0x00597800 0x004d6800 0x0061dc00 0x00455800 0x001d6400 0x00433c00 0x0023dc00

PRE LBA LIST:
LBA[ 0]: 0x002ba400 0x00124800 0x000ed000 0x000ca800 0x00616000 0x0055e800 0x00302800 0x00388c00 0x004c6c00 0x00305000

================================================================================================================

Current Time: 2024-03-21 10:39:55
```

LBA工具实现原理：磁盘组数据一致性(一致性备份、快照等)

- 条带策略：cluster-split (高级功能) --- 最多支持同时测试64个磁盘/文件



```
[root@localhost lba]# hd_write_verify -c -D -S 2048 -V all -T 10 -P split --stripe lba1.raw --stripe lba2.raw --stripe lba3.raw --stripe lba4.raw
File Information:
        File:      lba1.raw
        Size:      1073741824
        Blocks:    0
        IO Block:  65536
        Device:    2051
        Inode:     1678094425
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:      lba2.raw
        Size:      1073741824
        Blocks:    0
        IO Block:  65536
        Device:    2051
        Inode:     1678094426
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:      lba3.raw
        Size:      1073741824
        Blocks:    0
        IO Block:  65536
        Device:    2051
        Inode:     1678094427
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:      lba4.raw
        Size:      1073741824
        Blocks:    0
        IO Block:  65536
        Device:    2051
        Inode:     1678094428
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File: stripe | Thread: 10 | Total Sectors: 33554432 | Total Clusters: 4096 | Sectors of Cluster: 2048 | DIRECT IO & NO Flush | Verify: 5 | Notify: 0

Current Time: 2023-06-13 16:54:56
Thread 1 [tid: 1579593]: Starting check file ...
Thread 8 [tid: 1579600]: Starting check file ...
Thread 6 [tid: 1579598]: Starting check file ...
Thread 2 [tid: 1579594]: Starting check file ...
Thread 5 [tid: 1579597]: Starting check file ...
Thread 7 [tid: 1579599]: Starting check file ...
Thread 4 [tid: 1579596]: Starting check file ...
Thread 9 [tid: 1579601]: Starting check file ...
Thread 3 [tid: 1579595]: Starting check file ...
Thread 0 [tid: 1579592]: Starting check file ...

-------------------------------------------------------------------------------------------------------------

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-13 16:54:56
467, 412 | 443, 412 | 441, 415 | 434, 393 | 442, 402 | 442, 410 | 456, 409 | 488, 426 | 440, 407 | 441, 410 |

Current Time: 2023-06-13 16:56:42
Thread 4 [tid: 1579596]: Starting check file ...
Thread 7 [tid: 1579599]: Starting check file ...
Thread 1 [tid: 1579593]: Starting check file ...
Thread 3 [tid: 1579595]: Starting check file ...
Thread 2 [tid: 1579594]: Starting check file ...
Thread 8 [tid: 1579600]: Starting check file ...
Thread 0 [tid: 1579592]: Starting check file ...
Thread 9 [tid: 1579601]: Starting check file ...
Thread 5 [tid: 1579597]: Starting check file ...
Thread 6 [tid: 1579598]: Starting check file ...
```

```
hd_write_verify_dump -c -D -T 2 -P split -I lba1.raw -I lba2.raw -I lba3.raw -I lba4.raw
-------------------------------------------------------------------------------------------------------------

File Information:
        File:      lba1.raw
        Size:      1073741824
        Blocks:    2058240
        IO Block:  4096
        Device:    2064
        Inode:     166369
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:      lba2.raw
        Size:      1073741824
        Blocks:    2058240
        IO Block:  4096
        Device:    2064
        Inode:     167940
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:      lba3.raw
        Size:      1073741824
        Blocks:    2058240
        IO Block:  4096
        Device:    2064
        Inode:     172036
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

File Information:
        File:      lba4.raw
        Size:      1073741824
        Blocks:    2058240
        IO Block:  4096
        Device:    2064
        Inode:     179158
        Links:     1
        File Size: 1073741824 / 1024M / 1.00G

        Sector_Per_Cluster: 1024

Current Time: 2024-03-21 18:08:55

=============================================================================================================

BUG 007[1] LAST IO?: [Thread: 2] VERIFY SECTOR[709] DIFFER: 1 | read[412] | Filename: lba3.raw

hd_write_verify_dump -c -D -L 0x00021400 disk/file
hd_write_verify_dump -c -D -L 0x003ce800 disk/file

CORRECT LBA[0]: 0x198800 | HF: 0x04001272 F1: 0xe2c46383 F2: 0x1da174c6 [LOOP: 1 WRITE: 412 SPLIT-STRIPE: 0 LBA: 0x66200 SECTOR: 0 / NUM: 1024 / ZERO: 12]
CORRECT IO Time: 2024-03-21 15:49:02

ERROR LBA[709]: 0x198ac5 | HF: 0x04001272 F1: 0xe2c46383 F2: 0x1da174c6 [LOOP: 1 WRITE: 412 SPLIT-STRIPE: 2 LBA: 0x662c5 SECTOR: 709 / NUM: 1024 / ZERO: 12]
INCONSISTENT IO Time: 2024-03-21 15:49:02

CORRECT LBA[710]: 0x198ac6 | HF: 0x04001272 F1: 0xe2c46383 F2: 0x1da174c6 [LOOP: 1 WRITE: 412 SPLIT-STRIPE: 2 LBA: 0x662c6 SECTOR: 710 / NUM: 1024 / ZERO: 12]
CORRECT IO Time: 2024-03-21 15:49:02

-------------------------------------------------------------------------------------------------------------

LBA: 0x198ac4 | HF: 0x04001272 F1: 0xe2c46383 F2: 0x1da174c6 [LOOP: 1 WRITE: 412 SPLIT-STRIPE: 2 LBA: 0x662c4 SECTOR: 708 / NUM: 1024 / ZERO: 12]
IO Time: 2024-03-21 15:49:02

Buffer addr: 0x7fe6a2bf2800 | Physical addr: 0x286cdc800

LBA LIST:
LBA[ 0]: 0x00198800 0x003ce800 0x00556c00 0x00660c00 0x007a5800 0x004e1000 0x0024b000 0x00587800 0x0011a800 0x0024a000
LBA[10]: 0x0018a000 0x00312000 0x00248000 0x00203800 0x006d5000 0x004c7000 0x00792400 0x0070c000 0x0038cc00 0x001c7000
LBA[20]: 0x00284000 0x00198c00 0x0046c800 0x0020e400 0x00044c00 0x001a0400 0x004ca800 0x0016ac00 0x00101c00 0x007a1400
LBA[30]: 0x0044e800 0x003a1400 0x0018f000 0x00235c00 0x004e0000 0x00023c00 0x005c6400 0x00666000 0x000a6400 0x0009a400
LBA[40]: 0x00397400 0x0058b800 0x003e5000 0x00790800 0x00273400 0x004f7c00 0x0061b400 0x00047c00 0x001b6c00 0x00543800
LBA[50]: 0x002fe400 0x000d7800 0x007bfc00 0x002ccc00 0x004f8400 0x001be400 0x0077dc00 0x000ed400 0x00507800 0x00595400
LBA[60]: 0x00412c00 0x000cfc00 0x0055d400 0x00284400 0x007a3400 0x00166800 0x005a4800 0x0039a800 0x0031cc00 0x0035dc00
LBA[70]: 0x00079f400 0x004fd000 0x006b1c00 0x0059d800 0x00150400 0x00785400 0x003e8c00 0x00478400 0x000c8000 0x00346800
LBA[80]: 0x0071ac00 0x00569000 0x00706400 0x00749800 0x003a1c00 0x00053800 0x001d4800 0x005a9000 0x0064b800 0x006e9400
LBA[90]: 0x00532000 0x001b2000 0x00123c00 0x0012e400 0x007eb000 0x004a4400 0x00326000 0x0072f000 0x0064bc00 0x00749c00

PRE LBA LIST:
LBA[ 0]: 0x00021400 0x0043dc00 0x00163000 0x002d9400 0x00393400 0x00168400 0x002f4400 0x0017dc00 0x005ddc00 0x004d0400

-------------------------------------------------------------------------------------------------------------

LBA: 0x198ac5 | HF: 0x04001272 F1: 0xe2c46383 F2: 0x1da174c6 [LOOP: 1 WRITE: 412 SPLIT-STRIPE: 2 LBA: 0x662c5 SECTOR: 709 / NUM: 1024 / ZERO: 12]
IO Time: 2024-03-21 15:49:02

Buffer addr: 0x7fe6a2bf2a00 | Physical addr: 0x286cdca00

LBA LIST:
LBA[ 0]: 0x00198800 0x003ce800 0x00556c00 0x00660c00 0x007a5800 0x004e1000 0x0024b000 0x00587800 0x29429928 0x0024a000
LBA[10]: 0x0018a000 0x00312000 0x00248000 0x00203800 0x006d5000 0x004c7000 0x00792400 0x0070c000 0x0038cc00 0x001c7000
LBA[20]: 0x00284000 0x00198c00 0x0046c800 0x0020e400 0x00044c00 0x001a0400 0x004ca800 0x0016ac00 0x00101c00 0x007a1400
LBA[30]: 0x0044e800 0x003a1400 0x0018f000 0x00235c00 0x004e0000 0x00023c00 0x005c6400 0x00666000 0x000a6400 0x0009a400
LBA[40]: 0x00397400 0x0058b800 0x003e5000 0x00790800 0x00273400 0x004f7c00 0x0061b400 0x00047c00 0x001b6c00 0x00543800
LBA[50]: 0x002fe400 0x000d7800 0x007bfc00 0x002ccc00 0x004f8400 0x001be400 0x0077dc00 0x000ed400 0x00507800 0x00595400
LBA[60]: 0x00412c00 0x000cfc00 0x0055d400 0x00284400 0x007a3400 0x00166800 0x005a4800 0x0039a800 0x0031cc00 0x0035dc00
LBA[70]: 0x00079f400 0x004fd000 0x006b1c00 0x0059d800 0x00150400 0x00785400 0x003e8c00 0x00478400 0x000c8000 0x00346800
LBA[80]: 0x0071ac00 0x00569000 0x00706400 0x00749800 0x003a1c00 0x00053800 0x001d4800 0x005a9000 0x0064b800 0x006e9400
LBA[90]: 0x00532000 0x001b2000 0x00123c00 0x0012e400 0x007eb000 0x004a4400 0x00326000 0x0072f000 0x0064bc00 0x00749c00

PRE LBA LIST:
LBA[ 0]: 0x00021400 0x0043dc00 0x00163000 0x002d9400 0x00393400 0x00168400 0x002f4400 0x0017dc00 0x005ddc00 0x004d0400

-------------------------------------------------------------------------------------------------------------

LBA: 0x198ac6 | HF: 0x04001272 F1: 0xe2c46383 F2: 0x1da174c6 [LOOP: 1 WRITE: 412 SPLIT-STRIPE: 2 LBA: 0x662c6 SECTOR: 710 / NUM: 1024 / ZERO: 12]
IO Time: 2024-03-21 15:49:02

Buffer addr: 0x7fe6a2bf2c00 | Physical addr: 0x286cdcc00

LBA LIST:
LBA[ 0]: 0x00198800 0x003ce800 0x00556c00 0x00660c00 0x007a5800 0x004e1000 0x0024b000 0x00587800 0x0011a800 0x0024a000
LBA[10]: 0x0018a000 0x00312000 0x00248000 0x00203800 0x006d5000 0x004c7000 0x00792400 0x0070c000 0x0038cc00 0x001c7000
LBA[20]: 0x00284000 0x00198c00 0x0046c800 0x0020e400 0x00044c00 0x001a0400 0x004ca800 0x0016ac00 0x00101c00 0x007a1400
LBA[30]: 0x0044e800 0x003a1400 0x0018f000 0x00235c00 0x004e0000 0x00023c00 0x005c6400 0x00666000 0x000a6400 0x0009a400
LBA[40]: 0x00397400 0x0058b800 0x003e5000 0x00790800 0x00273400 0x004f7c00 0x0061b400 0x00047c00 0x001b6c00 0x00543800
LBA[50]: 0x002fe400 0x000d7800 0x007bfc00 0x002ccc00 0x004f8400 0x001be400 0x0077dc00 0x000ed400 0x00507800 0x00595400
LBA[60]: 0x00412c00 0x000cfc00 0x0055d400 0x00284400 0x007a3400 0x00166800 0x005a4800 0x0039a800 0x0031cc00 0x0035dc00
LBA[70]: 0x00079f400 0x004fd000 0x006b1c00 0x0059d800 0x00150400 0x00785400 0x003e8c00 0x00478400 0x000c8000 0x00346800
LBA[80]: 0x0071ac00 0x00569000 0x00706400 0x00749800 0x003a1c00 0x00053800 0x001d4800 0x005a9000 0x0064b800 0x006e9400
LBA[90]: 0x00532000 0x001b2000 0x00123c00 0x0012e400 0x007eb000 0x004a4400 0x00326000 0x0072f000 0x0064bc00 0x00749c00

PRE LBA LIST:
LBA[ 0]: 0x00021400 0x0043dc00 0x00163000 0x002d9400 0x00393400 0x00168400 0x002f4400 0x0017dc00 0x005ddc00 0x004d0400

=============================================================================================================

Current Time: 2024-03-21 18:09:05
```

LBA工具实现原理：fio和vdbench的verify功能实现原理

- **fio数据校验原理** (https://blog.csdn.net/weixin_52944590/article/details/137870811)

数据校验的工作流程：用-verify=str来选择校验算法：md5/crc16/crc32/crc32c/crc32c-intel/crc64/crc7/sha256/sha512/sha1等；开启校验功能，需要配置do_verify参数：如果是写，do_verify=1表示写完再读校验，但会占用大量内存，因为fio会把每个数据块的校验数据保存在内存里；而do_verify=0时只写校验数据，不做读校验。

另外：verify=meta时，fio会在数据块内写入时间戳、逻辑地址等，同时可以通过verify_pattern参数指定写入数据pattern。

```
struct verify_header {
        uint16_t magic;
        uint16_t verify_type;
        uint32_t len;
        uint64_t rand_seed;
        uint64_t offset;
        uint32_t time_sec;
        uint32_t time_nsec;
        uint16_t thread;
        uint16_t numberio;
        uint32_t crc32;
};
```

- **vdbench数据校验原理** (https://blog.csdn.net/zangjiaoshou/article/details/122063819)

数据校验的工作流程：每一个在存储系统中的第一次写操作记录在一个表中，假定写操作的块大小是1M，那么这个块大小中的每512字节中包含的两项---八个字节的逻辑字节地址(LBA)和一个字节的数据校验key值(标记是第几次写，范围为0-125，00代表创建写，01代表第一次覆盖写，以此类推，当到达126后折返00，重新来一轮)会被记录，这个过程为生成校验日志；第二次重新运行脚本(使用参数-jr或者-v)则根据第一次记录的日志进行数据校验。

./vdbench -f parmfile -jn

-jn：支持数据校验，可以验证数据一致性，并支持crash和reboot时的数据校验。

LBA工具实现原理

- 簇IO对齐、簇IO随机拆分：测试存储栈IO合并(高级功能)

- LBA bitmap：区分runtime与checktime (高级功能)

```
BUG 002[4]: [Thread: 7] VERIFY ZERO SECTOR[1020] DIFFER: 1 | CORRECT LBA[0]: 0xb50800
LBA: 0xb50bfc | thread_num: 10, bitmap: 0 0 0 0 0 0 0 1 0 0

LAST CORRECT WRITE:
LBA: 0x484000 | HF: 0x04001277 F1: 0xa1b5ec90 F2: 0x1a8453bd [LOOP: 1 WRITE: 86 SECTOR:
LAST WRITE IO Time: 2023-06-12 17:12:01
```

```
Current Time: 2023-06-13 10:03:57
BUG 004: [Thread: 5] DATA LOST | read[2]: 65536 | filename: /dev/vdc
PREV LBA: 0x4a5100 | LOST LBA: 0x35480 | NEXT LBA: 0x1c93780
LBA: 0x35480 | thread_num: 10, bitmap: 0 0 0 0 0 0 0 0 0 0
```

- 数据回写(负载保持)，参数：--writeback (高级功能)

  应用于虚拟机热迁移场景

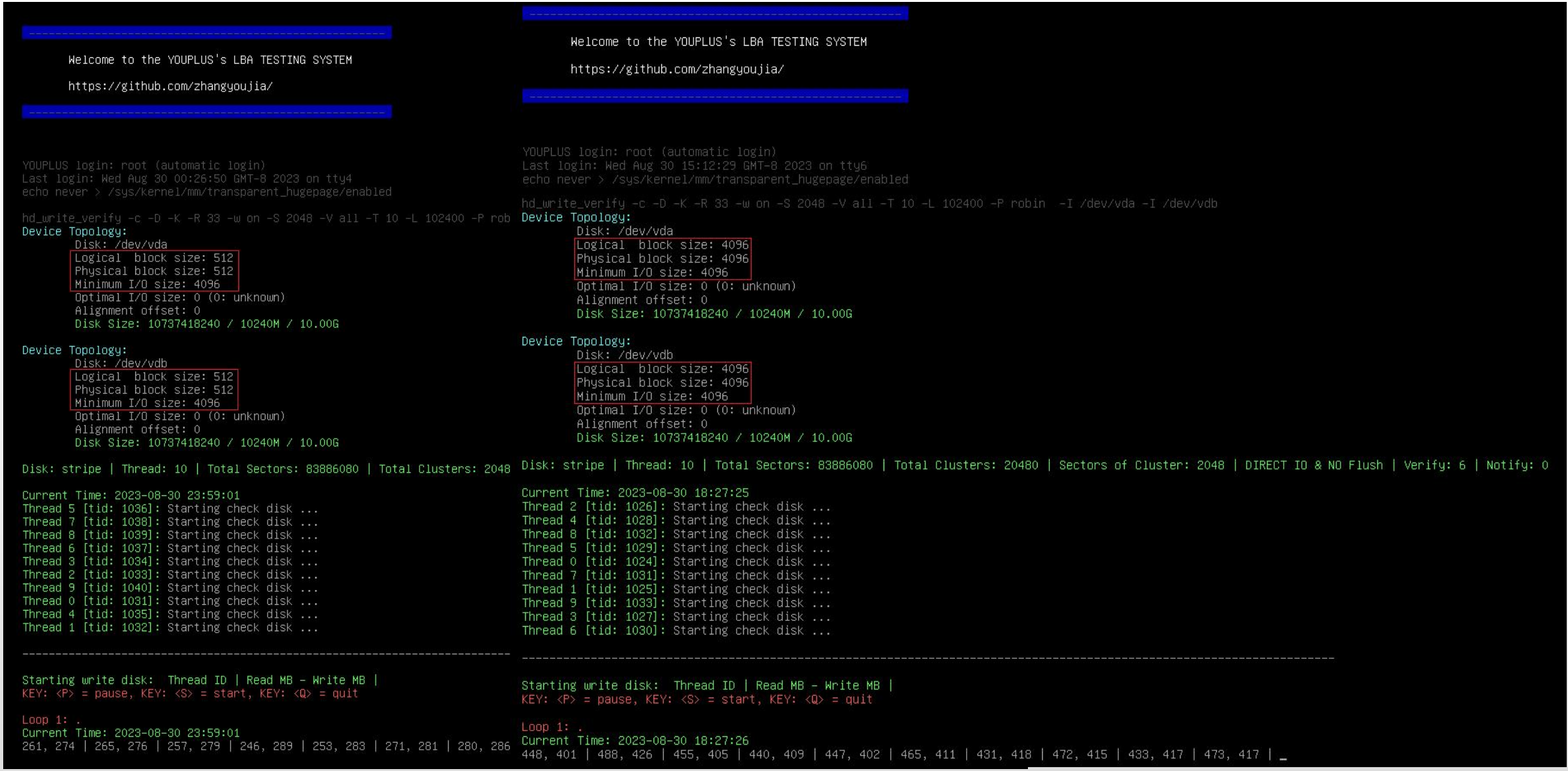
```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5.30    0.00   12.58   41.72    0.17   40.23

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz
vda               0.00     0.00   34.00   33.67 34816.00 22186.67  1684.81
vdb               0.00     0.00   25.33   31.67 25941.33 19235.67  1585.16
sr0               0.00     0.00    0.33    0.00     0.67     0.00     4.00
loop1             0.00     0.00  200.00  332.00 102400.00 102400.00   769.9

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           3.20    0.00   13.13   36.70    0.34   46.63

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz
vda               0.00     0.00   40.33   20.00 41301.33 11946.67  1765.13
vdb               0.00     0.00   33.33   25.33 34133.33 15018.67  1675.64
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00
loop1             0.00     0.00  200.00  319.33 102400.00 102400.00   788.7
```

```
nd_write_verify -c -D -K -R 33 -w on -S 1024 -V once -
Device Topology:
        Disk: /dev/loop1
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 3387949056 / 3231M / 3.16G

Disk: /dev/loop1 | Thread: 10 | Total Sectors: 6617088

Current Time: 2023-07-10 14:06:05
Thread 5 [tid: 1122]: Starting check disk ...
Thread 6 [tid: 1123]: Starting check disk ...
Thread 1 [tid: 1118]: Starting check disk ...
Thread 9 [tid: 1126]: Starting check disk ...
Thread 4 [tid: 1121]: Starting check disk ...
Thread 2 [tid: 1119]: Starting check disk ...
Thread 3 [tid: 1120]: Starting check disk ...
Thread 8 [tid: 1125]: Starting check disk ...
Thread 0 [tid: 1117]: Starting check disk ...
Thread 7 [tid: 1124]: Starting check disk ...
```

```
399: void virtio_blk_submit_multireq(BlockBackend *blk, MultiReqBuffer *mrb)
400: {
401:     int i = 0, start = 0, num_reqs = 0, niov = 0, nb_sectors = 0;
402:     int max_xfer_len = 0;
403:     int64_t sector_num = 0;
404:
405:     if (mrb->num_reqs == 1) {
406:         submit_requests(blk, mrb, 0, 1, -1);
407:         mrb->num_reqs = 0;
408:         return;
409:     }
410:
411:     max_xfer_len = blk_get_max_transfer_length(mrb->reqs[0]->dev->blk);
412:     max_xfer_len = MIN_NON_ZERO(max_xfer_len, BDRV_REQUEST_MAX_SECTORS);
413:
414:     qsort(mrb->reqs, mrb->num_reqs, sizeof(*mrb->reqs),
415:           &multireq_compare);
416:
417:     for (i = 0; i < mrb->num_reqs; i++) {
418:         VirtIOBlockReq *req = mrb->reqs[i];
419:         if (num_reqs > 0) {
420:             bool merge = true;
421:
422:             /* merge would exceed maximum number of IOVs */
423:             if (niov + req->qiov.niov > IOV_MAX) {
424:                 merge = false;
425:             }
426:
427:             /* merge would exceed maximum transfer length of backend device */
428:             if (req->qiov.size / BDRV_SECTOR_SIZE + nb_sectors > max_xfer_len) {
429:                 merge = false;
430:             }
431:
432:             /* requests are not sequential */
433:             if (sector_num + nb_sectors != req->sector_num) {
434:                 merge = false;
435:             }
436:
437:             if (!merge) {
438:                 submit_requests(blk, mrb, start, num_reqs, niov);
439:                 num_reqs = 0;
440:             }
441:         }
442:
443:         if (num_reqs == 0) {
444:             sector_num = req->sector_num;
445:             nb_sectors = niov = 0;
446:             start = i;
447:         }
448:
449:         nb_sectors += req->qiov.size / BDRV_SECTOR_SIZE;
450:         niov += req->qiov.niov;
451:         num_reqs++;
452:     }
453:
454:     submit_requests(blk, mrb, start, num_reqs, niov);
455:     mrb->num_reqs = 0;
456: }
```

LBA工具实现原理

- 支持4K磁盘(簇IO随机拆分功能仍然生效)



LBA工具实现原理

- 全0数据测试，参数：--zero-ratio 和 --zero-sectors (高级功能)
  1. 虚拟机热迁移测试：全0数据内存页迁移
  2. 虚拟机热迁移测试：全0数据IO簇迁移
  3. qcow2镜像L2表项：CLUSTER_ZERO_PLAIN和CLUSTER_ZERO_ALLOC特性测试
  4. 稀疏文件的(远程)拷贝测试：cp / scp / rsync / socat等

```c
static int save_zero_page_to_file(RAMState *rs, QEMUFile *file,
                                  RAMBlock *block, ram_addr_t offset)
{
    uint8_t *p = block->host + offset;
    int len = 0;

    if (buffer_is_zero(p, TARGET_PAGE_SIZE)) {
        len += save_page_header(rs, file, block, offset | RAM_SAVE_FLAG_ZERO);
        qemu_put_byte(file, 0);
        len += 1;
        ram_release_page(block->idstr, offset);
    }
    return len;
}
```

```c
static void blk_send(QEMUFile *f, BlkMigBlock * blk)
{
    int len;
    uint64_t flags = BLK_MIG_FLAG_DEVICE_BLOCK;

    if (block_mig_state.zero_blocks &&
        buffer_is_zero(blk->buf, BLK_MIG_BLOCK_SIZE)) {
        flags |= BLK_MIG_FLAG_ZERO_BLOCK;
    }
```

```c
static bool
extent_copy (int src_fd, int dest_fd, cha
             size_t hole_size, off_t src_
             enum Sparse_type sparse_mode
             char const *src_name, char c
             bool *require_normal_copy)
{
  struct extent_scan scan;
  off_t last_ext_start = 0;
  off_t last_ext_len = 0;

  /* Keep track of the output position.
     We may need this at the end, for a f
  off_t dest_pos = 0;

  extent_scan_init (src_fd, &scan);

  *require_normal_copy = false;
  bool wrote_hole_at_eof = true;
  do
    {
      bool ok = extent_scan_read (&scan);
      if (! ok)
```

```
    l2 table[8186], data offset: 0x6c650000  [0x800000006c650000: COPIED] | vdisk offset: 0xdffa0000
    l2 table[8187], data offset: 0x6c660000  [0x800000006c660000: COPIED] | vdisk offset: 0xdffb0000
    l2 table[8188], data offset: 0x6c670000  [0x800000006c670001: ZERO] | vdisk offset: 0xdffc0000
    l2 table[8189], data offset: 0x6c680000  [0x800000006c680001: ZERO] | vdisk offset: 0xdffd0000
    l2 table[8190], data offset: 0x6c690000  [0x800000006c690001: ZERO] | vdisk offset: 0xdffe0000
    l2 table[8191], data offset: 0x6c6a0000  [0x800000006c6a0001: ZERO] | vdisk offset: 0xdfff0000

    L2 Entry: unaligned: 0, invalid: 0, used: 4132, new: 4132,
              zero_alloc: 21, zero_plain: 76, unused: 3963
    ------------------------------------------------------------

l1 table[   7], l2 offset: 0xdf0000  [0x8000000000df0000: COPIED]
    l2 table[   0], data offset: 0xf4b0000  [0x800000000f4b0000: COPIED] | vdisk offset: 0xe0000000
    l2 table[   1], data offset: 0xf4c0000  [0x800000000f4c0000: COPIED] | vdisk offset: 0xe0010000
    l2 table[   2], data offset: 0xf4d0000  [0x800000000f4d0000: COPIED] | vdisk offset: 0xe0020000
    l2 table[   3], data offset: 0xf4e0000  [0x800000000f4e0000: COPIED] | vdisk offset: 0xe0030000
    l2 table[   4], data offset: 0xf4f0000  [0x800000000f4f0000: COPIED] | vdisk offset: 0xe0040000
    l2 table[   5], data offset: 0xf500000  [0x800000000f500000: COPIED] | vdisk offset: 0xe0050000
    l2 table[   6], data offset: 0xf510000  [0x800000000f510000: COPIED] | vdisk offset: 0xe0060000
    l2 table[   7], data offset: 0xf520000  [0x800000000f520000: COPIED] | vdisk offset: 0xe0070000
    l2 table[   8], data offset: 0xf530000  [0x800000000f530000: COPIED] | vdisk offset: 0xe0080000
    l2 table[   9], data offset: 0xf540000  [0x800000000f540000: COPIED] | vdisk offset: 0xe0090000
    l2 table[  10], data offset: 0xf550000  [0x800000000f550000: COPIED] | vdisk offset: 0xe00a0000
    l2 table[  11], data offset: 0xf560000  [0x800000000f560000: COPIED] | vdisk offset: 0xe00b0000
    l2 table[  12], data offset: 0xf570000  [0x800000000f570000: COPIED] | vdisk offset: 0xe00c0000
    l2 table[  13], data offset: 0xf580000  [0x800000000f580000: COPIED] | vdisk offset: 0xe00d0000
    l2 table[  14], data offset: 0x0  [0x1: ZERO] | vdisk offset: 0xe00e0000
    l2 table[  15], data offset: 0x0  [0x1: ZERO] | vdisk offset: 0xe00f0000
    l2 table[  96], data offset: 0x2520000  [0x8000000002520000: COPIED] | vdisk offset: 0xe0600000
    l2 table[  97], data offset: 0x2530000  [0x8000000002530000: COPIED] | vdisk offset: 0xe0610000
```

- 块设备指令/文件系统特性测试，参数：--write-zeroes 和 --discard (高级功能)

指令：

```c
#define WRITE_SAME      0x41
#define UNMAP           0x42
```

```c
/* I/O commands */
enum nvme_opcode {
    nvme_cmd_flush          = 0x00,
    nvme_cmd_write          = 0x01,
    nvme_cmd_read           = 0x02,
    nvme_cmd_write_uncor    = 0x04,
    nvme_cmd_compare        = 0x05,
    nvme_cmd_write_zeroes   = 0x08,
    nvme_cmd_dsm            = 0x09,
    nvme_cmd_verify         = 0x0c,
    nvme_cmd_resv_register  = 0x0d,
    nvme_cmd_resv_report    = 0x0e,
    nvme_cmd_resv_acquire   = 0x11,
    nvme_cmd_resv_release   = 0x15,
    nvme_cmd_zone_mgmt_send = 0x79,
    nvme_cmd_zone_mgmt_recv = 0x7a,
    nvme_cmd_zone_append    = 0x7d,
};
```

```c
static int blkdev_common_ioctl(struct block_device *bdev, fmode_t mode,
                               unsigned cmd, unsigned long arg, void __user *argp)
{
    unsigned int max_sectors;

    switch (cmd) {
    case BLKFLSBUF:
        return blkdev_flushbuf(bdev, mode, cmd, arg);
    case BLKROSET:
        return blkdev_roset(bdev, mode, cmd, arg);
    case BLKDISCARD:
        return blk_ioctl_discard(bdev, mode, arg, 0);
    case BLKSECDISCARD:
        return blk_ioctl_discard(bdev, mode, arg,
                BLKDEV_DISCARD_SECURE);
    case BLKZEROOUT:
        return blk_ioctl_zeroout(bdev, mode, arg);
```

(1) mkfs.ext4 /dev/sdb
(2) mount -o discard /dev/sdb /mnt/test/
(3) fstrim -a

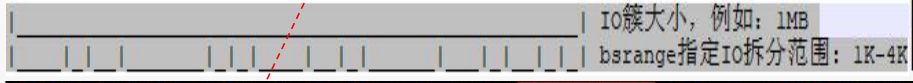
```
# cat /etc/fstab
UUID=3453g54-6628-2346-8123435f  /home  xfs  defaults,discard  0 0
```

```c
u16 nvmet_bdev_parse_io_cmd(struct nvmet_req *req)
{
    struct nvme_command *cmd = req->cmd;

    switch (cmd->common.opcode) {
    case nvme_cmd_read:
    case nvme_cmd_write:
        req->execute = nvmet_bdev_execute_rw;
        if (req->sq->ctrl->pi_support && nvmet_ns_has_pi(req->ns))
            req->metadata_len = nvmet_rw_metadata_len(req);
        return 0;
    case nvme_cmd_flush:
        req->execute = nvmet_bdev_execute_flush;
        return 0;
    case nvme_cmd_dsm:
        req->execute = nvmet_bdev_execute_dsm;
        return 0;
    case nvme_cmd_write_zeroes:
        req->execute = nvmet_bdev_execute_write_zeroes;
        return 0;
    default:
        return nvmet_report_invalid_opcode(req);
    }
}
```

LBA工具实现原理

- IO重试，参数：--io-retry (高级功能)　实例：微软iscsi target测试出数据不一致
- 存储整体trim，参数：--trim-all (高级功能)
- 识别文件存储的文件系统类型
- bsrange指定簇IO随机拆分范围 (适配4K盘)

```
[root@linux lba]# stat lba.raw
  File: lba.raw
  Size: 11811160064     Blocks: 22765488   IO Block: 65536  regular file
Device: 812h/2066d      Inode: 1291848976  Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2024-04-29 23:29:39.369332820 -0400
Modify: 2024-04-29 23:29:39.359332820 -0400
Change: 2024-04-29 23:29:39.359332820 -0400
 Birth: 2024-04-29 23:20:46.549332820 -0400
[root@linux lba]# stat lba.raw
  File: lba.raw
  Size: 11811160064     Blocks: 3136       IO Block: 65536  regular file
Device: 812h/2066d      Inode: 1291848976  Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2024-04-29 23:29:39.369332820 -0400
Modify: 2024-04-29 23:35:46.009332820 -0400
Change: 2024-04-29 23:35:46.009332820 -0400
 Birth: 2024-04-29 23:20:46.549332820 -0400
```




```
[root@linux lba]# ./hd_write_verify -c -D -S 1024 -A on -B 1-8 -V all -T 10 /root/lba/lba.raw
------------------------------------------------------------------------------------------------------------

File Information:
        FS type:    xfs
        File:       /root/lba/lba.raw
        Size:       4294967296
        Blocks:     0
        IO Block:   65536
        Device:     2066
        Inode:      1291848975
        Links:      1
        File Size: 4294967296 / 4096M / 4.00G

------------------------------------------------------------------------------------------------------------

File: /root/lba/lba.raw | Thread: 10 | Total Sectors: 8388608 | Total Clusters: 8192 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush | Policy: robin | Verify: 6

Current Time: 2024-05-13 09:43:17
Thread 9 [tid: 1586757]: Starting check file ...
Thread 4 [tid: 1586752]: Starting check file ...
Thread 8 [tid: 1586756]: Starting check file ...
Thread 1 [tid: 1586749]: Starting check file ...
Thread 2 [tid: 1586750]: Starting check file ...
Thread 6 [tid: 1586754]: Starting check file ...
Thread 3 [tid: 1586751]: Starting check file ...
Thread 7 [tid: 1586755]: Starting check file ...
Thread 5 [tid: 1586753]: Starting check file ...
Thread 0 [tid: 1586748]: Starting check file ...

------------------------------------------------------------------------------------------------------------

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2024-05-13 09:43:18
269, 230 | 304, 253 | 235, 207 | 236, 208 | 240, 210 | 283, 239 | 293, 252 | 228, 202 | 317, 262 | 232, 205 |
```

LBA工具实现原理

- IO精准限速：比较稳定的IO负载（高级功能）

应用场景举例：1. 虚拟机跨存储热迁移，多大的IO负载会导致迁移无法完成?
2. 虚拟机同存储热迁移，多大的内存负载(变脏速度)会导致迁移无法完成?

```
Welcome to the YOUPLUS's LBA TEST SYSTEM

https://github.com/zhangyoujia

YOUPLUS login: root (automatic login)
Last login: Sat Apr  1 11:06:21 GMT-8 2023 on tty2
echo never > /sys/kernel/mm/transparent_hugepage/enabled
root/YOUPLUS /var/iso/tools # hd_write_verify -c -D -K -a -S 1024 -L 40960 -V loop -T 10 /dev/vda
Device Topology:
        Disk: /dev/vda
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 10737418240 / 10240M / 10.00G

Disk: /dev/vda | Thread: 10 | Total Sectors: 20971520 | Total Clusters: 20480 | Sectors of Cluster:

Current Time: 2023-04-01 11:12:44
Thread 4 [tid: 9278]: Starting check disk ...
Thread 8 [tid: 9282]: Starting check disk ...
Thread 2 [tid: 9276]: Starting check disk ...
Thread 1 [tid: 9275]: Starting check disk ...
Thread 7 [tid: 9281]: Starting check disk ...
Thread 0 [tid: 9274]: Starting check disk ...
Thread 5 [tid: 9279]: Starting check disk ...
Thread 3 [tid: 9277]: Starting check disk ...
Thread 9 [tid: 9283]: Starting check disk ...
Thread 6 [tid: 9280]: Starting check disk ...

Starting write disk:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-04-01 11:12:45
0, 466 | 0, 466 | 0, 466 | 0, 466 | 0, 466 | 0, 466 | 0, 466 | 0, 466 | 0, 466 | 0, 466 |
Pause

Quit
0, 469 | 0, 470 | 0, 470 | 0, 470 | 0, 470 | 0, 469 | 0, 470 | 0, 470 | 0, 470 | 0, 470 |
Current Time: 2023-04-01 11:33:28
root/YOUPLUS /var/iso/tools #
```

```
root/YOUPLUS /var/iso/tools # iostat -x 1
Linux 4.18.0 (YOUPLUS)  04/01/23        _x86_64_        (2 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.51    0.00    3.05    2.54    0.00   93.91

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00    0.00   80.00     0.00 40960.00  1024.00     0.01    0.99    0.00    0.99   2.15  17.20
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
loop1             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.01    0.00    2.53    4.55    0.00   91.92

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00    0.00   80.00     0.00 40960.00  1024.00     0.06    1.54    0.00    1.54   2.50  20.00
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
loop1             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.02    0.00    4.06    8.63    0.00   86.29

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00    0.00   80.00     0.00 40960.00  1024.00     0.14    2.74    0.00    2.74   2.85  22.80
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
loop1             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.02    0.00    2.55    8.16    0.00   88.27

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00    0.00   80.00     0.00 40960.00  1024.00     0.28    4.81    0.00    4.81   3.39  26.80
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
loop1             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.51    0.00    3.54    7.07    0.00   88.89

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00    0.00   80.00     0.00 40960.00  1024.00     0.10    2.39    0.00    2.39   2.90  23.20
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
loop1             0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
```

LBA工具实现原理

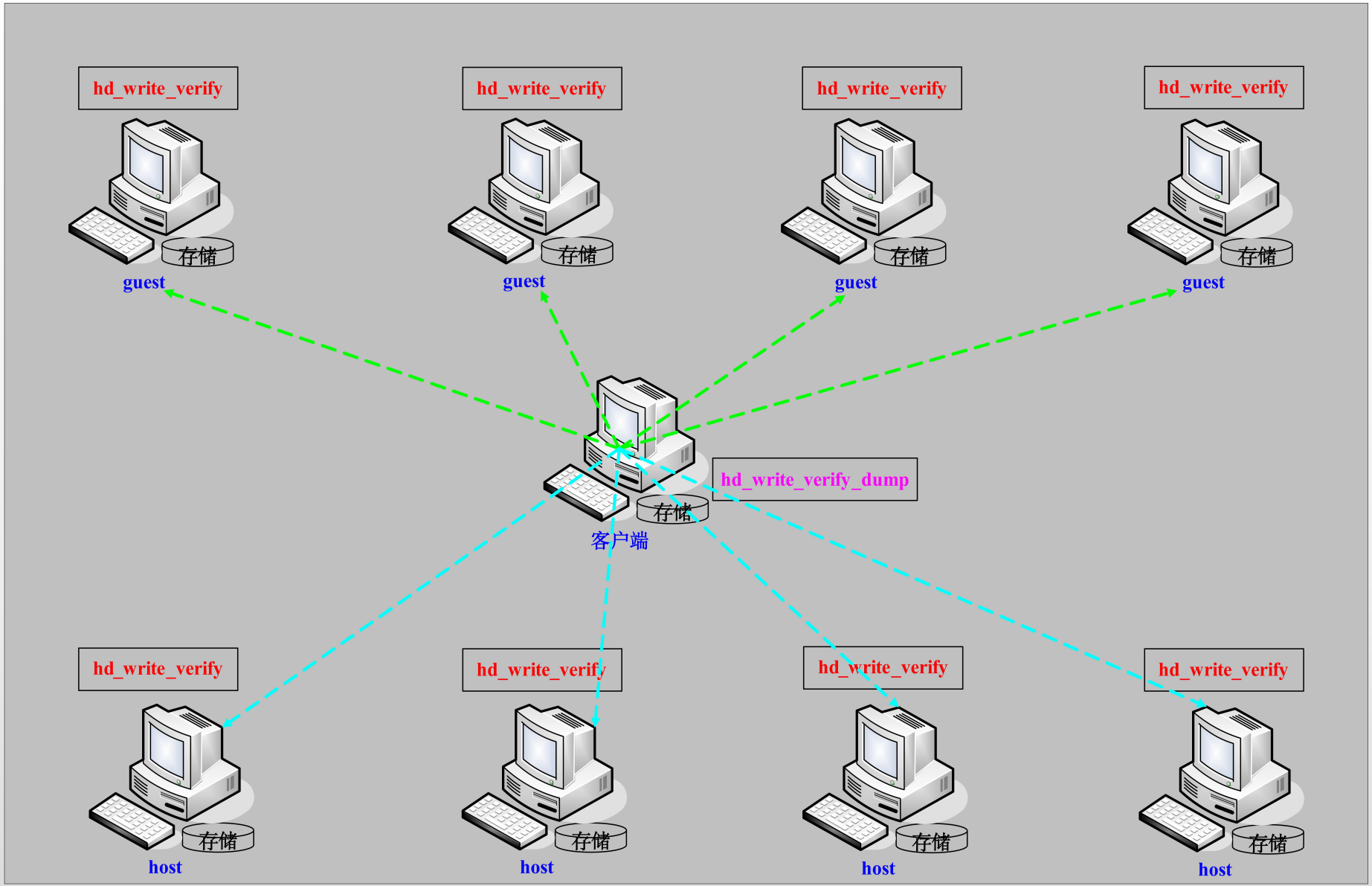
- 顺序IO: -s | --seq-io (适合用于测试大容量存储)

```
root@BYTEPLUS:~# hd_write_verify -c -D -s on -A on -S 32 -V all -T 10 lba.raw
--------------------------------------------------------------------------------

File Information:
        FS type:    ext4/ext3/ext2
        File:       lba.raw
        Size:       1073741824
        Blocks:     0
        IO Block:   4096
        Device:     8:1
        Inode:      7077894
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

--------------------------------------------------------------------------------

File: lba.raw | Thread: 10 | Total Sectors: 2097152 | Total Clusters: 65536 | Cluster Sectors

--------------------------------------------------------------------------------

Thread: 01 [tid: 1021]: Start verifying file ...
Thread: 00 [tid: 1020]: Start verifying file ...
Thread: 02 [tid: 1022]: Start verifying file ...
Thread: 03 [tid: 1023]: Start verifying file ...
Thread: 04 [tid: 1024]: Start verifying file ...
Thread: 05 [tid: 1025]: Start verifying file ...
Thread: 06 [tid: 1026]: Start verifying file ...
Thread: 07 [tid: 1027]: Start verifying file ...
Thread: 08 [tid: 1028]: Start verifying file ...
Thread: 09 [tid: 1029]: Start verifying file ...

Start verifying data: Thread ID | Read: MB - Write: MB | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Current Time: 2025-04-07 08:00:39

--------------------------------------------------------------------------------

Start writing file: Thread ID | Read: MB - Write: MB | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2025-04-07 08:00:39

64, 43 | 72, 48 | 95, 64 | 45, 30 | 45, 30 | 39, 26 | 44, 30 | 44, 30 | 41, 27 | 46, 31 | 15.76, 10.51 |
```

```
root@BYTEPLUS:~# hd_write_verify_dump -c -D -L 0x0 lba.raw
--------------------------------------------------------------------------------

File Information:
        FS type:    ext4/ext3/ext2
        File:       lba.raw
        Size:       1073741824
        Blocks:     1087688
        IO Block:   4096
        Device:     8:1
        Inode:      7077894
        Links:      1
        File Size: 1073741824 / 1024M / 1.00G

        Sector_Per_Cluster: 32
        Verify_Thread_NUM:  10

Current Time: 2025-04-07 07:41:42

================================================================================

LBA: 0x0 | HF: 0x02051a10 F1: 0x7b2ae6d5 F2: 0xc82bc568 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 32 / ZERO: 0]
IO Time: 2025-04-07 07:41:36

Buffer addr: 0xc6a000 | Physical addr: 0x6e2e8000

LBA LIST:
LBA[ 0]: 0x00000000 0x00199940 0x00199960 0x00199980 0x001999a0 0x001999c0 0x001999e0 0x00199a00 0x00199a20 0x00199a40
LBA[10]: 0x00199a60 0x00199a80 0x00199aa0 0x00199ac0 0x00199ae0 0x00199b00 0x00199b20 0x00199b40 0x00199b60 0x00199b80
LBA[20]: 0x00199ba0 0x00199bc0 0x00199be0 0x00199c00 0x00199c20 0x00199c40 0x00199c60 0x00199c80 0x00199ca0 0x00199cc0
LBA[30]: 0x00199ce0 0x00199d00 0x00199d20 0x00199d40 0x00199d60 0x00199d80 0x00199da0 0x00199dc0 0x00199de0 0x00199e00
LBA[40]: 0x00199e20 0x00199e40 0x00199e60 0x00199e80 0x00199ea0 0x00199ec0 0x00199ee0 0x00199f00 0x00199f20 0x00199f40
LBA[50]: 0x00199f60 0x00199f80 0x00199fa0 0x00199fc0 0x00199fe0 0x0019a000 0x0019a020 0x0019a040 0x0019a060 0x0019a080
LBA[60]: 0x0019a0a0 0x0019a0c0 0x0019a0e0 0x0019a100 0x0019a120 0x0019a140 0x0019a160 0x0019a180 0x0019a1a0 0x0019a1c0
LBA[70]: 0x0019a1e0 0x0019a200 0x0019a220 0x0019a240 0x0019a260 0x0019a280 0x0019a2a0 0x0019a2c0 0x0019a2e0 0x0019a300
LBA[80]: 0x0019a320 0x0019a340 0x0019a360 0x0019a380 0x0019a3a0 0x0019a3c0 0x0019a3e0 0x0019a400 0x0019a420 0x0019a440
LBA[90]: 0x0019a460 0x0019a480 0x0019a4a0 0x0019a4c0 0x0019a4e0 0x0019a500 0x0019a520 0x0019a540 0x0019a560 0x0019a580

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

================================================================================

Current Time: 2025-04-07 07:41:42

root@BYTEPLUS:~#
```

LBA工具实现原理

- remote-dump: 客户端(hd_write_verify_dump) => 通过网络链接访问 => 服务端(hd_write_verify) 获取测试结果



1. 目前的实现：多个hd_write_verify实例批量运行在物理主机或者虚拟机上，进行存储稳定性测试或者数据一致性校验，当查看每个实例的测试结果时，需要登录到每一台物理主机或者虚拟机上去查看，比较繁琐；

2. hd_write_verify作为服务端在物理主机或者虚拟机上运行，进行存储稳定性测试或者数据一致性校验，同时监听客户端的请求，把测试结果发送给hd_write_verify_dump客户端；

3. hd_write_verify_dump可以直接读取存储数据进行校验，也可以作为客户端，连接到hd_write_verify服务端，获取hd_write_verify工具稳定性测试或者数据一致性校验的结果；

4. hd_write_verify_dump作为客户端，可以批量连接同一台物理主机或者虚拟机上的多个hd_write_verify实例，来获取多个实例的测试结果，并进行格式化输出；

5. 实现此功能后，非常方便于 (shell脚本调用dump工具)自动化测试；

LBA工具实现原理

- remote-dump: 客户端(hd_write_verify_dump) => 通过网络链接访问 => 服务端(hd_write_verify) 获取测试结果

```
root@BYTEPLUS:~# hd_write_verify -c -D -S 1024 -V all -T 16 lba.raw

------------------------------------------------

      Welcome to the YOUPLUS's LBA TESTING SYSTEM

      https://github.com/zhangyoujia/

------------------------------------------------

------------------------------------------------------------------------------------------------------------------------

File Information:
        FS type:    ext4/ext3/ext2
        File:       lba.raw
        Size:       10737418240
        Blocks:     0
        IO Block:   4096
        Device:     253:0
        Inode:      1064198
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G  (MAX TEST: 1024.0T)

------------------------------------------------------------------------------------------------------------------------

File: lba.raw | Thread: 16 | Total Sectors: 20971520 | Total Clusters: 20480 | Cluster Sectors: 1024 | DIRECT IO & NO Flush | Policy

------------------------------------------------------------------------------------------------------------------------

Thread: 000 [tid: 13006]: Start verifying file ...
Thread: 001 [tid: 13007]: Start verifying file ...
Thread: 002 [tid: 13008]: Start verifying file ...
Thread: 003 [tid: 13009]: Start verifying file ...
Thread: 004 [tid: 13010]: Start verifying file ...
Thread: 005 [tid: 13011]: Start verifying file ...
Thread: 006 [tid: 13012]: Start verifying file ...
Thread: 007 [tid: 13013]: Start verifying file ...
Thread: 008 [tid: 13014]: Start verifying file ...
Thread: 009 [tid: 13015]: Start verifying file ...
Thread: 010 [tid: 13016]: Start verifying file ...
Thread: 011 [tid: 13017]: Start verifying file ...
Thread: 012 [tid: 13019]: Start verifying file ...
Thread: 013 [tid: 13021]: Start verifying file ...
Thread: 014 [tid: 13022]: Start verifying file ...
Thread: 015 [tid: 13023]: Start verifying file ...

Start verifying data: Thread ID | Read: MB/GB - Write: MB/GB | Total: GB | IOPS | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Current Time: 2025-11-04 19:22:09

------------------------------------------------------------------------------------------------------------------------

Start writing file: Thread ID | Read: MB/GB - Write: MB/GB | Total: GB | IOPS | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2025-11-04 19:22:09

171, 133 | 183, 137 | 190, 129 | 187, 137 | 200, 134 | 187, 125 | 162, 129 | 191, 128 | 212, 143 | 158, 128 | ... | Total: 2.9, 2.1
Pause

Start
248, 183 | 263, 190 | 255, 183 | 270, 193 | 284, 190 | 256, 180 | 264, 182 | 266, 184 | 280, 197 | 257, 179 | ... | Total: 4.1, 2.9
```

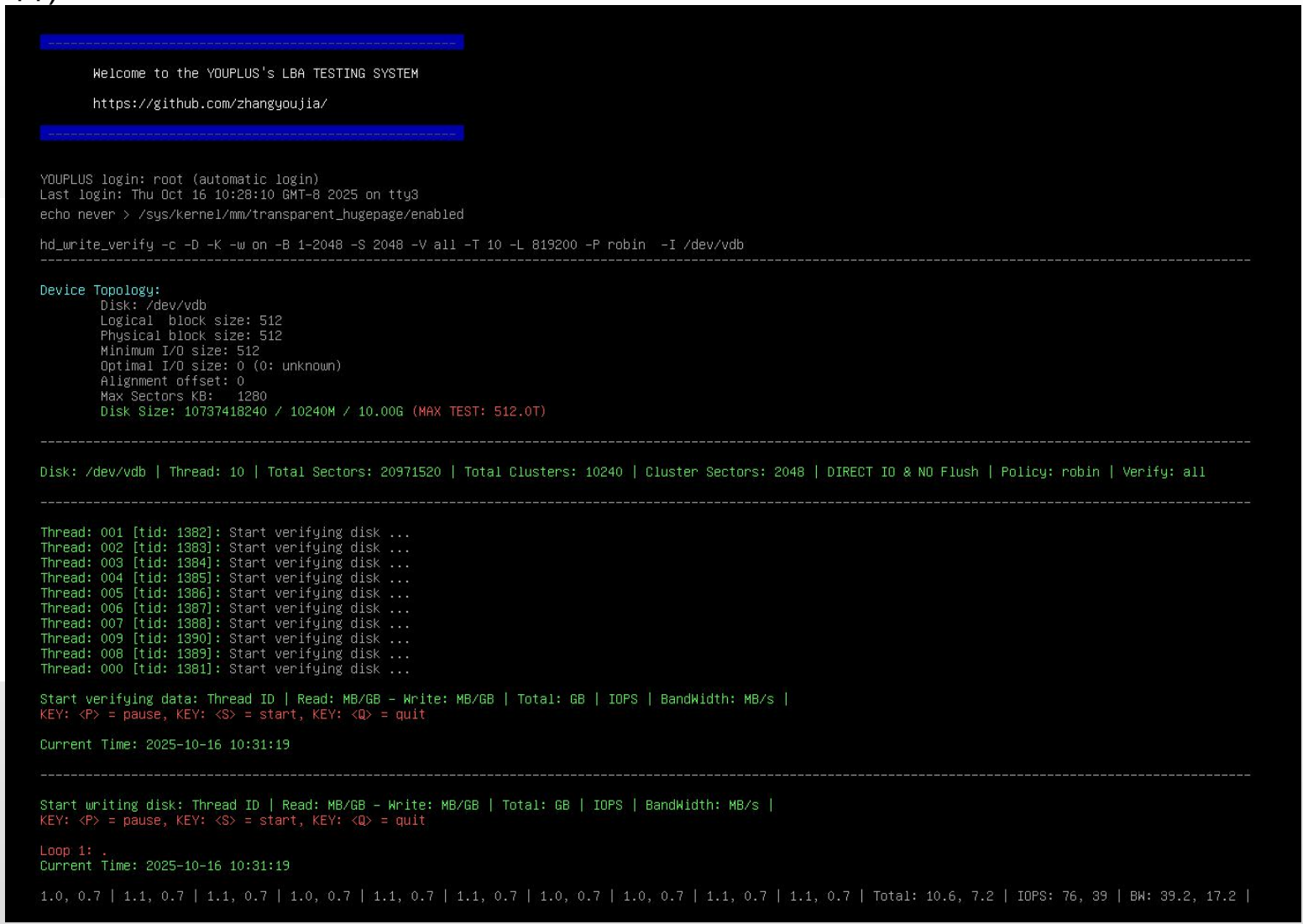
```
root@BYTEPLUS:~# hd_write_verify_dump -c -n 127.0.0.1

[127.0.0.1:2000]

------------------------------------------------

      Welcome to the YOUPLUS's LBA TESTING SYSTEM

      https://github.com/zhangyoujia/

------------------------------------------------

------------------------------------------------------------------------------------------------------------------------

File Information:
        FS type:    ext4/ext3/ext2
        File:       lba.raw
        Size:       10737418240
        Blocks:     0
        IO Block:   4096
        Device:     253:0
        Inode:      1064198
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G  (MAX TEST: 1024.0T)

------------------------------------------------------------------------------------------------------------------------

File: lba.raw | Thread: 16 | Total Sectors: 20971520 | Total Clusters: 20480 | Cluster Sectors: 1024 | DIRECT IO & NO Flush | Policy: robin | Verify: all

------------------------------------------------------------------------------------------------------------------------

Thread: 000 [tid: 13006]: Start verifying file ...
Thread: 001 [tid: 13007]: Start verifying file ...
Thread: 002 [tid: 13008]: Start verifying file ...
Thread: 003 [tid: 13009]: Start verifying file ...
Thread: 004 [tid: 13010]: Start verifying file ...
Thread: 005 [tid: 13011]: Start verifying file ...
Thread: 006 [tid: 13012]: Start verifying file ...
Thread: 007 [tid: 13013]: Start verifying file ...
Thread: 008 [tid: 13014]: Start verifying file ...
Thread: 009 [tid: 13015]: Start verifying file ...
Thread: 010 [tid: 13016]: Start verifying file ...
Thread: 011 [tid: 13017]: Start verifying file ...
Thread: 012 [tid: 13019]: Start verifying file ...
Thread: 013 [tid: 13021]: Start verifying file ...
Thread: 014 [tid: 13022]: Start verifying file ...
Thread: 015 [tid: 13023]: Start verifying file ...

Start verifying data: Thread ID | Read: MB/GB - Write: MB/GB | Total: GB | IOPS | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Current Time: 2025-11-04 19:22:09

------------------------------------------------------------------------------------------------------------------------

Start writing file: Thread ID | Read: MB/GB - Write: MB/GB | Total: GB | IOPS | BandWidth: MB/s |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2025-11-04 19:22:09

171, 133 | 183, 137 | 190, 129 | 187, 137 | 200, 134 | 187, 125 | 162, 129 | 191, 128 | 212, 143 | 158, 128 | ... | Total: 2.9, 2.1 | IOPS: 65, 61 | BW: 33.7, 21.1 |
Pause

Start
248, 183 | 263, 190 | 255, 183 | 270, 193 | 284, 190 | 256, 180 | 264, 182 | 266, 184 | 280, 197 | 257, 179 | ... | Total: 4.1, 2.9 | IOPS: 67, 65 | BW: 29.1, 20.0 |
```

LBA工具实现原理：特别版本未网上发布，可以单独申请试用

- LBA工具通用版本：只支持测试最大2T的存储(定义数据结构时，使用uint32_t lba[100]记录LBA地址，存在限制)
- LBA工具特别版本：支持测试最大1024T的存储(定义数据结构时，使用uint64_t lba[50]记录LBA地址，数据布局与LBA工具通用版本不兼容)

典型文件系统或Oracle：8KB
备份软件：64KB
流媒体：256K

应用类型	IO大小	读写比例	随机与顺序读写比例
OLTP-Data	8KB	70%读/30%写	100%随机
OLTP-Log	512B - 64KB	100%写	100%顺序
OLAP-TMP	256KB	50%读/50%写	100%随机
VDI	512B-16KB	20%读/80%写	100%顺序
Media Streaming	64KB	98%读/2%写	100%顺序
Web File Server	4KB、8KB、64KB	95%读/5%写	75%随机/25%顺序
Web Server Log	8KB	100% Write	100%顺序
OS Paging	64KB	90%读/10%写	100%顺序
Exchange Server	4KB	67%读/33%写	100%随机
Workstation	8KB	80%读/20%写	80%随机/20%顺序



LBA工具实现原理：特别版本未网上发布，可以单独申请试用

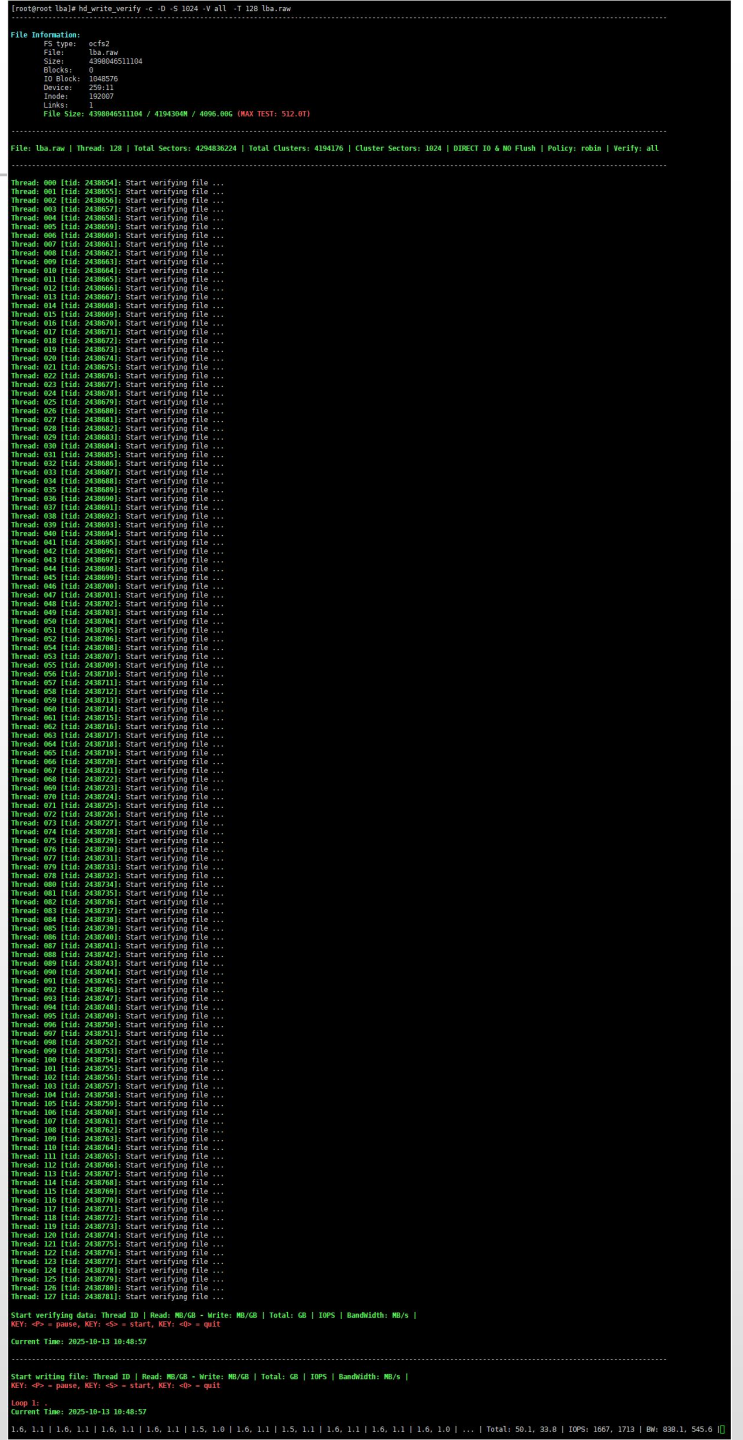
- LBA工具通用版本：默认只支持16个ioworker线程(可以通过激活程序修改配置)
- LBA工具特别版本：默认支持256个ioworker线程、最大4M簇大小等更多功能，可以测试更多极限场景；

```
Usage: hd_write_verify [opts] disk|file

-h | --help
-v | --version
-c | --color
-d | --daemon
-D | --direct-io
-F | --flush
-K | --dmesg
-T | --thread        0-16    [0:    random]
-n | --loop          0-2048  [0:   unlimit]
-S | --sector        2-2048  [default:   8]
-B | --bsrange       1-2048  [default: off]
-s | --seq-io        on|off  [default: off]
-A | --align-io      on|off  [default: off]
-r | --io-retry      on|off  [default: off]
-z | --zero-init     on|off  [default: off]
-I | --stripe        disk|file [stripe: 1-64]
-L | --bwlimit       KBPS [KBytes per second]
-Z | --zero-sectors  0-1024 [default: random]
-R | --zero-ratio    1-50   [default: 33 percent]
-U | --discard       on|off|random [default: off]
-t | --trim-all      on|off|random [default: off]
-O | --refill-data   on|off|random [default: off]
-W | --write-zeroes  on|off|random [default: off]
-w | --writeback     on|off|random [default: off]
-P | --policy        robin|split [default: robin]
-E | --timeout       0-3600 [default: 300 second]
-M | --write-limit   0-1024000 [default:  unlimit MB]
-V | --verify        check|once|loop|random|batch|all
-i | --inject-lba    off|flag|magic|pre|list|time|loop|
                     write|num|zero|index|random|sector

LICENSE PLATE:          license: 0x51a1, base: 0x51a10
CLUSTER SECTORS:        bit: 12, num: 2048
THREAD NUM:             bit: 4, num: 16, mask: 0xf
ACTIVE FUNCTION:        0x1
UNIQUE ID:              0xffffffffffffffff
```

LBA工具通用版本及激活信息

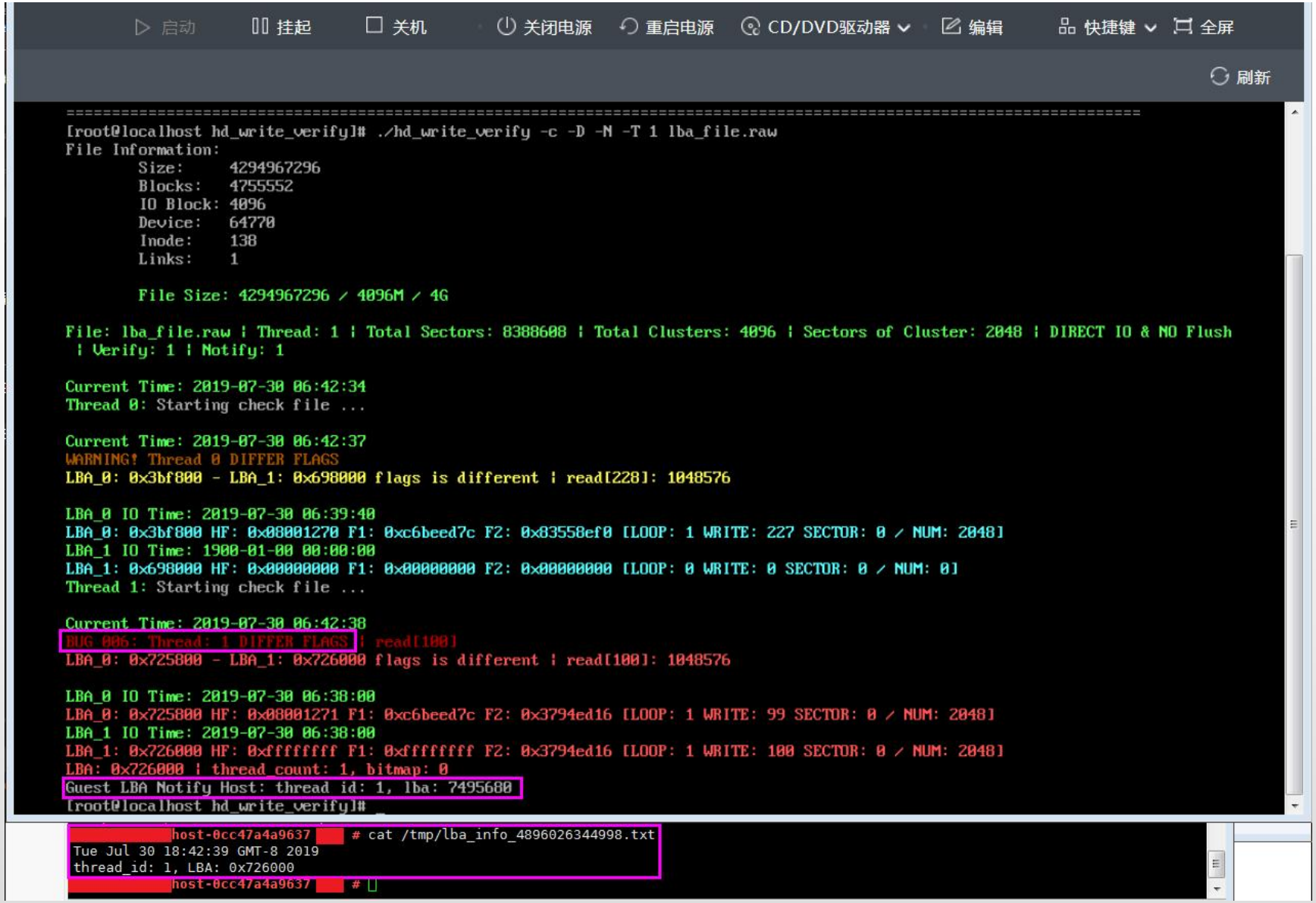
```
Usage: hd_write_verify [opts] disk|file

-h | --help
-v | --version
-c | --color
-d | --daemon
-D | --direct-io
-F | --flush
-K | --dmesg
-T | --thread        0-256   [0:    random]
-n | --loop          0-2048  [0:   unlimit]
-S | --sector        2-8192  [default:   8]
-B | --bsrange       1-8192  [default: off]
-s | --seq-io        on|off  [default: off]
-A | --align-io      on|off  [default: off]
-r | --io-retry      on|off  [default: off]
-z | --zero-init     on|off  [default: off]
-I | --stripe        disk|file [stripe: 1-64]
-L | --bwlimit       KBPS [KBytes per second]
-Z | --zero-sectors  0-4096 [default: random]
-R | --zero-ratio    1-50   [default: 33 percent]
-U | --discard       on|off|random [default: off]
-t | --trim-all      on|off|random [default: off]
-O | --refill-data   on|off|random [default: off]
-W | --write-zeroes  on|off|random [default: off]
-w | --writeback     on|off|random [default: off]
-P | --policy        robin|split [default: robin]
-E | --timeout       0-3600 [default: 300 second]
-M | --write-limit   0-1024000 [default:  unlimit MB]
-V | --verify        check|once|loop|random|batch|all
-i | --inject-lba    off|flag|magic|pre|list|time|loop|
                     write|num|zero|index|random|sector

LICENSE PLATE:          license: 0x51a1, base: 0x14600
CLUSTER SECTORS:        bit: 14, num: 8192
THREAD NUM:             bit: 8, num: 256, mask: 0xff
ACTIVE FUNCTION:        0xff
UNIQUE ID:              0xffffffffffffffff
```

LBA工具特别版本及激活信息



LBA工具实现原理

- 通知功能：guest中测试出数据一致性问题后，线程号和LBA地址通知到host--可实现短信、邮件告警 (高级功能)



LBA工具实现原理

- 虚拟内存地址与物理内存地址映射 (高级功能)

```
[root@localhost hd_write_verify]# ls -l
total 160
-rw------- 1 root root    0 Jun 13 17:05 lba1.raw.lock
-rw------- 1 root root    0 Jun 13 17:05 lba2.raw.lock
-rw------- 1 root root    0 Jun 13 17:05 lba3.raw.lock
-rw------- 1 root root    0 Jun 13 17:05 lba4.raw.lock
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_00_1579592
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_01_1579593
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_02_1579594
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_03_1579595
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_04_1579596
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_05_1579597
-rw-r--r-- 1 root root 4078 Jun 13 16:54 mem_map.1579591_06_1579598
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_07_1579599
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_08_1579600
-rw-r--r-- 1 root root 4183 Jun 13 16:54 mem_map.1579591_09_1579601
-rw-r--r-- 1 root root  112 Jun 13 16:54 mem_map.1579591_10_1579591
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_00_1579807
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_01_1579808
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_02_1579809
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_03_1579810
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_04_1579811
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_05_1579812
-rw-r--r-- 1 root root 4117 Jun 13 17:05 mem_map.1579805_06_1579813
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_07_1579814
-rw-r--r-- 1 root root 4183 Jun 13 17:05 mem_map.1579805_08_1579815
-rw-r--r-- 1 root root 4052 Jun 13 17:05 mem_map.1579805_09_1579816
-rw-r--r-- 1 root root  112 Jun 13 17:05 mem_map.1579805_10_1579805
[root@localhost hd_write_verify]#
```

```
[root@localhost hd_write_verify]# cat mem_map.1579591_00_1579592
verify_bitmap: ID: 0, Tid: 1579592, Thread: 0xffff9744f120
Buffer addr: 0xffff90010000 | Physical addr: 0x58349b30000

no_memalign: ID: 0, Tid: 1579592, Thread: 0xffff9744f120
Buffer addr: 0xffff949b0000 | Physical addr: 0x58321850000
Buffer addr: 0xffff949c0000 | Physical addr: 0x582b0dc0000
Buffer addr: 0xffff949d0000 | Physical addr: 0x583e88a0000
Buffer addr: 0xffff949e0000 | Physical addr: 0x58329ad0000
Buffer addr: 0xffff949f0000 | Physical addr: 0x583caf40000
Buffer addr: 0xffff94a00000 | Physical addr: 0x583f5500000
Buffer addr: 0xffff94a10000 | Physical addr: 0x583cca80000
Buffer addr: 0xffff94a20000 | Physical addr: 0x58318a60000
Buffer addr: 0xffff94a30000 | Physical addr: 0x58310000000
Buffer addr: 0xffff94a40000 | Physical addr: 0x5830fe70000
Buffer addr: 0xffff94a50000 | Physical addr: 0x58394730000
Buffer addr: 0xffff94a60000 | Physical addr: 0x582c5020000
Buffer addr: 0xffff94a70000 | Physical addr: 0x582bf910000
Buffer addr: 0xffff94a80000 | Physical addr: 0x583cd3c0000
Buffer addr: 0xffff94a90000 | Physical addr: 0x582a9840000
Buffer addr: 0xffff94aa0000 | Physical addr: 0x582b82a0000
Buffer addr: 0xffff94ab0000 | Physical addr: 0x58285240000

test_head: ID: 0, Tid: 1579592, Thread: 0xffff9744f120
Buffer addr: 0xffff7d540000 | Physical addr: 0x582bca60000
Buffer addr: 0xffff7d550000 | Physical addr: 0x582a1fb0000
Buffer addr: 0xffff7d560000 | Physical addr: 0x582b8ea0000
Buffer addr: 0xffff7d570000 | Physical addr: 0x582be010000
Buffer addr: 0xffff7d580000 | Physical addr: 0x582a1fe0000
Buffer addr: 0xffff7d590000 | Physical addr: 0x582a2200000
Buffer addr: 0xffff7d5a0000 | Physical addr: 0x582bf3a0000
Buffer addr: 0xffff7d5b0000 | Physical addr: 0x582a35b0000
Buffer addr: 0xffff7d5c0000 | Physical addr: 0x582a3590000
Buffer addr: 0xffff7d5d0000 | Physical addr: 0x582adbe0000
Buffer addr: 0xffff7d5e0000 | Physical addr: 0x582ae070000
Buffer addr: 0xffff7d5f0000 | Physical addr: 0x582adb00000
```

```
LBA: 0x5f1000 HF: 0xffffffff F1: 0xaaaaaaaa F2: 0xf043e3a4 [LOOP: 1 WRITE: 5  SE
CTOR: 0 / NUM: 20481
IO Time: 2020-03-03 01:25:21

Buffer addr: 0x7ffff7ef2000 | Physical addr: 0x3cd70000

Breakpoint 1, hd_verify_single_cluster (verify=0x7ffff7ef2000,
    sector_per_cluster=1, sector_per_cluster@entry=4096)
    at hd_write_verify_dump.c:728
728     in hd_write_verify_dump.c
```

200.201.184.10
200.201.24.154
200.201.33.40

10.103.128.90(my_pc)属性

名称	值
名称	10.103.128.90
类型	会话
主机	10.103.128.90
端口	22
协议	SSH

```
3CD6FEA0   40 00 00 00  00 00 00 00  E0 4E D7 00  00 00 00
3CD6FEC0   72 73 70 00  00 00 00 00  00 00 00 00  00 00 00
3CD6FEE0   64 61 74 61  5F 70 74 72  00 00 00 00  00 00 00
3CD6FF00   40 4F D7 00  00 00 00 00  08 00 00 00  00 00 00
3CD6FF20   40 00 00 00  00 00 00 00  60 4F D7 00  00 00 00
3CD6FF40   72 38 00 00  00 00 00 00  00 00 00 00  00 00 00
3CD6FF60   69 6E 74 36  34 00 00 00  00 00 00 00  00 00 00
3CD6FF80   C0 4F D7 00  00 00 00 00  11 00 00 00  00 00 00
3CD6FFA0   20 00 00 00  00 00 00 00  E0 4F D7 00  00 00 00
3CD6FFC0   65 66 6C 61  67 73 00 00  B8 9E E9 D8  7F 7F 00
3CD6FFE0   69 33 38 36  5F 65 66 6C  61 67 73 00  7F 7F 00
3CD70000   FF FF FF FF  AA AA AA AA  00 00 00 00  00 00 00
---  3733563982736.qemu_back_mem._objects_mem-node0.sdCMyD
```

```
cal addr: 0x582dae40000
cal addr: 0x582bac30000
cal addr: 0x58286850000
cal addr: 0x582a4030000

read: 0xffff9744f120
cal addr: 0x5828ab20000
cal addr: 0x582b4730000
cal addr: 0x5828b7e0000
cal addr: 0x582bd830000
cal addr: 0x582b4740000
```

目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

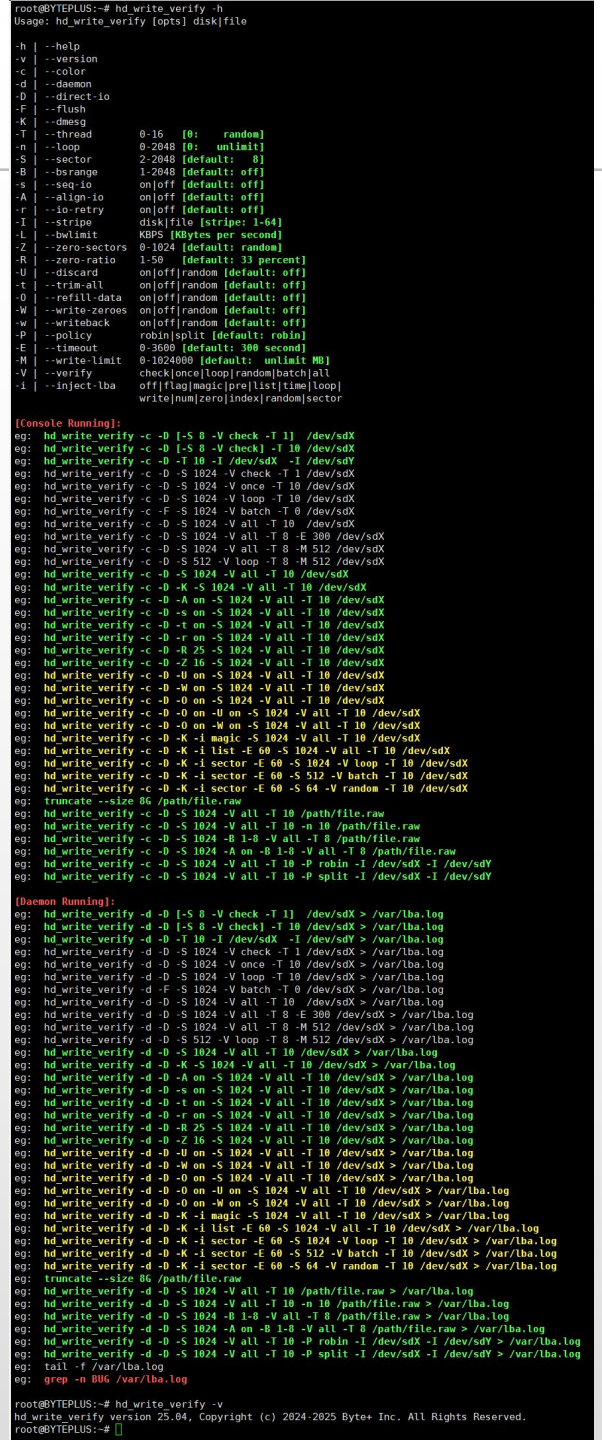
04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

LBA工具使用说明：--help

```
root@BYTEPLUS:~# hd_write_verify_dump -h
Usage: hd_write_verify_dump [opts] disk|file

-h | --help
-v | --version
-c | --color
-l | --layout
-D | --direct-io
-A | --align-io      on|off      [default:   on]
-L | --lba           sector_num  [dec/hex value]
-Q | --query         sector_num  [dec/hex value]
-C | --cluster       cluster_num [0:    unlimit]
-T | --thread        0-255[hd_write_verify -T N]
-S | --sector        sector_per_cluster [1-2048]
-I | --stripe        disk|file   [stripe:  1-64]
-P | --policy        robin|split [default:robin]
-Z | --zero-sectors  0-1024     [default: random]
-n | --network       ip[:port] [port: 2000-2015]
-i | --inject-lba    off|flag|magic|pre|list|time|loop|
                     write|num|zero|index|random|sector

[EXAMPLES]:
eg:  hd_write_verify_dump -c -n 192.168.0.1
eg:  hd_write_verify_dump -n 127.0.0.1:2000

eg:  hd_write_verify_dump -c -T 1  /dev/sdX
eg:  hd_write_verify_dump -c -D -T 1 /dev/sdX
eg:  hd_write_verify_dump -c -D -T 1 -Q lba /dev/sdX
eg:  hd_write_verify_dump -c -D -T 1 -C 100 /dev/sdX
eg:  hd_write_verify_dump -c -D -T 2 -C 100 -Z 16 /dev/sdX
eg:  hd_write_verify_dump -c -D -T 3 -C 100 /path/file.raw
eg:  hd_write_verify_dump -c -D -T 1 -I /dev/sdX -I /dev/sdY
eg:  hd_write_verify_dump -c -D -T 2 -P robin -I /dev/sdX -I /dev/sdY
eg:  hd_write_verify_dump -c -D -T 3 -P split -I /dev/sdX -I /dev/sdY

eg:  hd_write_verify_dump -c -D -L 0x0 -C 0   /dev/sdX
eg:  hd_write_verify_dump -c -D -L 0x0 -C 100 /dev/sdX
eg:  hd_write_verify_dump -c -L 0x8000 -C 100 /path/file.raw
eg:  hd_write_verify_dump -c -L 0x8000 -C 100 -I /dev/sdX -I /dev/sdY
eg:  hd_write_verify_dump -c -L 0x8000 -C 100 -P robin -I /dev/sdX -I /dev/sdY
eg:  hd_write_verify_dump -c -L 0x8000 -C 100 -P split -I /dev/sdX -I /dev/sdY

eg:  hd_write_verify_dump -c -D -L 0x0 -S 128 /dev/sdX
eg:  hd_write_verify_dump -c -L 0x200 -S 1024 /dev/sdX
eg:  hd_write_verify_dump -c -L 0x200 -S 1024 /path/file.raw
eg:  hd_write_verify_dump -c -L 0x200 -S 1024 -I /dev/sdX -I /dev/sdY

eg:  hd_write_verify_dump -c -D -L 0x10 [-S 1] /dev/sdX
eg:  hd_write_verify_dump -c -D -A off -L 0x9 -S 8 /dev/sdX
eg:  hd_write_verify_dump -c -D -L 0x10 -S 2048 /dev/sdX
eg:  hd_write_verify_dump -c -D -L 0x10 -S 2048 /path/file.raw
eg:  hd_write_verify_dump -D -L 10 -S 10 /dev/sdX > /var/dump.log
eg:  hd_write_verify_dump -c -D -L 0x100 -S 2048 -I /dev/sdX -I /dev/sdY

eg:  hd_write_verify_dump -c -D -i list -T 1 /dev/sdX
eg:  hd_write_verify_dump -c -D -i index -T 5 -C 100 /dev/sdX
eg:  hd_write_verify_dump -c -D -i magic -L 0x100000 /dev/sdX
eg:  hd_write_verify_dump -c -D -i sector -L 0x0 -C 100 /dev/sdX
```



hd_write_verify -h
测试与校验工具用法与用例

hd_write_verify_dump -h
校验与查看工具用法与用例

hd_write_verify_dump -l
数据布局与字段偏移

校验数据的三个重要命令：
hd_write_verify -c -D -T 1 disk/file

hd_write_verify -c -D -T 10 disk/file

hd_write_verify_dump -c -D -T 0-9 disk/file

远程查看测试进度和结果的命令：
hd_write_verify_dump -c -n ip
hd_write_verify_dump -c -n ip:port

LBA工具基本功能演示：hd_write_verify工具参数说明

参数	说明
console running (默认)	以前端控制台方式运行(可以交互: 暂停-P键 / 继续-S键 / 退出-Q键)
daemon running (-d | --daemon)	以后端守护进程方式运行，输出可以重定向到文件
-c | --color	以彩色输出关键信息(突出重点信息)
-D | --direct-io	以direct io方式读写，否则以pagecache方式读写
-F | --flush	当以pagecache方式读写时，刷新pagecache缓存
-K | --dmesg	LBA工具关键信息输出到dmesg日志
-A | --align-io on|off[default: off]	簇对齐IO(默认：关闭) --- 开启后，写IO簇不会被拆分 (高级功能)
-L | --bwlimit KBPS [KBytes per second]	IO限速，单位：KB/s
-S | --sector 2-2048 [default: 8]	指定一簇包含多少扇区，LBA工具读写IO以簇为单位
-n | --loop 0-4096 [0: unlimit]	指定测试轮数，如果设置为0，表示不限制，一直循环测试
-T | --thread 0-16 [0: random]	指定工作线程数，如果设置为0，表示工作线程数是随机的，每隔一分钟，工作的线程会进行切换，线程数也随机的变换
-V | --verify check|once|loop|random| batch|all	数据校验方式： 循环运行中校验：loop(全盘数据校验)，random(随机数据校验)，batch(批量数据校验 + 随机数据校验)，all(loop + batch) 校验完即退出：check(只读全盘数据校验) = loop 一次, once(只测试校验一次) = all 一次
-I | --stripe disk|file [stripe: 1-64]	条带，最多支持同时测试64个磁盘/文件，例如: -I /dev/sda -I /dev/sdb -I /dev/sdc
-P | --policy robin|split[default:robin]	条带策略: round-robin 和 cluster-split，多磁盘/文件测试时，条带策略才生效 (高级功能)
其它参数	高级功能，略

LBA工具基本功能演示：hd_write_verify_dump工具参数说明

参数	说明
-c | --color	以彩色输出关键信息(突出重点信息)
-l | --layout	输出数据布局和字段偏移及长度
-D | --direct-io	以direct io方式读写，否则以pagecache方式读写
-A | --align-io on|off [default: on]	簇对齐IO(默认：开启) --- 开启后，读IO簇对齐；特殊场景，需要以小于簇的单位读数据
-L | --lba sector_num [dec/hex value]	指定开始扇区号(支持十进制和十六进制)，读取数据进行校验或者输出查看
-S | --sector sector_per_cluster[1-2048]	簇内数据校验：指定扇区数 (默认值：1)
-C | --cluster cluster_num [0: unlimit]	簇间数据校验：指定校验数据的簇数(例如: -C 100，从指定的位置开始校验100簇数据)， 0: 表示一直校验到末尾 (默认值：1)
-T | --thread 0-9[hd_write_verify -T N]	线程数据校验(簇间数据校验)：指定线程号；另：hd_write_verify工具-T参数是指定线程数
-I | --stripe disk|file [stripe: 1-64]	条带，最多支持同时测试64个磁盘/文件，例如: -I /dev/sda -I /dev/sdb -I /dev/sdc
-P | --policy robin|split[default:robin]	条带策略: round-robin 和 cluster-split，多磁盘/文件测试时，条带策略才生效 (高级功能)
-Z | --zero-sectors 0-1024[default: random]	高级功能，略
-n | --network ip[:port] [default: 2000-2015]	高级功能，略
-i | --inject-lba off|flag|magic|pre|list|time|loop| write|num|zero|index|random|sector	高级功能：注入LBA错误，演示LBA工具测试校验数据一致性效果

LBA工具基本功能演示：hd_write_verify工具输出信息说明

```
[root@localhost lba]# hd_write_verify -c -D -S 1024 -V all -T 10 lba.raw
File Information:                    磁盘规格/文件信息
        File:       lba.raw
        Size:       4294967296
        Blocks:     0
        IO Block:   65536
        Device:     2051
        Inode:      1678094425
        Links:      1
        File Size: 4294967296 / 4096M / 4.00G

File: lba.raw | Thread: 10 | Total Sectors: 8388608 | Total Clusters: 8192 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush | Verify: 5 | Notify: 0
                                                    磁盘/文件信息及生效测试参数
Current Time: 2023-06-06 15:31:05   测试开始时间
Thread 6 [tid: 805477]: Starting check file ...
Thread 7 [tid: 805478]: Starting check file ...
Thread 0 [tid: 805471]: Starting check file ...
Thread 2 [tid: 805473]: Starting check file ...
Thread 9 [tid: 805480]: Starting check file ...   工作线程: 序号及ID
Thread 5 [tid: 805476]: Starting check file ...
Thread 3 [tid: 805474]: Starting check file ...
Thread 1 [tid: 805472]: Starting check file ...
Thread 8 [tid: 805479]: Starting check file ...
Thread 4 [tid: 805475]: Starting check file ...

-------------------------------------------------------------------------------------------------------------

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-06 15:31:05   第1轮测试及开始时间
17, 74 | 34, 72 | 19, 72 | 32, 71 | 20, 72 | 31, 71 | 31, 70 | 29, 72 | 34, 72 | 35, 73 |   统计信息: 10个工作线程，读写数据量，单位MB
Pause

Start 测试过程中可交互: 暂停 / 继续 / 退出
105, 129 | 108, 125 | 101, 122 | 107, 121 | 113, 125 | 112, 124 | 107, 121 | 97, 130 | 117, 128 | 119, 130 |
Quit

Current Time: 2023-06-06 15:31:32   测试结束时间
[root@localhost lba]#
```

安全测试措施：

1.加锁互斥：
每个磁盘/文件限制为单实例读写测试；

2. 文件系统挂载检测：
已挂载文件系统的磁盘不测试，防止误操作破坏磁盘数据；

3. 跨主机测试识别：
通过唯一性flag和时间戳识别测试者，防止乌龙操作，
举例：一个iscsi lun被同时映射到两台主机，因为没有互斥，跨主机同时测试同一个lun会导致误判，乌龙问题还很难排查；

LBA工具基本功能演示

- 完成第一轮存储测试及数据校验，没有检测出数据一致性问题，开始第二轮测试……第N轮测试

```
-------------------------------------------------------------------------------------------------------------------

Current Time: 2023-06-12 10:54:44
WARNING! [Thread: 0] VERIFY DIFFER FLAGS | filename: lba.raw
PREV LBA: 0x13b8000 - NEXT LBA: 0x321000 flags is different | read[959]

PREV LBA: 0x13b8000 | HF: 0x08001270 F1: 0xbef02254 F2: 0xbda73efe [LOOP: 1 WRITE: 958 SECTOR: 0 / NUM: 2048 / ZERO: 176]
PREV LBA IO Time: 2023-06-12 10:54:00

NEXT LBA: 0x321000 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
NEXT LBA IO Time: 1900-01-00 00:00:00

-------------------------------------------------------------------------------------------------------------------

Current Time: 2023-06-12 10:54:45
WARNING! [Thread: 2] VERIFY DIFFER FLAGS | filename: lba.raw
PREV LBA: 0x2ea000 - NEXT LBA: 0xaa4800 flags is different | read[941]

PREV LBA: 0x2ea000 | HF: 0x08001272 F1: 0xbef02254 F2: 0xe58elecd [LOOP: 1 WRITE: 940 SECTOR: 0 / NUM: 2048 / ZERO: 45]
PREV LBA IO Time: 2023-06-12 10:54:00

NEXT LBA: 0xaa4800 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
NEXT LBA IO Time: 1900-01-00 00:00:00

-------------------------------------------------------------------------------------------------------------------

Current Time: 2023-06-12 10:54:45
WARNING! [Thread: 7] VERIFY DIFFER FLAGS | filename: lba.raw
PREV LBA: 0xbf4800 - NEXT LBA: 0x9c6000 flags is different | read[963]

PREV LBA: 0xbf4800 | HF: 0x08001277 F1: 0xbef02254 F2: 0xe9ba0e40 [LOOP: 1 WRITE: 962 SECTOR: 0 / NUM: 2048 / ZERO: 544]
PREV LBA IO Time: 2023-06-12 10:54:00

NEXT LBA: 0x9c6000 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
NEXT LBA IO Time: 1900-01-00 00:00:00

-------------------------------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------------------------------

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 2: ..
Current Time: 2023-06-12 10:54:45
0, 708 | 0, 705 | 0, 706 | 0, 731 | 0, 689 | 0, 704 | 0, 709 | 0, 730 | 0, 727 | 0, 703 |
```

LBA工具基本功能演示：hd_write_verify_dump工具输出信息说明

```
[root@localhost lba]# hd_write_verify_dump -c -D -T 5 lba.raw
File Information:                    磁盘规格/文件信息
        File:       lba.raw
        Size:       10737418240
        Blocks:     20918840
        IO Block:   65536
        Device:     2051
        Inode:      1678059328
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G

        Sector_Per_Cluster: 2048  簇大小：2048扇区

Current Time: 2023-06-12 10:59:31  数据校验开始时间

================================================================================================================
        5号线程                              本次数据校验共读取827次IO
WARNING: [Thread: 5] DIFFER FLAGS | read[827] | filename: lba.raw
LBA_0: 0xaa4800 - LBA_1: 0x754800 flags is different

----------------------------------------------------------------------------------------------------------------
LBA地址：扇区号          硬编码flag、全局魔数、每轮测试魔数          第3轮测试 第826次写IO 每簇数据2048扇区，当前数据簇内偏移0扇区
LBA: 0xaa4800 | HF: 0x08001275 F1: 0xbef02254 F2: 0xd6dfda70 [LOOP: 3 WRITE: 826 SECTOR: 0 / NUM: 2048 / ZERO: 294]
IO Time: 2023-06-12 10:59:08  LBA工具写入本簇数据的IO时间                                  高级功能：全0数据扇区数

Buffer addr: 0xffffb16d0000 | Physical addr: 0x3006ebc0000    高级功能：虚拟地址与物理地址的映射

LBA LIST:本簇数据的LBA地址      99个指针：指向后向节点的LBA地址
LBA[ 0]: 0x00aa4800 0x00754800 0x012a8800 0x0126b000 0x01272000 0x0087c000 0x0011a000 0x00e89000 0x006e6000 0x012ad800
LBA[10]: 0x00232800 0x00577800 0x001cc800 0x00afe800 0x00a7b800 0x0073c000 0x00e4d000 0x00c6e000 0x00e2e000 0x01386800
LBA[20]: 0x0134c800 0x007c5800 0x0029e800 0x0016c000 0x012e5800 0x00ccf000 0x00a2d000 0x003eb800 0x0125c000 0x0139b000
LBA[30]: 0x004e0800 0x00b01800 0x00ce7000 0x0069e000 0x013aa800 0x00fec800 0x00dc2000 0x01241800 0x012af800 0x008e5800
LBA[40]: 0x00b13800 0x008d3000 0x00266800 0x013e6800 0x001c4000 0x007ec800 0x0053d800 0x010e1800 0x00d0b800 0x00681800
LBA[50]: 0x00b73800 0x00dc7000 0x01360800 0x01078800 0x0122f000 0x01345800 0x004a1000 0x00746000 0x000c1800 0x011a8000
LBA[60]: 0x008eb000 0x00cdc000 0x00a33800 0x00e62800 0x00b4d800 0x00a4e000 0x002f8800 0x0085a800 0x008ec000 0x00c05800
LBA[70]: 0x00aa7000 0x007f6800 0x00772800 0x00a45800 0x0055a800 0x00893800 0x00109000 0x00563000 0x012b8800 0x013fa000
LBA[80]: 0x00797800 0x00870800 0x005df000 0x00c07800 0x00e9b800 0x0115c800 0x00ad8000 0x00269800 0x0074f800 0x00dc8800
LBA[90]: 0x009e3800 0x00751800 0x0038c800 0x00c0b800 0x00dc9800 0x00885000 0x0064f000 0x00e7e800 0x009be000 0x0014c800

PRE LBA LIST:10个指针：指向前向节点的LBA地址
LBA[ 0]: 0x00f0e800 0x01392000 0x0136e800 0x00efa800 0x001ae800 0x01179800 0x00c21800 0x0132d800 0x0077e800 0x00fe4000

----------------------------------------------------------------------------------------------------------------
LBA地址：扇区号          硬编码flag、全局魔数、每轮测试魔数          第2轮测试 第333次写IO 每簇数据2048扇区，当前数据簇内偏移0扇区
LBA: 0x754800 | HF: 0x08001275 F1: 0xbef02254 F2: 0xe2a4c11f [LOOP: 2 WRITE: 333 SECTOR: 0 / NUM: 2048 / ZERO: 159]
IO Time: 2023-06-12 10:55:26  LBA工具写入本簇数据的IO时间                                  高级功能：全0数据扇区数

Buffer addr: 0xffffb17f0000 | Physical addr: 0x3008ce70000    高级功能：虚拟地址与物理地址的映射

LBA LIST:本簇数据的LBA地址      99个指针：指向后向节点的LBA地址
LBA[ 0]: 0x00754800 0x01271800 0x009c0000 0x00576800 0x00b3d000 0x00961000 0x00f29800 0x0101f000 0x00c1a800 0x00bf4000
LBA[10]: 0x00980800 0x00bda000 0x00fad800 0x00781800 0x00f9c800 0x00336800 0x00add000 0x00c80800 0x008b0800 0x0098b000
LBA[20]: 0x01388800 0x00e90000 0x00981000 0x01321800 0x011d7000 0x01084800 0x00aa7000 0x011b0800 0x0067f800 0x00f0f800
LBA[30]: 0x00731000 0x00ca1000 0x00d0c800 0x00e76000 0x0058f000 0x007f0800 0x00410000 0x0094f000 0x009d1000 0x0032e800
LBA[40]: 0x00fe7000 0x00e63000 0x01022800 0x008cc800 0x00778000 0x00181000 0x00650800 0x012a9800 0x013f7000 0x00129000
LBA[50]: 0x0041d000 0x00670000 0x00c82800 0x011cb800 0x00040000 0x00bd7800 0x002c9000 0x00ed0800 0x00844000 0x00f36800
LBA[60]: 0x005cb800 0x00fe1800 0x00b9d000 0x00d82000 0x00ef3800 0x011d5000 0x00716000 0x00c18000 0x00897000 0x00438000
LBA[70]: 0x00e80000 0x01125000 0x009e5800 0x00bdd000 0x009a7000 0x00c90800 0x00b29800 0x007cf000 0x0074f800 0x01150800
LBA[80]: 0x00048000 0x00d79000 0x00cb4800 0x011ca000 0x00ae5000 0x00a19000 0x010c5000 0x0078a800 0x00074800 0x00720000
LBA[90]: 0x01112800 0x00420800 0x00e75000 0x0028f800 0x0121e000 0x002e1000 0x00f16000 0x003dc800 0x007a6800 0x00e4b800

PRE LBA LIST:10个指针：指向前向节点的LBA地址
LBA[ 0]: 0x00067000 0x0059f800 0x00222800 0x009f6000 0x0117d000 0x00768000 0x00ca0800 0x00dbe000 0x007e6000 0x00fba000

================================================================================================================

Current Time: 2023-06-12 10:59:41  数据校验结束时间

[root@localhost lba]#
```

```
DATA LAYOUT:
================================================

offsetof HF:                    0    [4   bytes]
offsetof MAGIC:                 4    [4   bytes]
offsetof PRE_LBA[10]:           8    [40  bytes]
offsetof LBA_LIST[100]:         48   [400 bytes]
offsetof TIME:                  448  [56  bytes]
offsetof RANDOM:                504  [4   bytes]
offsetof NUM:                   508  [2   bytes]
offsetof INDEX:                 510  [2   bytes]

================================================
```

```
/*
 *  ----------------    ----------------    ----------------    ----------------
 *  | HF           |    | HF           |    | HF           |    | HF           |
 *  ----------------    ----------------    ----------------    ----------------
 *  | MAGIC        |    | MAGIC        |    | MAGIC        |    | MAGIC        |
 *  ----------------    ----------------    ----------------    ----------------
 *  | PRE_LBA[10]  |    | PRE_LBA[10]  |    | PRE_LBA[10]  |    | PRE_LBA[10]  |
 *  ----------------    ----------------    ----------------    ----------------
 *  | LBA_LIST[100]|    | LBA_LIST[100]|    | LBA_LIST[100]|    | LBA_LIST[100]|
 *  ----------------    ----------------    ----------------    ----------------
 *  | TIME         |    | TIME         |    | TIME         |    | TIME         |
 *  ----------------    ----------------    ----------------    ----------------
 *  | RANDOM       |    | RANDOM       |    | RANDOM       |    | RANDOM       |
 *  ----------------    ----------------    ----------------    ----------------
 *  | NUM          |    | NUM          |    | NUM          |    | NUM          |
 *  ----------------    ----------------    ----------------    ----------------
 *  | INDEX        |    | INDEX        |    | INDEX        |    | INDEX        |
 *  ----------------    ----------------    ----------------    ----------------
 *
 *  |<---sector---->|    |<---sector---->|   ...  |<---sector---->|    |<---sector---->|
 *  |<-------------------------------------------cluster-------------------------------------------->|
 *
 *  |-----------------------------------------------------------------------------------------------|
 *  |   cluster    |   cluster    |   cluster ... |   cluster    |   cluster    |   cluster    |
 *  |-----------------------------------------------------------------------------------------------|
 *  |<-----------------------------------------------disk----------------------------------------->|
 */
```

LBA工具基本功能演示

- 数据校验错误类型表：(关键字：BUG 00X)

错误类型	参数	校验类型	校验时间	数据校验错误说明
BUG 001[1]	-V random	随机校验：簇间校验	runtime	FLAGS数据校验出错
BUG 001[2]	-V random	随机校验：簇间校验	runtime	LBA_LIST数据校验出错
BUG 002[1]	-V random	随机校验：簇内校验	runtime	非零扇区数据校验出错
BUG 002[2]	-V random	随机校验：簇内校验	runtime	全零扇区数据校验出错
BUG 001[3]	-V batch	批量校验：簇间校验	runtime	FLAGS数据校验出错
BUG 001[4]	-V batch	批量校验：簇间校验	runtime	LBA_LIST数据校验出错
BUG 002[3]	-V batch	批量校验：簇内校验	runtime	非零扇区数据校验出错
BUG 002[4]	-V batch	批量校验：簇内校验	runtime	全零扇区数据校验出错
BUG 003	-V loop	整体校验：簇间校验	runtime & checktime	PRE_LIST数据 或者 LBA_LIST数据校验出错
BUG 004	-V loop	整体校验：簇间校验	runtime & checktime	中间簇数据丢失
BUG 005	-V loop	整体校验：簇内校验	runtime	第一簇数据魔数出错
BUG 006	-V loop	整体校验：簇间校验	runtime	簇间FLAGS数据校验出错
BUG 007[1]	-V loop	整体校验：簇内校验	runtime & checktime	非零扇区：中间扇区的数据与前后扇区的数据不一致
BUG 007[2]	-V loop	整体校验：簇内校验	runtime & checktime	全零扇区：中间扇区的数据与前后扇区的数据不一致
BUG 007[3]	-V loop	整体校验：簇内校验	runtime & checktime	前后簇数据正确，中间簇非零扇区数据校验出错
BUG 007[4]	-V loop	整体校验：簇内校验	runtime & checktime	前后簇数据正确，中间簇全零扇区数据校验出错

LBA工具基本功能演示

- BUG 001[1]：案例一，inject SECTOR lba bug

批量校验数据，簇间flag数据不一致

BUG 001[1]: [Thread: 1] RANDOM VERIFY DIFFER FLAGS | ERROR LBA: 0x794400 | filename: lba.raw
LBA: 0x794400 | thread_num: 10, bitmap: 0 1 0 0 0 0 0 0 0 0

PREV LBA: 0xe03800 | HF: 0x00401271 F1: 0x76f16720 F2: 0xa09e9c22 [LOOP: 1 WRITE: 951 SECTOR: 0 / NUM: 64 / ZERO: 4]
PREV IO Time: 2023-07-03 19:14:28

LOST LBA: 0x794400 | HF: 0x40404040 F1: 0x40404040 F2: 0x40404040 [LOOP: 1077952576 WRITE: 1077952576 SECTOR: 16448 / NUM: 16448 / ZERO: 1077952576]
LOST IO Time: 1077954476-1077952577-1077952576

NEXT LBA: 0xe22640 | HF: 0x00401271 F1: 0x76f16720 F2: 0xa09e9c22 [LOOP: 1 WRITE: 953 SECTOR: 0 / NUM: 64 / ZERO: 4]
NEXT IO Time: 2023-07-03 19:14:29
---runtime校验

```
BUG 004: [Thread: 1] DATA LOST | read[953] | filename: lba.raw
PREV LBA: 0xe03800 | LOST LBA: 0x794400 | NEXT LBA: 0xe22640

PREV LBA: 0xe03800 | HF: 0x00401271 F1: 0x76f16720 F2: 0xa09e9c22 [LOOP: 1 WRITE:
PREV IO Time: 2023-07-03 19:14:28

LOST LBA: 0x794400 | HF: 0x40404040 F1: 0x40404040 F2: 0x40404040 [LOOP: 107795257
LOST IO Time: 1077954476-1077952577-1077952576 1077952576:1077952576:1077952576

NEXT LBA: 0xe22640 | HF: 0x00401271 F1: 0x76f16720 F2: 0xa09e9c22 [LOOP: 1 WRITE:
NEXT IO Time: 2023-07-03 19:14:29
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-03 19:13:29
29, 33 | 30, 33 | 29, 33 | 29, 33 | 28, 32 | 30, 33 | 29, 33 | 30, 33 | 30, 33 | 29, 33 |

Current Time: 2023-07-03 19:14:34
BUG 001[1]: [Thread: 1] RANDOM VERIFY DIFFER FLAGS | ERROR LBA: 0x794400 | filename: lba.raw
LBA: 0x794400 | thread_num: 10, bitmap: 0 1 0 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x11eb5c0 | HF: 0x00401271 F1: 0x76f16720 F2: 0xa09e9c22 [LOOP: 1 WRITE: 1037 SECTOR: 0 / NUM: 64 / ZERO: 4]
LAST WRITE IO Time: 2023-07-03 19:14:34

Buffer addr: 0xffff8c050000 | Physical addr: 0x481fc130000

LBA LIST:
LBA[ 0]: 0x011eb5c0 0x011e8a00 0x00bb6680 0x00e9d700 0x0075bb00 0x010b8e00 0x01199000 0x008eca00 0x00a4c5c0 0x00f2a340
LBA[10]: 0x00095a00 0x00a75bc0 0x008b9880 0x00d80b00 0x012095c0 0x01358340 0x01236480 0x0119a300 0x01185900 0x00b40f80
LBA[20]: 0x00b4e100 0x00ff9000 0x0029c780 0x00cff900 0x00884200 0x0120bcc0 0x0082d5c0 0x0132ce40 0x00d04700 0x0013c840
LBA[30]: 0x008738c0 0x012ad2c0 0x00184380 0x00adaa00 0x0088abc0 0x00b92000 0x00ae2a80 0x00079c40 0x013b0780 0x00291e00
LBA[40]: 0x005f4e00 0x00c50880 0x010a4640 0x01327f40 0x00dca980 0x0072dec0 0x003bdc80 0x00230c80 0x0042e8c0 0x0021e7c0
LBA[50]: 0x01013300 0x005fe880 0x00044400 0x006aa000 0x00a82400 0x009fec00 0x001aafc0 0x00669300 0x00e1b680 0x00c7bbc0
LBA[60]: 0x0047b900 0x00094280 0x005ff040 0x00d40c80 0x00d2c600 0x0102ae40 0x0067b6c0 0x000d7780 0x00dce780 0x00365c00
LBA[70]: 0x01337080 0x002f2400 0x0108b600 0x00bdf000 0x00e03700 0x00850f40 0x00a83000 0x0098f2c0 0x001dff00 0x00847c40
LBA[80]: 0x0013a100 0x01175e00 0x013251c0 0x011f2880 0x00e94b00 0x00df3a00 0x0102b200 0x00eef8c0 0x00a64100 0x0051fd40
LBA[90]: 0x01259900 0x00174640 0x000bab00 0x00629680 0x004fdf40 0x00d537c0 0x010a9cc0 0x00524200 0x00da4f00 0x00851fc0

PRE LBA LIST:
LBA[ 0]: 0x0100c980 0x00e4ab80 0x00a03500 0x00c28e80 0x007e5580 0x00acb080 0x00d30c00 0x00b215c0 0x001cf300 0x00384500

DETECT READ ERROR:
LBA: 0x794400 | HF: 0x40404040 F1: 0x40404040 F2: 0x40404040 [LOOP: 1077952576 WRITE: 1077952576 SECTOR: 16448 / NUM: 16448
DETECT ERROR IO Time: 1077954476-1077952577-1077952576 1077952576:1077952576:1077952576

Buffer addr: 0xffff8c070000 | Physical addr: 0x481cdc80000

LBA LIST:
LBA[ 0]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[10]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[20]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[30]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[40]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[50]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[60]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[70]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[80]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040
LBA[90]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

PRE LBA LIST:
LBA[ 0]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040


Current Time: 2023-07-03 19:14:34

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 001[1]：案例二，inject FLAG lba bug

随机校验数据，簇间flag数据不一致

BUG 001[1]: [Thread: 0] RANDOM VERIFY DIFFER FLAGS |
ERROR LBA: 0x0 | filename: test.raw
LBA: 0x0 | thread_num: 10, bitmap: 1 0 0 0 0 0 0 0 0 0

Line: 2572 | HF: 0x4001270 != hd_flag: 0x400ffff, sector_num: 1024

LBA: 0x0 | HF: 0x0400ffff F1: 0x523f39d9 F2: 0xf71538a0 [LOOP: 1
WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 255]
IO Time: 2023-07-09 23:33:05

---runtime校验

```
Line: 2572 | HF: 0x4001270 != hd_flag: 0x400ffff, sector_num: 1024

LBA: 0x0 | HF: 0x0400ffff F1: 0x523f39d9 F2: 0xf71538a0 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 /
IO Time: 2023-07-09 23:33:05

Buffer addr: 0x1f810000 | Physical addr: 0x302d2550000

LBA LIST:
LBA[ 0]: 0x00000000 0x006c6c00 0x002a6000 0x00565000 0x0049bc00 0x00659000 0x007fe000 0x004c0000
LBA[10]: 0x00198c00 0x004e8400 0x0003f800 0x00530000 0x00738000 0x00239800 0x007a5800 0x00080400
LBA[20]: 0x000cd400 0x0060b800 0x00242000 0x0018a800 0x00080000 0x0041fc00 0x00207000 0x00618000
LBA[30]: 0x005a5400 0x00091400 0x002e9400 0x0044fc00 0x00050000 0x004f9c00 0x0049a800 0x00320000
LBA[40]: 0x004bc000 0x006c9000 0x005fd000 0x00497000 0x00142000 0x005f1800 0x00320800 0x001b5800
LBA[50]: 0x00675800 0x00739c00 0x00245000 0x00378400 0x00198000 0x0009b800 0x002f4c00 0x005a3400
LBA[60]: 0x002e4000 0x007e4000 0x00036400 0x005ff000 0x001f3800 0x00410c00 0x00329800 0x00450400
LBA[70]: 0x004c1000 0x003ed000 0x0047f000 0x00279c00 0x0064e800 0x0001c000 0x003bbc00 0x004a2000
LBA[80]: 0x00182000 0x00036800 0x000b0000 0x0064f000 0x005f7000 0x00367000 0x004fc000 0x0022ac00
LBA[90]: 0x0028e800 0x0010a000 0x006bd800 0x0060a000 0x0057a400 0x00530c00 0x0067a000 0x0051f800

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
```

---checktime校验

```
158, 165 | 175, 170 | 160, 156 | 162, 174 | 164, 166 | 153, 160 | 182, 172 | 171, 164 | 189, 176 | 157, 166 |
--------------------------------------------------------------------------------------------------------------

Current Time: 2023-07-09 23:36:12
BUG 001[1]: [Thread: 0] RANDOM VERIFY DIFFER FLAGS | ERROR LBA: 0x0 | filename: test.raw
LBA: 0x0 | thread_num: 10, bitmap: 1 0 0 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x4b5000 | HF: 0x04001270 F1: 0x523f39d9 F2: 0xf71538a0 [LOOP: 1 WRITE: 282 SECTOR: 0 / NUM: 1024 / ZERO: 255]
LAST WRITE IO Time: 2023-07-09 23:36:09

Buffer addr: 0xffffb80f0000 | Physical addr: 0x183ef3e0000

LBA LIST:
LBA[ 0]: 0x004b5000 0x001c4000 0x0053e000 0x00685800 0x000d4c00 0x0038d800 0x003e9000 0x001f1800 0x00623000 0x003d4800
LBA[10]: 0x005eac00 0x005c0000 0x0073c000 0x005c1000 0x002d8400 0x00352800 0x007ff000 0x002b9000 0x004c0c00 0x00067400
LBA[20]: 0x00228400 0x0015a800 0x00617c00 0x0055f000 0x00055400 0x0036a000 0x00651c00 0x0074b800 0x00342000 0x00783400
LBA[30]: 0x004eac00 0x00179c00 0x007dbc00 0x004c9800 0x0017c400 0x00143000 0x004a6c00 0x00118c00 0x00456000 0x003b1800
LBA[40]: 0x005bc800 0x005dd400 0x004c3400 0x00296000 0x001d0400 0x004d9800 0x00009800 0x002cf400 0x007e0800 0x00185000
LBA[50]: 0x0032b000 0x0079ac00 0x00144c00 0x006d5400 0x00771800 0x006a9800 0x0008c800 0x005ac800 0x002c3400 0x00611400
LBA[60]: 0x003a1400 0x007b9c00 0x006c7800 0x004b5c00 0x003e1c00 0x00434c00 0x00289c00 0x00020000 0x0042dc00 0x00423400
LBA[70]: 0x006dbc00 0x004e7000 0x002c1400 0x00420400 0x0015c000 0x00652400 0x004e0400 0x004d7000 0x00242c00 0x00693000
LBA[80]: 0x00539400 0x0040f400 0x000c9000 0x00401000 0x007e5800 0x00211800 0x00330000 0x00409400 0x004df000 0x0056c400
LBA[90]: 0x0053a000 0x00377800 0x005fa400 0x00593800 0x0035e400 0x003cac00 0x003a4000 0x007b2400 0x0024e000 0x001aac00

PRE LBA LIST:
LBA[ 0]: 0x0009e000 0x001d6800 0x006d2c00 0x00780800 0x00345800 0x00085000 0x002e2c00 0x00437000 0x005fc400 0x0014c800

--------------------------------------------------------------------------------------------------------------

DETECT READ ERROR:
LBA: 0x0 | HF: 0x0400ffff F1: 0x523f39d9 F2: 0xf71538a0 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 255]
DETECT ERROR IO Time: 2023-07-09 23:33:05

Buffer addr: 0xffff9aa00000 | Physical addr: 0x181aca30000

LBA LIST:
LBA[ 0]: 0x00000000 0x006c6c00 0x002a6000 0x00565000 0x0049bc00 0x00659000 0x007fe000 0x004c0000 0x0062f800 0x0019b000
LBA[10]: 0x00198c00 0x004e8400 0x0003f800 0x00530000 0x00738000 0x00239800 0x007a5800 0x00080400 0x0049f000 0x001cb800
LBA[20]: 0x000cd400 0x0060b800 0x00242000 0x0018a800 0x00080000 0x0041fc00 0x00207000 0x00618000 0x005e6000 0x00101000
LBA[30]: 0x005a5400 0x00091400 0x002e9400 0x0044fc00 0x00050000 0x004f9c00 0x0049a800 0x00320000 0x007ef400 0x00329000
LBA[40]: 0x004bc000 0x006c9000 0x005fd000 0x00497000 0x00142000 0x005f1800 0x00320800 0x001b5800 0x0043e800 0x0076f800
LBA[50]: 0x00675800 0x00739c00 0x00245000 0x00378400 0x00198000 0x0009b800 0x002f4c00 0x005a3400 0x0064fc00 0x00150400
LBA[60]: 0x002e4000 0x007e4000 0x00036400 0x005ff000 0x001f3800 0x00410c00 0x00329800 0x00450400 0x003e8400 0x0076d400
LBA[70]: 0x004c1000 0x003ed000 0x0047f000 0x00279c00 0x0064e800 0x0001c000 0x003bbc00 0x004a2000 0x007c4800 0x0011d000
LBA[80]: 0x00182000 0x00036800 0x000b0000 0x0064f000 0x005f7000 0x00367000 0x004fc000 0x0022ac00 0x0014b400 0x001cb400
LBA[90]: 0x0028e800 0x0010a000 0x006bd800 0x0060a000 0x0057a400 0x00530c00 0x0067a000 0x0051f800 0x00035800 0x00768400

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

--------------------------------------------------------------------------------------------------------------


Current Time: 2023-07-09 23:36:12

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 001[2]：案例一，inject LIST lba bug

随机校验数据，簇间list数据不一致

BUG 001[2]: [Thread: 3] RANDOM VERIFY DIFFER LBA_LIST
| ERROR LBA: 0x28c00 | filename: lba.raw
LBA: 0x28c00 | thread_num: 10, bitmap: 0 0 0 1 0 0 0 0 0 0

CORRECT LBA: 0x1c3480 | HF: 0x00801273 F1: 0x9c38fbd9 F2: 0x16672d2a [LOOP: 1 WRITE: 138 SECTOR: 0 / NUM: 128 / ZERO: 2]
CORRECT LBA IO Time: 2023-07-04 01:13:15

ERROR LBA: 0x28c00 | HF: 0x00801273 F1: 0x9c38fbd9 F2: 0x16672d2a [LOOP: 1 WRITE: 139 SECTOR: 0 / NUM: 128 / ZERO: 2]
ERROR LBA IO Time: 2023-07-04 01:13:16

---runtime校验

```
BUG 003: [Thread: 3] WRITE PART SECTOR | read[139] | filename: lba.raw

CORRECT LBA: 0x1c3480 | HF: 0x00801273 F1: 0x9c38fbd9 F2: 0x16672d2a [LOOP:
CORRECT LBA IO Time: 2023-07-04 01:13:15

ERROR LBA: 0x28c00 | HF: 0x00801273 F1: 0x9c38fbd9 F2: 0x16672d2a [LOOP: 1 W
ERROR LBA IO Time: 2023-07-04 01:13:16
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-04 01:13:06
8, 14 | 9, 15 | 8, 14 | 8, 15 | 8, 14 | 7, 15 | 6, 15 | 8, 14 | 8, 14 | 8, 15 |

Current Time: 2023-07-04 01:13:20
BUG 001[2]: [Thread: 3] RANDOM VERIFY DIFFER LBA LIST | ERROR LBA: 0x28c00 | filename: lba.raw
LBA: 0x28c00 | thread_num: 10, bitmap: 0 0 0 1 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x156280 | HF: 0x00801273 F1: 0x9c38fbd9 F2: 0x16672d2a [LOOP: 1 WRITE: 193 SECTOR: 0 / NUM: 128 / ZERO: 2]
LAST WRITE IO Time: 2023-07-04 01:13:19

Buffer addr: 0xffff8c030000 | Physical addr: 0x383dfe30000

LBA LIST:
LBA[ 0]: 0x00156280 0x0007a800 0x0001e900 0x00123d00 0x000d3d80 0x001a9a00 0x001cd900 0x00077800 0x001fac80 0x001bd280
LBA[10]: 0x00103800 0x000ef900 0x00082000 0x001c0080 0x000d1580 0x0007c600 0x001b0f00 0x0000b400 0x00009100 0x00160900
LBA[20]: 0x00007000 0x0012c380 0x000e8780 0x0009b580 0x00173a80 0x0009a100 0x00080f00 0x0006ea80 0x001d1400 0x00042780
LBA[30]: 0x000ca600 0x00081100 0x001a2300 0x0012b980 0x00061e80 0x000af080 0x00123800 0x00096f80 0x000bdc00 0x001c3d00
LBA[40]: 0x00152500 0x0013a100 0x000f9c00 0x0007fd80 0x00102000 0x0006f300 0x00170900 0x001b9580 0x00084f00 0x0014c400
LBA[50]: 0x001f0780 0x0001f400 0x00196c00 0x001b5e00 0x00075a80 0x00193a00 0x0011c600 0x0018b880 0x0001lc00 0x000fd700
LBA[60]: 0x00090280 0x0014ba00 0x00045a00 0x0002a680 0x0019d200 0x00060400 0x000d3780 0x00033980 0x000d4c00 0x00194400
LBA[70]: 0x000a5600 0x00163400 0x00064300 0x00121b00 0x000fbc00 0x00130700 0x00046080 0x0005b900 0x00130d80 0x00013800
LBA[80]: 0x0001bc80 0x001e3980 0x0006ee00 0x00107480 0x000a9600 0x000ae180 0x00178f80 0x001c0180 0x001e1000 0x0005ed00
LBA[90]: 0x00104680 0x0015c400 0x0012e000 0x000b3b80 0x0015cf80 0x000ba080 0x00072880 0x001ce080 0x001cde00 0x0004e000

PRE LBA LIST:
LBA[ 0]: 0x0006ae00 0x000f2680 0x00197080 0x000e7400 0x00020900 0x00084700 0x0014c200 0x001a7200 0x001c1580 0x000ae700

DETECT READ ERROR:
LBA: 0x28c00 | HF: 0x00801273 F1: 0x9c38fbd9 F2: 0x16672d2a [LOOP: 1 WRITE: 139 SECTOR: 0 / NUM: 128 / ZERO: 2]
DETECT ERROR IO Time: 2023-07-04 01:13:16

Buffer addr: 0xffff9c300000 | Physical addr: 0x383774a0000

LBA LIST:
LBA[ 0]: 0x00028c00 0x0009da80 0x000d0700 0x00135b00 0x00008000 0x00141480 0x000d9000 0x001ff300 0x00033d80 0x000bc300
LBA[10]: 0x001ed600 0x00077580 0x0007d500 0x0010ac80 0x000ad400 0x001a0700 0x00167b80 0x00172b80 0x00055d00 0x0003f980
LBA[20]: 0x001c9080 0x00054180 0x000bc780 0x000f2e00 0x00111700 0x00118d00 0x0003e800 0x00087000 0x00129d00 0x000f1b80
LBA[30]: 0x00087280 0x00020400 0x00107600 0x001db000 0x001bef80 0x00085480 0x0015d980 0x000e7680 0x00187b00 0x001f3800
LBA[40]: 0x00130000 0x000e8680 0x001a8e80 0x001f2600 0x000ae700 0x001c1580 0x001a7200 0x0014c200 0x00084700 0x00020900
LBA[50]: 0x000e7400 0x00197080 0x000f2680 0x0006ae00 0xce462e3e 0x0007a800 0x0001e900 0x00123d00 0x000d3d80 0x001a9a00
LBA[60]: 0x001cd900 0x00077800 0x001fac80 0x001bd280 0x00103800 0x000ef900 0x00082000 0x001c0080 0x000d1580 0x0007c600
LBA[70]: 0x001b0f00 0x0000b400 0x00009100 0x00160900 0x00007000 0x0012c380 0x000e8780 0x0009b580 0x00173a80 0x0009a100
LBA[80]: 0x00080f00 0x0006ea80 0x001d1400 0x00042780 0x000ca600 0x00081100 0x001a2300 0x0012b980 0x00061e80 0x000af080
LBA[90]: 0x00123800 0x00096f80 0x000bdc00 0x001c3d00 0x00152500 0x0013a100 0x000f9c00 0x0007fd80 0x00102000 0x0006f300

PRE LBA LIST:
LBA[ 0]: 0x001c3480 0x001b3400 0x00050800 0x00032500 0x00199500 0x0014f100 0x0007b580 0x00138600 0x0006bf00 0x00043500

Current Time: 2023-07-04 01:13:20

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 001[2]：案例二，inject LIST lba bug

随机校验数据，簇间list数据不一致

BUG 001[2]: [Thread: 7] RANDOM VERIFY DIFFER LBA_LIST
| ERROR LBA: 0x380 | filename: lba.raw
LBA: 0x380 | thread_num: 10, bitmap: 0 0 0 0 0 0 0 1 0 0

CORRECT LBA[0]: 0x380 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 1 SECTOR: 0 / NUM: 128 / ZERO: 14]
CORRECT IO Time: 2023-07-04 01:13:00

ERROR LBA[1]: 0x381 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 1 SECTOR: 1 / NUM: 128 / ZERO: 14]
INCONSISTENT IO Time: 2023-07-04 01:13:00

NEXT LBA: 0xe0000 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 2 SECTOR: 0 / NUM: 128 / ZERO: 14]
NEXT IO Time: 2023-07-04 01:13:00
---runtime校验

```
BUG 007[3]: [Thread: 7] VERIFY SECTOR[1] DIFFER: 113 | read[2] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x00000000 disk/file
hd_write_verify_dump -c -D -L 0x000e0000 disk/file

CORRECT LBA[0]: 0x380 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 1 SECTOR: 0 / NUM: 128 /
CORRECT IO Time: 2023-07-04 01:13:00

ERROR LBA[1]: 0x381 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 1 SECTOR: 1 / NUM: 128 / ZE
INCONSISTENT IO Time: 2023-07-04 01:13:00

NEXT LBA: 0xe0000 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 2 SECTOR: 0 / NUM: 128 / ZERO
NEXT IO Time: 2023-07-04 01:13:00
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 5: .....
Current Time: 2023-07-04 01:12:44
58, 65 | 59, 67 | 58, 65 | 57, 64 | 58, 64 | 58, 65 | 58, 65 | 60, 66 | 60, 66 | 55, 61 |

Current Time: 2023-07-04 01:13:00

Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 6: ......
Current Time: 2023-07-04 01:13:00

Current Time: 2023-07-04 01:13:00
BUG 001[2]: [Thread: 7] RANDOM VERIFY DIFFER LBA LIST | ERROR LBA: 0x380 | filename: lba.raw
LBA: 0x380 | thread_num: 10, bitmap: 0 0 0 0 0 0 0 1 0 0

LAST CORRECT WRITE:
LBA: 0x5fa80 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 30 SECTOR: 0 / NUM: 128 / ZERO: 14]
LAST WRITE IO Time: 2023-07-04 01:13:00

Buffer addr: 0xfffd04030000 | Physical addr: 0x23503c0000

LBA LIST:
LBA[ 0]: 0x0005fa80 0x0004c000 0x00123c00 0x00080000 0x001bb000 0x00008200 0x0011ec00 0x000dc800 0x00148700 0x00126500
LBA[10]: 0x00084000 0x0000fe00 0x00146200 0x00089000 0x001ad500 0x00069e00 0x00126000 0x0003f400 0x00178200 0x001de900
LBA[20]: 0x0011fe00 0x00189080 0x0002a880 0x001b7400 0x000cea00 0x000b4180 0x00016880 0x00118b80 0x000e2000 0x001c8680
LBA[30]: 0x00061e00 0x00021f00 0x00179980 0x00034e80 0x0012ff00 0x0008c680 0x001cfc00 0x00056080 0x0006c300 0x001d6280
LBA[40]: 0x001d9380 0x00167000 0x0008c400 0x00191400 0x001b0800 0x001d1100 0x001e6d00 0x00192c80 0x00181480 0x001fb800
LBA[50]: 0x0019c200 0x000b4200 0x000a0280 0x00014800 0x000c9600 0x0016a000 0x001f7800 0x0015a100 0x00050280 0x00164000
LBA[60]: 0x0012e780 0x0000e900 0x0016e600 0x0001d400 0x0010ee00 0x00140100 0x00021e00 0x00006100 0x00152680 0x00110000
LBA[70]: 0x000e0400 0x0014d180 0x00026500 0x001e5400 0x000cc700 0x0010d800 0x00115180 0x0014da00 0x00021000 0x001d2e80
LBA[80]: 0x000b9b80 0x00086380 0x00181400 0x00066280 0x00058a80 0x00005700 0x00090a80 0x0019db80 0x001d3a00 0x0007e080
LBA[90]: 0x00092100 0x00045680 0x00144300 0x0007f680 0x000da300 0x001fbc00 0x00186d00 0x0018b680 0x001bad80 0x000be280

PRE LBA LIST:
LBA[ 0]: 0x00007e00 0x0017c000 0x0007a000 0x00035000 0x000f3400 0x00090d80 0x000e2200 0x000f3b80 0x00031480 0x001db400

DETECT READ ERROR:
LBA: 0x380 | HF: 0x00801277 F1: 0x87ae66e6 F2: 0x2cdf9a00 [LOOP: 6 WRITE: 1 SECTOR: 0 / NUM: 128 / ZERO: 14]
DETECT ERROR IO Time: 2023-07-04 01:13:00

Buffer addr: 0xfffeeea30000 | Physical addr: 0x24b3ba0000

LBA LIST:
LBA[ 0]: 0x00000380 0x000e0000 0x000d7900 0x00064880 0x001c2600 0x0002fa00 0x0007f880 0x00042200 0x0010ed80 0x0015ec00
LBA[10]: 0x001b1800 0x0010a980 0x001c3e00 0x00111800 0x0012ad80 0x0002c900 0x001e0880 0x00071800 0x001c6000 0x001db400
LBA[20]: 0x00031480 0x000f3b80 0x000e2200 0x00090d80 0x000f3400 0x00035000 0x0007a000 0x0017c000 0x00007e00 0x0005fa80
LBA[30]: 0x0004c000 0x00123c00 0x00080000 0x001bb000 0x00008200 0x0011ec00 0x000dc800 0x00148700 0x00126500 0x00084000
LBA[40]: 0x0000fe00 0x00146200 0x00089000 0x001ad500 0x00069e00 0x00126000 0x0003f400 0x00178200 0x001de900 0x0011fe00
LBA[50]: 0x00189080 0x0002a880 0x001b7400 0x000cea00 0x000b4180 0x00016880 0x00118b80 0x000e2000 0x001c8680 0x00061e00
LBA[60]: 0x00021f00 0x00179980 0x00034e80 0x0012ff00 0x0008c680 0x001cfc00 0x00056080 0x0006c300 0x001d6280 0x001d9380
LBA[70]: 0x00167000 0x0008c400 0x00191400 0x001b0800 0x001d1100 0x001e6d00 0xefec3940 0x00181480 0x001fb800 0x0019c200
LBA[80]: 0x000b4200 0x000a0280 0x00014800 0x000c9600 0x0016a000 0x001f7800 0x0015a100 0x00050280 0x00164000 0x0012e780
LBA[90]: 0x0000e900 0x0016e600 0x0001d400 0x0010ee00 0x00140100 0x00021e00 0x00006100 0x00152680 0x00110000 0x000e0400

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Current Time: 2023-07-04 01:13:00

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 001[3]：批量校验数据，簇间flag数据不一致

BUG 001[3]: [Thread: 1] BATCH VERIFY DIFFER FLAGS
| ERROR LBA: 0x4f5000
| filename: lba.raw

PREV LBA: 0x24cc00 | HF: 0x04001271 F1: 0xda8a6f04 F2: 0x51043500 [LOOP: 1 WRITE: 404 SECTOR: 0 / NUM: 1024 / ZERO: 106]
PREV IO Time: 2023-06-25 20:11:47

LOST LBA: 0x4f5000 | HF: 0xb0b0b0b0 F1: 0xb0b0b0b0 F2: 0xb0b0b0b0 [LOOP: 2964369584 WRITE: 2964369584 SECTOR: 45232 / NUM: 45232 / ZERO: 2964369584]
LOST IO Time: -1330595812--1330597711--1330597712 -1330597712:-1330597712:-1330597712

NEXT LBA: 0x2c6400 | HF: 0x04001271 F1: 0xda8a6f04 F2: 0x51043500 [LOOP: 1 WRITE: 406 SECTOR: 0 / NUM: 1024 / ZERO: 106]
NEXT IO Time: 2023-06-25 20:11:48

---runtime校验

```
BUG 004: [Thread: 1] DATA LOST | read[406] | filename: lba.raw
PREV LBA: 0x24cc00 | LOST LBA: 0x4f5000 | NEXT LBA: 0x2c6400

PREV LBA: 0x24cc00 | HF: 0x04001271 F1: 0xda8a6f04 F2: 0x51043500 [LOOP: 1 WRITE: 404 SECTOR: 0 / NUM: 1024 / Z
PREV IO Time: 2023-06-25 20:11:47

LOST LBA: 0x4f5000 | HF: 0xb0b0b0b0 F1: 0xb0b0b0b0 F2: 0xb0b0b0b0 [LOOP: 2964369584 WRITE: 2964369584 SECTOR: 4
LOST IO Time: -1330595812--1330597711--1330597712 -1330597712:-1330597712:-1330597712

NEXT LBA: 0x2c6400 | HF: 0x04001271 F1: 0xda8a6f04 F2: 0x51043500 [LOOP: 1 WRITE: 406 SECTOR: 0 / NUM: 1024 / Z
NEXT IO Time: 2023-06-25 20:11:48
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-25 20:10:48
270, 230 | 292, 245 | 273, 232 | 260, 245 | 276, 234 | 270, 249 | 283, 239 | 274, 253 | 277, 243 | 270, 237 |

Current Time: 2023-06-25 20:11:49
BUG 001[3]: [Thread: 1] BATCH VERIFY DIFFER FLAGS | ERROR LBA: 0x4f5000 | filename: lba.raw
LBA: 0x4f5000 | thread_num: 10, bitmap: 0 1 0 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x2c6400 | HF: 0x04001271 F1: 0xda8a6f04 F2: 0x51043500 [LOOP: 1 WRITE: 406 SECTOR: 0 / NUM: 1024 / ZERO: 106]
LAST WRITE IO Time: 2023-06-25 20:11:48

Buffer addr: 0xffff98090000 | Physical addr: 0x58218d50000

LBA LIST:
LBA[ 0]: 0x002c6400 0x001d0800 0x0054ec00 0x00661400 0x00380800 0x00464800 0x0065ec00 0x005da800 0x00241c00 0x003bc800
LBA[10]: 0x006b6000 0x00105c00 0x0020e400 0x003b1400 0x0027cc00 0x000b7c00 0x005d4000 0x00301800 0x001e6800 0x004b8800
LBA[20]: 0x003c4400 0x00029000 0x0025cc00 0x00057000 0x00531800 0x002fcc00 0x00631800 0x005b5000 0x0050a000 0x004a8000
LBA[30]: 0x002af000 0x004f1c00 0x00581c00 0x00006000 0x0037b400 0x0008fc00 0x003cf800 0x007a5400 0x003db400 0x00799400
LBA[40]: 0x002bf400 0x003e7c00 0x000e7800 0x005ea800 0x0013ec00 0x003c6400 0x00070000 0x00130000 0x002cac00 0x001ad400
LBA[50]: 0x00446800 0x00787000 0x0072ac00 0x002a6800 0x002c2800 0x00231800 0x002c7c00 0x007a0c00 0x007bb000 0x007a8400
LBA[60]: 0x00452400 0x0054d000 0x00389000 0x002c5800 0x002ca000 0x00556c00 0x00081800 0x007ac800 0x0046e800 0x00166c00
LBA[70]: 0x000b0800 0x00600c00 0x002d6800 0x00131c00 0x001a9c00 0x007ed000 0x001f1c00 0x0046a800 0x0008b000 0x00125800
LBA[80]: 0x00723800 0x00574c00 0x00753800 0x006a3400 0x004a9400 0x00283c00 0x00484c00 0x006d9400 0x0024a400 0x00106000
LBA[90]: 0x000e8400 0x00234400 0x00490400 0x0055f000 0x00758000 0x00512c00 0x002c2c00 0x000c2800 0x007be400 0x00366400

PRE LBA LIST:
LBA[ 0]: 0x004f5000 0x0024cc00 0x007f8800 0x003fe000 0x0002c400 0x0033f800 0x007a4c00 0x007aac00 0x001a5c00 0x003f6000

DETECT READ ERROR:
LBA: 0x4f5000 | HF: 0xb0b0b0b0 F1: 0xb0b0b0b0 F2: 0xb0b0b0b0 [LOOP: 2964369584 WRITE: 2964369584 SECTOR: 45232 / NUM: 45232 / ZERO: 2964369584]
DETECT ERROR IO Time: -1330595812--1330597711--1330597712 -1330597712:-1330597712:-1330597712

Buffer addr: 0xffff6fd90000 | Physical addr: 0x50298440000

LBA LIST:
LBA[ 0]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[10]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[20]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[30]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[40]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[50]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[60]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[70]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[80]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0
LBA[90]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0

PRE LBA LIST:
LBA[ 0]: 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0 0xb0b0b0b0

Current Time: 2023-06-25 20:11:49

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 001[4]：批量校验数据，簇间list数据不一致

BUG 001[4]: [Thread: 9] BATCH VERIFY DIFFER LBA_LIST | ERROR LBA: 0x404800 | filename: lba.raw
LBA: 0x404800 | thread_num: 10, bitmap: 0 0 0 0 0 0 0 0 0 1

CORRECT LBA: 0x454400 | HF: 0x04001279 F1: 0x47877c66 F2: 0x92e7f068 [LOOP: 1 WRITE: 406 SECTOR: 0 / NUM: 1024 / ZERO: 176]
CORRECT LBA IO Time: 2023-06-26 00:22:20

ERROR LBA: 0x404800 | HF: 0x04001279 F1: 0x47877c66 F2: 0x92e7f068 [LOOP: 1 WRITE: 407 SECTOR: 0 / NUM: 1024 / ZERO: 176]
ERROR LBA IO Time: 2023-06-26 00:22:21

---runtime校验

```
BUG 003: [Thread: 9] WRITE PART SECTOR | read[407] | filename: lba.raw

CORRECT LBA: 0x454400 | HF: 0x04001279 F1: 0x47877c66 F2: 0x92e7f068 [LOOP: 1 WRITE: 406 SECT(
CORRECT LBA IO Time: 2023-06-26 00:22:20

ERROR LBA: 0x404800 | HF: 0x04001279 F1: 0x47877c66 F2: 0x92e7f068 [LOOP: 1 WRITE: 407 SECTOR:
ERROR LBA IO Time: 2023-06-26 00:22:21
```

---checktime校验

```
362, 292 | 373, 301 | 365, 294 | 355, 298 | 365, 294 | 351, 292 | 361, 308 | 357, 293 | 354, 292 | 360, 303 |

Current Time: 2023-06-26 00:22:23
BUG 001[4]: [Thread: 9] BATCH VERIFY DIFFER LBA_LIST | ERROR LBA: 0x404800 | filename: lba.raw
LBA: 0x404800 | thread_num: 10, bitmap: 0 0 0 0 0 0 0 0 0 1

LAST CORRECT WRITE:
LBA: 0x17b000 | HF: 0x04001279 F1: 0x47877c66 F2: 0x92e7f068 [LOOP: 1 WRITE: 416 SECTOR: 0 / NUM: 1024 / ZERO: 176]
LAST WRITE IO Time: 2023-06-26 00:22:21

Buffer addr: 0xffff756e0000 | Physical addr: 0x3811c9f0000

LBA LIST:
LBA[ 0]: 0x0017b000 0x003eac00 0x00021800 0x003cec00 0x0006fc00 0x00697800 0x007c8800 0x002cc000 0x004c6c00 0x0077f400
LBA[10]: 0x00639800 0x0022c400 0x002ea400 0x000fe000 0x001be000 0x000a0400 0x000a4c00 0x00124000 0x0014f800 0x0054e000
LBA[20]: 0x00260800 0x007b9000 0x00301400 0x006a1c00 0x006e3c00 0x00162000 0x002be800 0x00199400 0x003ca000 0x004e8c00
LBA[30]: 0x001a6c00 0x004c6400 0x00672000 0x00491c00 0x005de000 0x00331800 0x00625400 0x00251800 0x0055a400 0x0010e000
LBA[40]: 0x00336800 0x0020ac00 0x004b1000 0x003c5c00 0x000db000 0x005aec00 0x006cdc00 0x007f0800 0x001a1000 0x004c9c00
LBA[50]: 0x0021e400 0x006ecc00 0x005b5400 0x004f7800 0x005c8800 0x001f9800 0x00492400 0x001e4400 0x003d6c00 0x00038c00
LBA[60]: 0x002d0400 0x006b3800 0x00263400 0x004d2c00 0x005b9400 0x007a1c00 0x0068a400 0x00033800 0x003e2400 0x00231400
LBA[70]: 0x00353800 0x006e5000 0x00528800 0x007b7800 0x00421800 0x00678400 0x00413800 0x001f9c00 0x00225000 0x007f6400
LBA[80]: 0x005f5400 0x002ccc00 0x0054c400 0x00733400 0x00728400 0x00556000 0x0022e800 0x0035a800 0x00570c00 0x007db400
LBA[90]: 0x00160800 0x007b9400 0x00772c00 0x005db000 0x0049bc00 0x005d5800 0x0031e800 0x006e5800 0x007bbc00 0x000d5c00

PRE LBA LIST:
LBA[ 0]: 0x005eb000 0x00176400 0x0008b000 0x00152c00 0x0014e800 0x003efc00 0x00054000 0x006ca400 0x00404800 0x00454400

DETECT READ ERROR:
LBA: 0x404800 | HF: 0x04001279 F1: 0x47877c66 F2: 0x92e7f068 [LOOP: 1 WRITE: 407 SECTOR: 0 / NUM: 1024 / ZERO: 176]
DETECT ERROR IO Time: 2023-06-26 00:22:21

Buffer addr: 0xffff74ba0000 | Physical addr: 0x300dd980000

LBA LIST:
LBA[ 0]: 0x00404800 0x006ca400 0x00054000 0x003efc00 0x0014e800 0x00152c00 0x0008b000 0x00176400 0x005eb000 0x0017b000
LBA[10]: 0x003eac00 0x00021800 0x003cec00 0x0006fc00 0x00697800 0x007c8800 0x002cc000 0x004c6c00 0x0077f400 0x00639800
LBA[20]: 0x0022c400 0x002ea400 0x000fe000 0x001be000 0x000a0400 0x000a4c00 0x00124000 0x0014f800 0x0054e000 0x00260800
LBA[30]: 0x007b9000 0x00301400 0x0b38ffa0 0x006e3c00 0x00162000 0x002be800 0x00199400 0x003ca000 0x004e8c00 0x001a6c00
LBA[40]: 0x004c6400 0x00672000 0x00491c00 0x005de000 0x00331800 0x00625400 0x00251800 0x0055a400 0x0010e000 0x00336800
LBA[50]: 0x0020ac00 0x004b1000 0x003c5c00 0x000db000 0x005aec00 0x006cdc00 0x007f0800 0x001a1000 0x004c9c00 0x0021e400
LBA[60]: 0x006ecc00 0x005b5400 0x004f7800 0x005c8800 0x001f9800 0x00492400 0x001e4400 0x003d6c00 0x00038c00 0x002d0400
LBA[70]: 0x006b3800 0x00263400 0x004d2c00 0x005b9400 0x007a1c00 0x0068a400 0x00033800 0x003e2400 0x00231400 0x00353800
LBA[80]: 0x006e5000 0x00528800 0x007b7800 0x00421800 0x00678400 0x00413800 0x001f9c00 0x00225000 0x007f6400 0x005f5400
LBA[90]: 0x002ccc00 0x0054c400 0x00733400 0x00728400 0x00556000 0x0022e800 0x0035a800 0x00570c00 0x007db400 0x00160800

PRE LBA LIST:
LBA[ 0]: 0x00454400 0x00578000 0x00065c00 0x005f2400 0x0004e800 0x00317400 0x0070d400 0x0056b800 0x0012d000 0x0032e400

Current Time: 2023-06-26 00:22:23

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[1]：案例一，inject SECTOR lba bug

随机校验数据，簇内校验数据出错

BUG 002[1]: [Thread: 5] RANDOM VERIFY SECTOR[647] DIFFER: 1
| CORRECT LBA[0]: 0x63f400 | ERROR LBA[647]: 0x63f687 | filename: lba.raw
LBA: 0x63f687 | thread_num: 10, bitmap: 0 0 0 0 0 1 0 0 0 0

CORRECT LBA[0]: 0x13fe000 | HF: 0x04001276 F1: 0x59f10fb8 F2: 0xe0d6306e
[LOOP: 1 WRITE: 542 SECTOR: 0 / NUM: 1024 / ZERO: 14]
CORRECT IO Time: 2023-06-27 01:46:46

ERROR LBA[112]: 0x13fe070 | HF: 0xd9d9d9d9 F1: 0xd9d9d9d9 F2:
0xd9d9d9d9 [LOOP: 3654932953 WRITE: 3654932953 SECTOR: 55769 / NUM:
55769 / ZERO: 3654932953]
INCONSISTENT IO Time: -640032443--640034342--640034343--640034343

CORRECT LBA[125]: 0x13fe07d | HF: 0x04001276 F1: 0x59f10fb8 F2:
0xe0d6306e [LOOP: 1 WRITE: 542 SECTOR: 125 / NUM: 1024 / ZERO: 14]
CORRECT IO Time: 2023-06-27 01:46:46

---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 6] VERIFY SECTOR[112] DIFFER: 13 | read[542] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x01341000 disk/file
hd_write_verify_dump -c -D -L 0x01248000 disk/file

CORRECT LBA[0]: 0x13fe000 | HF: 0x04001276 F1: 0x59f10fb8 F2: 0xe0d6306e [LOOP: 1 WRITE: 542 SECTOR: 0 / NUM:
CORRECT IO Time: 2023-06-27 01:46:46

ERROR LBA[112]: 0x13fe070 | HF: 0xd9d9d9d9 F1: 0xd9d9d9d9 F2: 0xd9d9d9d9 [LOOP: 3654932953 WRITE: 3654932953 S
INCONSISTENT IO Time: -640032443--640034342--640034343 -640034343:-640034343:-640034343

CORRECT LBA[125]: 0x13fe07d | HF: 0x04001276 F1: 0x59f10fb8 F2: 0xe0d6306e [LOOP: 1 WRITE: 542 SECTOR: 125 / N
CORRECT IO Time: 2023-06-27 01:46:46
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-27 01:45:46
479, 374 | 470, 375 | 492, 391 | 505, 388 | 503, 387 | 485, 394 | 498, 391 | 487, 384 | 513, 394 | 488, 378 |

Current Time: 2023-06-27 01:47:06
BUG 002[1]: [Thread: 6] RANDOM VERIFY SECTOR[112] DIFFER: 13 | CORRECT LBA[0]: 0x13fe000 | ERROR LBA[112]: 0x13fe070 | filename: lba.raw
LBA: 0x13fe070 | thread_num: 10, bitmap: 0 0 0 0 0 0 1 0 0 0

LAST CORRECT WRITE:
LBA: 0xd21c00 | HF: 0x04001276 F1: 0x59f10fb8 F2: 0xe0d6306e [LOOP: 1 WRITE: 738 SECTOR: 0 / NUM: 1024 / ZERO: 14]
LAST WRITE IO Time: 2023-06-27 01:47:06

Buffer addr: 0xffff89c00000 | Physical addr: 0x2026ff30000

LBA LIST:
LBA[ 0]: 0x00d21c00 0x00c50c00 0x00bdc400 0x009f2800 0x006eb400 0x00bf1400 0x00e5f800 0x00abd000 0x01399400 0x00eea400
LBA[10]: 0x00ae5c00 0x008ce000 0x00f74400 0x00040800 0x00534400 0x01245c00 0x0055b400 0x0014b000 0x00c4a800 0x00276000
LBA[20]: 0x0066ec00 0x00ac2c00 0x0090c400 0x00a36c00 0x003cf800 0x007b1000 0x00acb000 0x00ccfc00 0x00a85800 0x007a1400
LBA[30]: 0x00416c00 0x0083b400 0x00178000 0x003a6400 0x0068d800 0x01160400 0x011d3800 0x010ddc00 0x009e7400 0x0081cc00
LBA[40]: 0x00685000 0x0112ac00 0x00820c00 0x012db800 0x0129cc00 0x012cd400 0x006c7c00 0x01116000 0x00caf400 0x003d5800
LBA[50]: 0x00bd9000 0x005cd000 0x013a7400 0x00f70000 0x00958400 0x007f7000 0x01222400 0x00c7e800 0x00ff7800 0x00315c00
LBA[60]: 0x00192000 0x0011bc00 0x00274800 0x00114000 0x00004c00 0x00b3a000 0x0090c800 0x001d8400 0x01353400 0x00795000
LBA[70]: 0x000be000 0x011e8000 0x00c80800 0x00719000 0x012e7000 0x00d46800 0x009f8c00 0x012e3400 0x01233800 0x00e6d800
LBA[80]: 0x00226800 0x0101d400 0x01301400 0x010cac00 0x00f2dc00 0x0120a400 0x008d2400 0x00988000 0x00231800 0x008a5000
LBA[90]: 0x00140800 0x0053e400 0x00c2c800 0x000bcc00 0x00d2d000 0x0088b000 0x01180c00 0x012ee800 0x00944400 0x0061bc00

PRE LBA LIST:
LBA[ 0]: 0x0037b800 0x00b9c800 0x010a9c00 0x005ef800 0x00f15c00 0x00b6d000 0x003c4000 0x0114f400 0x00cc2800 0x00c30c00

RANDOM VERIFY:
LBA: 0x13fe000 | HF: 0x04001276 F1: 0x59f10fb8 F2: 0xe0d6306e [LOOP: 1 WRITE: 542 SECTOR: 0 / NUM: 1024 / ZERO: 14]
RANDOM VERIFY IO Time: 2023-06-27 01:46:46

Buffer addr: 0xffff89200000 | Physical addr: 0x2026da50000

LBA LIST:
LBA[ 0]: 0x013fe000 0x01248000 0x012e0400 0x00db7000 0x002c9400 0x00a7d800 0x002bd400 0x00d61400 0x00058c00 0x00485800
LBA[10]: 0x00760400 0x0016f800 0x00b89800 0x008ad400 0x01316000 0x0012e800 0x0023ec00 0x00404800 0x00008c00 0x0040f400
LBA[20]: 0x01260800 0x00a6fc00 0x01005800 0x00447800 0x00255c00 0x00f31c00 0x0038b400 0x003d4000 0x009bcc00 0x00401c00
LBA[30]: 0x00b8c000 0x00ec6800 0x008c3c00 0x00731400 0x008d3000 0x011ae000 0x00974400 0x011ab800 0x00b04800 0x009a8800
LBA[40]: 0x00e49c00 0x0121f000 0x0096cc00 0x000e4000 0x001e9000 0x00a15000 0x00a4fc00 0x00345000 0x00651400 0x0093f800
LBA[50]: 0x002ba800 0x0116c400 0x00c1fc00 0x00317000 0x00a97400 0x0060c000 0x008a2000 0x000e9800 0x01140800 0x00786c00
LBA[60]: 0x002c2400 0x00a76400 0x00d4bc00 0x013e8000 0x012d2000 0x00494c00 0x00f0b400 0x00c7c400 0x005dc400 0x0109dc00
LBA[70]: 0x001d1000 0x0005cc00 0x006a1400 0x007f9400 0x01202800 0x0107f400 0x00399c00 0x0024c000 0x002fdc00 0x01244800
LBA[80]: 0x0001e400 0x003a4000 0x00c65800 0x00626800 0x0099b800 0x003a8000 0x00e83000 0x00422400 0x00818400 0x00c1c400
LBA[90]: 0x0092c800 0x001bf000 0x008a6000 0x013f4000 0x007fa000 0x00db5800 0x00058800 0x00731800 0x00530400 0x00e43c00

PRE LBA LIST:
LBA[ 0]: 0x01341000 0x005ab000 0x012a0400 0x009dc000 0x006e3800 0x00f26000 0x00b98800 0x00396800 0x007da000 0x00227400

DETECT READ ERROR:
LBA: 0x13fe070 | HF: 0xd9d9d9d9 F1: 0xd9d9d9d9 F2: 0xd9d9d9d9 [LOOP: 3654932953 WRITE: 3654932953 SECTOR: 55769 / NUM: 55769 / ZERO: 3654932953]
DETECT ERROR IO Time: -640032443--640034342--640034343 -640034343:-640034343:-640034343

Buffer addr: 0xffff8920e000 | Physical addr: 0x2026da5e000

LBA LIST:
LBA[ 0]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[10]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[20]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[30]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[40]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[50]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[60]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[70]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[80]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9
LBA[90]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9

PRE LBA LIST:
LBA[ 0]: 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9 0xd9d9d9d9

Current Time: 2023-06-27 01:47:06

pause and exit work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[1]：案例二，inject LIST lba bug

随机校验数据，簇内校验数据出错

BUG 002[1]: [Thread: 5] RANDOM VERIFY SECTOR[647] DIFFER: 1
| CORRECT LBA[0]: 0x63f400 | ERROR LBA[647]: 0x63f687 | filename: lba.raw
LBA: 0x63f687 | thread_num: 10, bitmap: 0 0 0 0 0 1 0 0 0 0

CORRECT LBA[0]: 0x63f400 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1
[LOOP: 1 WRITE: 1353 SECTOR: 0 / NUM: 1024 / ZERO: 181]
CORRECT IO Time: 2023-06-27 02:09:39

ERROR LBA[647]: 0x63f687 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1
[LOOP: 1 WRITE: 1353 SECTOR: 647 / NUM: 1024 / ZERO: 181]
INCONSISTENT IO Time: 2023-06-27 02:09:39

CORRECT LBA[648]: 0x63f688 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1
[LOOP: 1 WRITE: 1353 SECTOR: 648 / NUM: 1024 / ZERO: 181]
CORRECT IO Time: 2023-06-27 02:09:39

---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 5] VERIFY SECTOR[647] DIFFER: 1 | read[1353] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x011cfc00 disk/file
hd_write_verify_dump -c -D -L 0x00cec400 disk/file

CORRECT LBA[0]: 0x63f400 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1 [LOOP: 1 WRITE: 1353 SECTOR: 0 / NUM: 1024 /
CORRECT IO Time: 2023-06-27 02:09:39

ERROR LBA[647]: 0x63f687 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1 [LOOP: 1 WRITE: 1353 SECTOR: 647 / NUM: 1024
INCONSISTENT IO Time: 2023-06-27 02:09:39

CORRECT LBA[648]: 0x63f688 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1 [LOOP: 1 WRITE: 1353 SECTOR: 648 / NUM: 102
CORRECT IO Time: 2023-06-27 02:09:39
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-27 02:08:40
1183, 843 | 1203, 855 | 1152, 843 | 1180, 842 | 1170, 840 | 1195, 850 | 1179, 849 | 1174, 844 | 1142, 815 | 1177, 839 |

Current Time: 2023-06-27 02:09:55
BUG 002[1]: [Thread: 5] RANDOM VERIFY SECTOR[647] DIFFER: 1 | CORRECT LBA[0]: 0x63f400 | ERROR LBA[647]: 0x63f687 | filename: lba.raw
LBA: 0x63f687 | thread_num: 10, bitmap: 0 0 0 0 0 1 0 0 0 0

LAST CORRECT WRITE:
LBA: 0xff4c00 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1 [LOOP: 1 WRITE: 1629 SECTOR: 0 / NUM: 1024 / ZERO: 181]
LAST WRITE IO Time: 2023-06-27 02:09:55

Buffer addr: 0xfffea4ee0000 | Physical addr: 0x42bd030000

LBA LIST:
LBA[ 0]: 0x00ff4c00 0x00cd5c00 0x00ad7400 0x0019d000 0x000d6c00 0x00d03c00 0x00908000 0x008df800 0x01239000 0x002b7400
LBA[10]: 0x00baa400 0x0082cc00 0x00b33c00 0x00eec800 0x0078e800 0x00e4dc00 0x00e4d400 0x0055f800 0x0135ec00 0x00989400
LBA[20]: 0x00ded400 0x006d7400 0x00f37400 0x00a05c00 0x003e5400 0x00c2a400 0x006bd000 0x00fe6c00 0x001eb800 0x006a6c00
LBA[30]: 0x0018e800 0x00faa400 0x00ce1c00 0x0045c000 0x01276c00 0x002b8000 0x00368c00 0x010d6400 0x00f55800 0x006f1000
LBA[40]: 0x006b7c00 0x0124dc00 0x0070c000 0x00318c00 0x0085dc00 0x0098a400 0x004b9c00 0x00337c00 0x00456400 0x010a9400
LBA[50]: 0x005af000 0x0048f400 0x00089400 0x001ec400 0x00c2b400 0x00fe3400 0x00396c00 0x0064c400 0x00b03800 0x0033bc00
LBA[60]: 0x002b8800 0x002dc400 0x00db3800 0x00db3c00 0x004d5800 0x00d5dc00 0x00d0cc00 0x01031400 0x0102d400 0x00818c00
LBA[70]: 0x00d7b400 0x00c09400 0x00d6e000 0x00253400 0x00e61400 0x00f8f000 0x00acb000 0x00848c00 0x00116400 0x007adc00
LBA[80]: 0x00e2e800 0x0056dc00 0x00b8c800 0x012ea800 0x0129ac00 0x00163800 0x01329400 0x01124400 0x00f95400 0x002ffc00
LBA[90]: 0x0003d400 0x0110f000 0x00a5c800 0x00fdfc00 0x00e2b800 0x002cc000 0x00c2bc00 0x009ef400 0x00981c00 0x005b3000

PRE LBA LIST:
LBA[ 0]: 0x0110c800 0x0015d000 0x003ff400 0x003b1c00 0x0112b400 0x012b8c00 0x005e2c00 0x006dc800 0x013ba400 0x007d2800

RANDOM VERIFY:
LBA: 0x63f400 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1 [LOOP: 1 WRITE: 1353 SECTOR: 0 / NUM: 1024 / ZERO: 181]
RANDOM VERIFY IO Time: 2023-06-27 02:09:39

Buffer addr: 0xfffea43a0000 | Physical addr: 0x435adb0000

LBA LIST:
LBA[ 0]: 0x0063f400 0x00cec400 0x00fb3000 0x011ed800 0x00dab000 0x00483c00 0x0028ac00 0x001b5800 0x00a12800 0x00183800
LBA[10]: 0x00668000 0x000b4c00 0x0131fc00 0x001ef000 0x006c7c00 0x00234c00 0x006b9800 0x008f9800 0x00057000 0x00a56800
LBA[20]: 0x00957c00 0x01057400 0x00af5000 0x0052fc00 0x00db5c00 0x0082ac00 0x00da5000 0x00b5c000 0x00953400 0x00d2c000
LBA[30]: 0x002ca400 0x00c5cc00 0x00415800 0x010cd400 0x00368400 0x0011bc00 0x00a73800 0x002e1800 0x006e9000 0x00c69c00
LBA[40]: 0x003c3000 0x00ec3400 0x005f3c00 0x001ad400 0x0012b800 0x01067800 0x00cddc00 0x00ab6800 0x008ea800 0x001b6400
LBA[50]: 0x00837c00 0x00316400 0x00666000 0x013f9000 0x0057cc00 0x011ee400 0x01055000 0x00f16c00 0x01333000 0x00cb7400
LBA[60]: 0x001dd400 0x00a21c00 0x00a66000 0x00c3dc00 0x0108e400 0x003eac00 0x00fabc00 0x008c1000 0x00c6c800 0x00147400
LBA[70]: 0x0028bc00 0x00d21400 0x00701400 0x011d0400 0x013f5000 0x0096b800 0x00febc00 0x01298800 0x009a5000 0x008d4400
LBA[80]: 0x01135c00 0x00e1e800 0x00924000 0x00b51c00 0x013a1400 0x003b2800 0x0096c800 0x005c1000 0x0036b400 0x0138ec00
LBA[90]: 0x012bf800 0x00bb1400 0x00186800 0x008cdc00 0x00b1a800 0x00d6f000 0x01018400 0x012a6c00 0x00a7cc00 0x00f1d400

PRE LBA LIST:
LBA[ 0]: 0x011cfc00 0x00970c00 0x0040a800 0x01078c00 0x01298c00 0x01069000 0x001fdc00 0x01389800 0x00cba800 0x00b62800

DETECT READ ERROR:
LBA: 0x63f687 | HF: 0x04001275 F1: 0x69f556d6 F2: 0xa270f2e1 [LOOP: 1 WRITE: 1353 SECTOR: 647 / NUM: 1024 / ZERO: 181]
DETECT ERROR IO Time: 2023-06-27 02:09:39

Buffer addr: 0xfffea43f0e00 | Physical addr: 0x443ac80e00

LBA LIST:
LBA[ 0]: 0x0063f400 0x00cec400 0x00fb3000 0x011ed800 0x00dab000 0x00483c00 0x0028ac00 0x001b5800 0x00a12800 0x00183800
LBA[10]: 0x00668000 0x000b4c00 0x0131fc00 0x001ef000 0x006c7c00 0x00234c00 0x006b9800 0x008f9800 0x00057000 0x00a56800
LBA[20]: 0x00957c00 0x01057400 0x00af5000 0x0052fc00 0x42b99f68 0x0082ac00 0x00da5000 0x00b5c000 0x00953400 0x00d2c000
LBA[30]: 0x002ca400 0x00c5cc00 0x00415800 0x010cd400 0x00368400 0x0011bc00 0x00a73800 0x002e1800 0x006e9000 0x00c69c00
LBA[40]: 0x003c3000 0x00ec3400 0x005f3c00 0x001ad400 0x0012b800 0x01067800 0x00cddc00 0x00ab6800 0x008ea800 0x001b6400
LBA[50]: 0x00837c00 0x00316400 0x00666000 0x013f9000 0x0057cc00 0x011ee400 0x01055000 0x00f16c00 0x01333000 0x00cb7400
LBA[60]: 0x001dd400 0x00a21c00 0x00a66000 0x00c3dc00 0x0108e400 0x003eac00 0x00fabc00 0x008c1000 0x00c6c800 0x00147400
LBA[70]: 0x0028bc00 0x00d21400 0x00701400 0x011d0400 0x013f5000 0x0096b800 0x00febc00 0x01298800 0x009a5000 0x008d4400
LBA[80]: 0x01135c00 0x00e1e800 0x00924000 0x00b51c00 0x013a1400 0x003b2800 0x0096c800 0x005c1000 0x0036b400 0x0138ec00
LBA[90]: 0x012bf800 0x00bb1400 0x00186800 0x008cdc00 0x00b1a800 0x00d6f000 0x01018400 0x012a6c00 0x00a7cc00 0x00f1d400

PRE LBA LIST:
LBA[ 0]: 0x011cfc00 0x00970c00 0x0040a800 0x01078c00 0x01298c00 0x01069000 0x001fdc00 0x01389800 0x00cba800 0x00b62800

Current Time: 2023-06-27 02:09:55

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[1]：案例三，inject PRE lba bug

随机校验数据，簇内校验数据出错

BUG 002[1]: [Thread: 4] RANDOM VERIFY SECTOR[365] DIFFER: 1
| CORRECT LBA[0]: 0x2577c00 | ERROR LBA[365]: 0x2577d6d | filename: lba.raw
LBA: 0x2577d6d | thread_num: 10, bitmap: 0 0 0 0 1 0 0 0 0 0

CORRECT LBA[0]: 0x2577c00 | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b
[LOOP: 1 WRITE: 1355 SECTOR: 0 / NUM: 1024 / ZERO: 38]
CORRECT IO Time: 2023-06-28 22:41:43

ERROR LBA[365]: 0x2577d6d | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b
[LOOP: 1 WRITE: 1355 SECTOR: 365 / NUM: 1024 / ZERO: 38]
INCONSISTENT IO Time: 2023-06-28 22:41:43

CORRECT LBA[366]: 0x2577d6e | HF: 0x04001274 F1: 0x3dafe710 F2:
0xb0241d6b [LOOP: 1 WRITE: 1355 SECTOR: 366 / NUM: 1024 / ZERO: 38]
CORRECT IO Time: 2023-06-28 22:41:43

---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 4] VERIFY SECTOR[365] DIFFER: 1 | read[1355] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x00853400 disk/file
hd_write_verify_dump -c -D -L 0x01c3a800 disk/file

CORRECT LBA[0]: 0x2577c00 | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b [LOOP: 1 WRITE: 1355 SECTOR: 0 / NUM: 1024 /
CORRECT IO Time: 2023-06-28 22:41:43

ERROR LBA[365]: 0x2577d6d | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b [LOOP: 1 WRITE: 1355 SECTOR: 365 / NUM: 1024
INCONSISTENT IO Time: 2023-06-28 22:41:43

CORRECT LBA[366]: 0x2577d6e | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b [LOOP: 1 WRITE: 1355 SECTOR: 366 / NUM: 102
CORRECT IO Time: 2023-06-28 22:41:43
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-06-28 22:40:43
2477, 1709 | 2504, 1737 | 2470, 1706 | 2489, 1732 | 2497, 1734 | 2511, 1733 | 2472, 1723 | 2503, 1728 | 2509, 1754 | 2522, 1748 |

Current Time: 2023-06-28 22:43:39
BUG 002[1]: [Thread: 4] RANDOM VERIFY SECTOR[365] DIFFER: 1 | CORRECT LBA[0]: 0x2577c00 | ERROR LBA[365]: 0x2577d6d | filename: lba.raw
LBA: 0x2577d6d | thread_num: 10, bitmap: 0 0 0 0 1 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x9eb800 | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b [LOOP: 1 WRITE: 3462 SECTOR: 0 / NUM: 1024 / ZERO: 38]
LAST WRITE IO Time: 2023-06-28 22:43:39

Buffer addr: 0xfffee9d70000 | Physical addr: 0x203705230000

LBA LIST:
LBA[ 0]: 0x009eb800 0x018d5400 0x005eb800 0x0221b400 0x00490c00 0x016e1c00 0x00dbf400 0x01703c00 0x02573000 0x0129f000
LBA[10]: 0x00fb0400 0x01a11800 0x004bf800 0x01d6fc00 0x0002a800 0x00d4ec00 0x01df1000 0x00e29000 0x012fc400 0x00118c00
LBA[20]: 0x015ac800 0x00683c00 0x018cb400 0x01cb2c00 0x0027c800 0x021a5400 0x00457400 0x026cdc00 0x026abc00 0x02546c00
LBA[30]: 0x010dc800 0x00ec7c00 0x002bbc00 0x01f7f400 0x01f59c00 0x027ef800 0x01f91800 0x01ef4c00 0x025fb000 0x0006b400
LBA[40]: 0x002bcc00 0x016b5c00 0x017f5000 0x013fb000 0x00dd6400 0x02087000 0x0221d000 0x01919c00 0x00a34400 0x006e1c00
LBA[50]: 0x0087b800 0x02734c00 0x00a3b400 0x02656000 0x00476400 0x0059c400 0x0058fc00 0x0258bc00 0x00e9d400 0x0176ac00
LBA[60]: 0x0162c400 0x007e3c00 0x01add800 0x00050800 0x00809400 0x0063a000 0x02629000 0x01833800 0x021dd400 0x00688c00
LBA[70]: 0x01b36800 0x00dd7000 0x00f02800 0x01757800 0x0027e800 0x02385000 0x008bf800 0x0084b000 0x00107c00 0x02048c00
LBA[80]: 0x0017b400 0x0161bc00 0x019d9800 0x012c8c00 0x00df2000 0x01338c00 0x018bd400 0x02429800 0x01f91c00 0x00b5b800
LBA[90]: 0x000ac800 0x00678c00 0x00797400 0x025f3c00 0x01b83800 0x00ad6800 0x01376c00 0x00136000 0x01be7c00 0x0101d400

PRE LBA LIST:
LBA[ 0]: 0x006e1800 0x00b3f000 0x02249c00 0x01e6b800 0x007c4800 0x0098bc00 0x01d14400 0x0060d800 0x022da400 0x00457000

RANDOM VERIFY:
LBA: 0x2577c00 | HF: 0x04001274 F1: 0x3dafe710 F2: 0xb0241d6b [LOOP: 1 WRITE: 1355 SECTOR: 0 / NUM: 1024 / ZERO: 38]
RANDOM VERIFY IO Time: 2023-06-28 22:41:43

Buffer addr: 0xfffee9190000 | Physical addr: 0x2025880a0000

LBA LIST:
LBA[ 0]: 0x02577c00 0x01c3a800 0x0059b400 0x006f5000 0x026f0c00 0x02345000 0x01603400 0x027ee000 0x02387c00 0x01340400
LBA[10]: 0x01ff4000 0x01b6a000 0x006a8800 0x01b49400 0x021b8000 0x013b6800 0x00648400 0x00e81800 0x014da400 0x014e6400
LBA[20]: 0x01eda800 0x0265f000 0x021bc800 0x027fb800 0x00a44000 0x023bd800 0x00945400 0x01393000 0x00dc8400 0x00f0d400
LBA[30]: 0x01805800 0x003ffc00 0x007b9000 0x004ee000 0x0002a000 0x01ad2400 0x017a2000 0x010c9800 0x01fff400 0x0064c400
LBA[40]: 0x00716400 0x00072000 0x01b49c00 0x023da000 0x00e6c000 0x00248800 0x01b47800 0x001ae000 0x00d97800 0x00b90000
LBA[50]: 0x015aec00 0x0067f000 0x027e7400 0x004b2000 0x0075f400 0x01849800 0x00d18c00 0x00965000 0x00dea400 0x01603800
LBA[60]: 0x00667800 0x01ca8800 0x02621000 0x026be000 0x00cf5000 0x02275800 0x01221800 0x0092a800 0x01900000 0x022d8c00
LBA[70]: 0x00309000 0x02489000 0x01091800 0x00b28800 0x00e9b400 0x020fc800 0x00b9a000 0x0226d400 0x021c4800 0x01818800
LBA[80]: 0x02537800 0x00338800 0x00599c00 0x0146ec00 0x0197a400 0x02014800 0x01ef5c00 0x00792c00 0x01e90000 0x020ae000
LBA[90]: 0x023bec00 0x01f79000 0x00ebf000 0x00cc1800 0x02587800 0x007c0400 0x01a9f800 0x00be1400 0x024db000 0x016b1000

PRE LBA LIST:
LBA[ 0]: 0x00853400 0x009e0800 0x00e46000 0x02578000 0x026a3400 0x01a06400 0x01c42000 0x0036e400 0x6c3ae975 0x00b99c00

Current Time: 2023-06-28 22:43:39

pause and exit work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[2]：案例一，inject SECTOR lba bug

随机校验数据，簇内校验数据出错

BUG 002[2]: [Thread: 2] RANDOM VERIFY ZERO SECTOR[63] DIFFER: 1 | CORRECT LBA[0]: 0x57ef00 | ERROR LBA[63]: 0x57ef3f | filename: lba.raw
LBA: 0x57ef3f | thread_num: 10, bitmap: 0 0 1 0 0 0 0 0 0 0

CORRECT LBA[0]: 0x57ef00 | HF: 0x00401272 F1: 0x7f5b05a5 F2: 0x26e71923 [LOOP: 1 WRITE: 1005 SECTOR: 0 / NUM: 64 / ZERO: 2]
CORRECT IO Time: 2023-07-03 18:10:52

ERROR LBA[63]: 0x57ef3f | HF: 0x7c7c7c7c F1: 0x7c7c7c7c F2: 0x7c7c7c7c [LOOP: 2088533116 WRITE: 2088533116 SECTOR: 31868 / NUM: 31868 / ZERO:]
INCONSISTENT IO Time: 2088535016-2088533117-2088533116

NEXT LBA: 0x115dc00 | HF: 0x00401272 F1: 0x7f5b05a5 F2: 0x26e71923 [LOOP: 1 WRITE: 1006 SECTOR: 0 / NUM: 64 / ZERO: 2]
NEXT IO Time: 2023-07-03 18:10:52

---runtime校验

```
BUG 007[4]: [Thread: 2] VERIFY ZERO SECTOR[63] DIFFER: 1 | read[1006] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x001f2200 disk/file
hd_write_verify_dump -c -D -L 0x0115dc00 disk/file

CORRECT LBA[0]: 0x57ef00 | HF: 0x00401272 F1: 0x7f5b05a5 F2: 0x26e71923 [LOOP: 1 WRITE: 1005 SECTOR: 0 / N
CORRECT IO Time: 2023-07-03 18:10:52

ERROR LBA[63]: 0x57ef3f | HF: 0x7c7c7c7c F1: 0x7c7c7c7c F2: 0x7c7c7c7c [LOOP: 2088533116 WRITE: 208853311
INCONSISTENT IO Time: 2088535016-2088533117-2088533116 2088533116:2088533116:2088533116

NEXT LBA: 0x115dc00 | HF: 0x00401272 F1: 0x7f5b05a5 F2: 0x26e71923 [LOOP: 1 WRITE: 1006 SECTOR: 0 / NUM: 6
NEXT IO Time: 2023-07-03 18:10:52
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-03 18:09:52
43, 47 | 42, 46 | 42, 46 | 44, 47 | 44, 47 | 43, 47 | 44, 47 | 43, 46 | 43, 46 | 44, 47 |

Current Time: 2023-07-03 18:11:13
BUG 002[2]: [Thread: 2] RANDOM VERIFY ZERO SECTOR[63] DIFFER: 1 | CORRECT LBA[0]: 0x57ef00 | ERROR LBA[63]: 0x57ef3f | filename: lba.raw
LBA: 0x57ef3f | thread_num: 10, bitmap: 0 0 1 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x13691c0 | HF: 0x00401272 F1: 0x7f5b05a5 F2: 0x26e71923 [LOOP: 1 WRITE: 1402 SECTOR: 0 / NUM: 64 / ZERO: 2]
LAST WRITE IO Time: 2023-07-03 18:11:13

Buffer addr: 0xffff6c050000 | Physical addr: 0x28308370000

LBA LIST:
LBA[ 0]: 0x013691c0 0x00fa2f80 0x013c3880 0x007ldd80 0x001fbe00 0x0087f800 0x003ad400 0x00455000 0x0013c180 0x0121b480
LBA[10]: 0x00fb7400 0x0044b640 0x006d0640 0x00a889c0 0x004e7840 0x00b01140 0x003dc040 0x0093da00 0x00f5e480 0x00a8a640
LBA[20]: 0x00a85f00 0x00ece480 0x011c2980 0x000f6b00 0x00530600 0x010f7c80 0x0014da00 0x010e6300 0x000073c0 0x00788d00
LBA[30]: 0x001e7300 0x002c0e40 0x00309b80 0x012d7000 0x0047af40 0x004fa000 0x01065a00 0x0038cf40 0x00d79380 0x00abc800
LBA[40]: 0x00683000 0x01289b40 0x0030d780 0x013d6300 0x00520c00 0x00a75180 0x00a34e40 0x00521e80 0x00d4af80 0x000184c0
LBA[50]: 0x00e5c600 0x00b94000 0x00f33500 0x00edcf00 0x012c8c80 0x00a8d780 0x00dc3140 0x005bf000 0x00640040 0x006f9180
LBA[60]: 0x00193300 0x003a2000 0x00a68180 0x011f9840 0x0034a140 0x00f02000 0x00e96fc0 0x00d197c0 0x00074380 0x0054e980
LBA[70]: 0x006c5280 0x01115700 0x00af5e80 0x00ed1bc0 0x00992a80 0x00cc1a80 0x006d9140 0x000dc500 0x012b8080 0x00df0900
LBA[80]: 0x000eb980 0x004e83c0 0x00e22e00 0x00f67e00 0x01161d80 0x004ab280 0x008de900 0x004c6040 0x00678300 0x001d9fc0
LBA[90]: 0x008ec740 0x0072eac0 0x0132b4c0 0x013d8040 0x00166040 0x00bb7f80 0x012faa40 0x011e3940 0x01325540 0x0035a440

PRE LBA LIST:
LBA[ 0]: 0x00fbd600 0x00c0ebc0 0x00988000 0x008d1b40 0x000f3800 0x0052dec0 0x0066d680 0x00df7800 0x00204040 0x00e7d400

RANDOM VERIFY:
LBA: 0x57ef00 | HF: 0x00401272 F1: 0x7f5b05a5 F2: 0x26e71923 [LOOP: 1 WRITE: 1005 SECTOR: 0 / NUM: 64 / ZERO: 2]
RANDOM VERIFY IO Time: 2023-07-03 18:10:52

Buffer addr: 0xffff6c070000 | Physical addr: 0x2005f1b0000

LBA LIST:
LBA[ 0]: 0x0057ef00 0x0115dc00 0x00176880 0x0136fc00 0x00ac8480 0x0119f1c0 0x00e129c0 0x00289a40 0x00fba000 0x00d3fdc0
LBA[10]: 0x00cf7700 0x00727240 0x00ec27c0 0x00afe200 0x00ea8380 0x003d0c80 0x00084100 0x00e43d40 0x00bc8c00 0x00c60600
LBA[20]: 0x0103ff80 0x004a0740 0x00686000 0x009999c0 0x00aba040 0x00f03540 0x00453f00 0x00081440 0x00d33b80 0x00682800
LBA[30]: 0x00785440 0x005c1d80 0x006dfdc0 0x00337540 0x00cfe100 0x00cf7740 0x01298940 0x009b4a80 0x00791640 0x00f081c0
LBA[40]: 0x0054b6c0 0x0023f940 0x0022d040 0x00d722c0 0x00dfd000 0x00a43c00 0x008696c0 0x005676c0 0x01069580 0x01388500
LBA[50]: 0x00383d80 0x0124e200 0x001e7100 0x0032f000 0x00658a40 0x00ec9800 0x000f6a00 0x00b86780 0x00660540 0x01247f00
LBA[60]: 0x0025b800 0x00fabcc0 0x012ff740 0x0034d800 0x00cd9800 0x00023900 0x0048be40 0x007ec800 0x00768040 0x003a8000
LBA[70]: 0x0054f000 0x01077940 0x009b4d80 0x000e2f80 0x0137de00 0x00ee9ec0 0x00e46400 0x01258200 0x0012c800 0x011bd5c0
LBA[80]: 0x0100d600 0x002bd540 0x01007000 0x00ecfc00 0x00584b80 0x00fd4980 0x004ec100 0x003e4800 0x00cf0000 0x0084be80
LBA[90]: 0x00500c00 0x0035dd40 0x01398000 0x006d9800 0x00bc4280 0x006f8080 0x00cb4f00 0x009bf800 0x006ece80 0x0110a500

PRE LBA LIST:
LBA[ 0]: 0x001f2200 0x01099d80 0x00a47680 0x00829800 0x00f3f300 0x00529880 0x0091e2c0 0x00f7be00 0x00a00880 0x013bcb80

DETECT READ ERROR:
LBA: 0x57ef3f | HF: 0x7c7c7c7c F1: 0x7c7c7c7c F2: 0x7c7c7c7c [LOOP: 2088533116 WRITE: 2088533116 SECTOR: 31868 / NUM: 31868 / ZERO: 2088533116]
DETECT ERROR IO Time: 2088535016-2088533117-2088533116 2088533116:2088533116:2088533116

Buffer addr: 0xffff6c077e00 | Physical addr: 0x2005f1b7e00

LBA LIST:
LBA[ 0]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[10]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[20]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[30]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[40]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[50]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[60]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[70]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[80]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
LBA[90]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c

PRE LBA LIST:
LBA[ 0]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c

Current Time: 2023-07-03 18:11:13

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[2]：案例二，inject LIST lba bug

随机校验数据，簇内校验数据出错

BUG 002[2]: [Thread: 1] RANDOM VERIFY ZERO SECTOR[47] DIFFER: 1
| CORRECT LBA[0]: 0x115e140 | ERROR LBA[47]: 0x115e16f | filename: lba.raw
LBA: 0x115e16f | thread_num: 10, bitmap: 0 1 0 0 0 0 0 0 0 0

CORRECT LBA[0]: 0x115e140 | HF: 0x00401271 F1: 0x2025f303 F2: 0xac7f3f18
[LOOP: 1 WRITE: 5348 SECTOR: 0 / NUM: 64 / ZERO: 18]
CORRECT IO Time: 2023-07-03 20:04:09

ERROR LBA[47]: 0x115e16f | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000
[LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[48]: 0x115e170 | HF: 0x00000000 F1: 0x00000000 F2:
0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
CORRECT IO Time: 1900-01-00 00:00:00
---runtime校验

```
BUG 007[2] LAST IO?: [Thread: 1] VERIFY ZERO SECTOR[47] DIFFER: 1 | read[5348] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x012ed1c0 disk/file
hd_write_verify_dump -c -D -L 0x00bef680 disk/file

CORRECT LBA[0]: 0x115e140 | HF: 0x00401271 F1: 0x2025f303 F2: 0xac7f3f18 [LOOP: 1 WRITE: 5348 SECTOR: 0 / NUM: 64 / Z
CORRECT IO Time: 2023-07-03 20:04:09

ERROR LBA[47]: 0x115e16f | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO:
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[48]: 0x115e170 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO
CORRECT IO Time: 1900-01-00 00:00:00
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-03 20:03:09
169, 173 | 167, 171 | 170, 173 | 169, 173 | 167, 170 | 166, 169 | 169, 172 | 163, 166 | 165, 168 | 171, 174 |

Current Time: 2023-07-03 20:04:19
BUG 002[2]: [Thread: 1] RANDOM VERIFY ZERO SECTOR[47] DIFFER: 1 | CORRECT LBA[0]: 0x115e140 | ERROR LBA[47]: 0x115e16f | filename: lba.raw
LBA: 0x115e16f | thread_num: 10, bitmap: 0 1 0 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x1185880 | HF: 0x00401271 F1: 0x2025f303 F2: 0xac7f3f18 [LOOP: 1 WRITE: 5863 SECTOR: 0 / NUM: 64 / ZERO: 18]
LAST WRITE IO Time: 2023-07-03 20:04:19

Buffer addr: 0xfffeb8050000 | Physical addr: 0x373df70000

LBA LIST:
LBA[ 0]: 0x01185880 0x00489e00 0x0119c000 0x013dfb00 0x00aee000 0x00cf5480 0x005044c0 0x01359180 0x004e5400 0x00015000
LBA[10]: 0x00992740 0x00eef100 0x00e3f580 0x00344600 0x00c8b680 0x01111b80 0x0059e340 0x00dc50c0 0x01039a00 0x001ae980
LBA[20]: 0x00507000 0x01019380 0x00d81240 0x00f08380 0x0021ac80 0x000a5800 0x0088de00 0x00eb9800 0x003c3d00 0x009d1840
LBA[30]: 0x000c3900 0x010a3080 0x008d4480 0x004ca200 0x00336c00 0x00d35080 0x0096c480 0x006e6ec0 0x00e27080 0x00b4da80
LBA[40]: 0x00bf4c80 0x00424c00 0x00a94a00 0x00440c80 0x00d7e400 0x0058d7c0 0x000aa400 0x00f27640 0x00435ac0 0x01305c00
LBA[50]: 0x00957a40 0x000a2900 0x00354880 0x008b0880 0x01269680 0x0125f840 0x00686000 0x001d1880 0x00846a80 0x00ce14c0
LBA[60]: 0x003e8980 0x010f0e00 0x0098e640 0x0063e040 0x008938c0 0x01022cc0 0x00f0d5c0 0x008da4c0 0x008cb600 0x00591440
LBA[70]: 0x013b7c00 0x002ae640 0x0089cc40 0x00631440 0x00d6f000 0x003e0400 0x004746c0 0x003ed380 0x00c36200 0x0005de80
LBA[80]: 0x01395e40 0x00196180 0x00f9ce00 0x0105ffc0 0x0087ca00 0x00fd1e80 0x003f7700 0x00ac9880 0x007f7300 0x0006f280
LBA[90]: 0x011fb4c0 0x002d8380 0x0093c440 0x00abbb40 0x00f92cc0 0x008e5540 0x00f65080 0x00ba0100 0x00919a00 0x00702ac0

PRE LBA LIST:
LBA[ 0]: 0x00462d80 0x01094540 0x00db0240 0x008a8700 0x00cbff00 0x01025780 0x01279800 0x008d5380 0x00c19c80 0x00facf00

RANDOM VERIFY:
LBA: 0x115e140 | HF: 0x00401271 F1: 0x2025f303 F2: 0xac7f3f18 [LOOP: 1 WRITE: 5348 SECTOR: 0 / NUM: 64 / ZERO: 18]
RANDOM VERIFY IO Time: 2023-07-03 20:04:09

Buffer addr: 0xfffeb8070000 | Physical addr: 0x2544d50000

LBA LIST:
LBA[ 0]: 0x0115e140 0x00bef680 0x00b1d980 0x00211680 0x01301900 0x013ad480 0x0032a580 0x01217dc0 0x003c0d00 0x008cb180
LBA[10]: 0x010bd180 0x00aebf00 0x00458740 0x0082eac0 0x00ecdb80 0x0129bd40 0x00040640 0x00e5cd80 0x00235200 0x002e69c0
LBA[20]: 0x00ed5d80 0x01088000 0x00920780 0x009d8400 0x000c9400 0x01137a00 0x0134fec0 0x0001e2c0 0x008658c0 0x013491c0
LBA[30]: 0x00d68080 0x010b8ac0 0x00216c00 0x009f4300 0x0020bd80 0x002e6f00 0x000bc880 0x0076c200 0x0122d200 0x010de100
LBA[40]: 0x00cfd580 0x00c67280 0x00ae4800 0x0072a940 0x00bf6180 0x00b82c00 0x000e9b80 0x00f27440 0x003d2d80 0x00066680
LBA[50]: 0x007098c0 0x00391a40 0x00210200 0x01158c00 0x00619a80 0x008cd300 0x008efb80 0x01090a00 0x011ef240 0x00c6a800
LBA[60]: 0x00ee2000 0x005f61c0 0x0024df80 0x00df0980 0x00b3ec80 0x003fa800 0x002c9d80 0x00039800 0x009ce040 0x01269b00
LBA[70]: 0x00f48040 0x013b9300 0x013142c0 0x004d8e00 0x006f2280 0x00121840 0x00aecb80 0x00e6d180 0x00551200 0x010d4240
LBA[80]: 0x00410040 0x00b7fd40 0x011ec0c0 0x00aed600 0x000858c0 0x008d5a00 0x012e6cc0 0x000db780 0x011df280 0x00554780
LBA[90]: 0x009b8780 0x0119c2c0 0x012f15c0 0x01154ac0 0x011da100 0x0063db80 0x00841300 0x000ab7c0 0x00fd6680 0x0051f000

PRE LBA LIST:
LBA[ 0]: 0x012ed1c0 0x00fe0880 0x00844280 0x000bb740 0x00c39440 0x00843540 0x000ed1c0 0x012f1e00 0x00c90640 0x012094c0

DETECT READ ERROR:
LBA: 0x115e16f | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
DETECT ERROR IO Time: 1900-01-00 00:00:00

Buffer addr: 0xfffeb8075e00 | Physical addr: 0x2544d55e00

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x5aa6e910 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Current Time: 2023-07-03 20:04:19

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[3]：案例一，inject SECTOR lba bug

批量校验数据，簇内校验数据出错

BUG 002[3]: [Thread: 3] VERIFY SECTOR[306] DIFFER: 1
| CORRECT LBA[0]: 0xb2a000 | ERROR LBA[306]: 0xb2a132 | filename: lba.raw
LBA: 0xb2a132 | thread_num: 10, bitmap: 0 0 0 1 0 0 0 0 0 0

CORRECT LBA[0]: 0xb2a000 | HF: 0x08001273 F1: 0x0bc823ae F2: 0xa2e9975f [LOOP: 1 WRITE: 515 SECTOR: 0 / NUM: 2048 / ZERO: 76]
CORRECT IO Time: 2023-06-26 14:44:58

ERROR LBA[306]: 0xb2a132 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[307]: 0xb2a133 | HF: 0x08001273 F1: 0x0bc823ae F2: 0xa2e9975f [LOOP: 1 WRITE: 515 SECTOR: 307 / NUM: 2048 / ZERO: 76]
CORRECT IO Time: 2023-06-26 14:44:58

---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 3] VERIFY SECTOR[306] DIFFER: 1 | read[515] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x008cf800 disk/file
hd_write_verify_dump -c -D -L 0x00de7800 disk/file

CORRECT LBA[0]: 0xb2a000 | HF: 0x08001273 F1: 0x0bc823ae F2: 0xa2e9975f [LOOP: 1 WRITE: 515 SECTOR: 0 / NUM: 2048 / ZERO: 76]
CORRECT IO Time: 2023-06-26 14:44:58

ERROR LBA[306]: 0xb2a132 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[307]: 0xb2a133 | HF: 0x08001273 F1: 0x0bc823ae F2: 0xa2e9975f [LOOP: 1 WRITE: 515 SECTOR: 307 / NUM: 2048 / ZERO: 76]
CORRECT IO Time: 2023-06-26 14:44:58
```

---checktime校验

```
Current Time: 2023-06-26 14:44:59
BUG 002[3]: [Thread: 3] BATCH VERIFY SECTOR[306] DIFFER: 1 | CORRECT LBA[0]: 0xb2a000 | ERROR LBA[306]: 0xb2a132 | filename: lba.raw
LBA: 0xb2a132 | thread_num: 10, bitmap: 0 0 0 1 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x39d800 | HF: 0x08001273 F1: 0x0bc823ae F2: 0xa2e9975f [LOOP: 1 WRITE: 530 SECTOR: 0 / NUM: 2048 / ZERO: 76]
LAST WRITE IO Time: 2023-06-26 14:44:59

Buffer addr: 0xfffee4430000 | Physical addr: 0x2824e20000

LBA LIST:
LBA[ 0]: 0x0039d800 0x00724000 0x00011800 0x00a21800 0x0023d800 0x012a8800 0x005eb800 0x0000e000 0x00f8a800 0x00af5000
LBA[10]: 0x009ad000 0x002cc800 0x00fda000 0x000f4000 0x00767800 0x005c6800 0x00402000 0x007d8800 0x01242800 0x0010b000
LBA[20]: 0x00322800 0x010e3000 0x00c9a800 0x00f0c000 0x00dca800 0x00217000 0x0013f000 0x003b7000 0x00de6800 0x0094d800
LBA[30]: 0x003dd800 0x0028e000 0x00cfd800 0x009e2000 0x00670000 0x00e0f800 0x01253800 0x012fa000 0x00f59800 0x00481000
LBA[40]: 0x00e1d800 0x005d6000 0x0119f000 0x0135f000 0x0094e000 0x0113f800 0x00424000 0x00964000 0x00fa5800 0x00ba3000
LBA[50]: 0x00a9d000 0x008c5800 0x000d1800 0x002dc000 0x0112d800 0x00af1800 0x000f8000 0x00456800 0x00103000 0x00126000
LBA[60]: 0x00e37000 0x006e6800 0x00289000 0x00995000 0x00a89800 0x0029d000 0x00e77800 0x00185000 0x0058a800 0x001c4800
LBA[70]: 0x00aec800 0x0044c800 0x00e5b800 0x00dd6000 0x00f92800 0x01036000 0x00648800 0x0084b800 0x01236000 0x00b79000
LBA[80]: 0x006b6800 0x0041d800 0x00995800 0x00dc6000 0x00a9a000 0x0117d800 0x0069d000 0x001df800 0x010cf800 0x005ff000
LBA[90]: 0x0080b800 0x00434000 0x012df000 0x01395800 0x003b5000 0x00f2f000 0x00c87800 0x002b9000 0x0055f000 0x005d1800

PRE LBA LIST:
LBA[ 0]: 0x0074b000 0x0000a000 0x009cc000 0x00fca000 0x0110a800 0x008d0800 0x0014e000 0x011c3000 0x0028a000 0x00e43000

----------------------------------------------------------------------------------------------------------------------------

BATCH VERIFY:
LBA: 0xb2a000 | HF: 0x08001273 F1: 0x0bc823ae F2: 0xa2e9975f [LOOP: 1 WRITE: 515 SECTOR: 0 / NUM: 2048 / ZERO: 76]
BATCH VERIFY IO Time: 2023-06-26 14:44:58

Buffer addr: 0xfffea0180000 | Physical addr: 0x21e5cd0000

LBA LIST:
LBA[ 0]: 0x00b2a000 0x00de7800 0x000f2000 0x010f3800 0x01041800 0x00e43000 0x0028a000 0x011c3000 0x0014e000 0x008d0800
LBA[10]: 0x0110a800 0x00fca000 0x009cc000 0x0000a000 0x0074b000 0x0039d800 0x00724000 0x00011800 0x00a21800 0x0023d800
LBA[20]: 0x012a8800 0x005eb800 0x0000e000 0x00f8a800 0x00af5000 0x009ad000 0x002cc800 0x00fda000 0x000f4000 0x00767800
LBA[30]: 0x005c6800 0x00402000 0x007d8800 0x01242800 0x0010b000 0x00322800 0x010e3000 0x00c9a800 0x00f0c000 0x00dca800
LBA[40]: 0x00217000 0x0013f000 0x003b7000 0x00de6800 0x0094d800 0x003dd800 0x0028e000 0x00cfd800 0x009e2000 0x00670000
LBA[50]: 0x00e0f800 0x01253800 0x012fa000 0x00f59800 0x00481000 0x00e1d800 0x005d6000 0x0119f000 0x0135f000 0x0094e000
LBA[60]: 0x0113f800 0x00424000 0x00964000 0x00fa5800 0x00ba3000 0x00a9d000 0x008c5800 0x000d1800 0x002dc000 0x0112d800
LBA[70]: 0x00af1800 0x000f8000 0x00456800 0x00103000 0x00126000 0x00e37000 0x006e6800 0x00289000 0x00995000 0x00a89800
LBA[80]: 0x0029d000 0x00e77800 0x00185000 0x0058a800 0x001c4800 0x00aec800 0x0044c800 0x00e5b800 0x00dd6000 0x00f92800
LBA[90]: 0x01036000 0x00648800 0x0084b800 0x01236000 0x00b79000 0x006b6800 0x0041d800 0x00995800 0x00dc6000 0x00a9a000

PRE LBA LIST:
LBA[ 0]: 0x008cf800 0x000f1800 0x01032800 0x001aa800 0x00113800 0x00710800 0x00fe4800 0x00fe4000 0x0011b000 0x00144000

----------------------------------------------------------------------------------------------------------------------------

DETECT READ ERROR:
LBA: 0xb2a132 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
DETECT ERROR IO Time: 1900-01-00 00:00:00

Buffer addr: 0xfffea01a6400 | Physical addr: 0x237d7a6400

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

----------------------------------------------------------------------------------------------------------------------------


Current Time: 2023-06-26 14:44:59

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[3]：案例二，inject SECTOR lba bug

批量校验数据，簇内校验数据出错

BUG 002[3]: [Thread: 4] VERIFY SECTOR[283] DIFFER: 84
| CORRECT LBA[0]: 0x39e800 | ERROR LBA[283]: 0x39e91b
| filename: lba.raw

CORRECT LBA[0]: 0x39e800 | HF: 0x04001274 F1: 0xc079f392 F2: 0x42a8b61c [LOOP: 1 WRITE: 474 SECTOR: 0 / NUM: 1024 / ZERO: 98]
CORRECT IO Time: 2023-06-23 00:02:28

ERROR LBA[283]: 0x39e91b | HF: 0xbfbfbfbf F1: 0xbfbfbfbf F2: 0xbfbfbfbf [LOOP: 3217014719 WRITE: 3217014719 SECTOR: 49087 / NUM: 49087 / ZERO: 3217014719]
INCONSISTENT IO Time: -1077950677--1077952576--1077952577 -1077952577:-1077952577:-1077952577

CORRECT LBA[367]: 0x39e96f | HF: 0x04001274 F1: 0xc079f392 F2: 0x42a8b61c [LOOP: 1 WRITE: 474 SECTOR: 367 / NUM: 1024 / ZERO: 98]
CORRECT IO Time: 2023-06-23 00:02:28
---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 4] VERIFY SECTOR[283] DIFFER: 84 | read[474] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x00621000 disk/file
hd_write_verify_dump -c -D -L 0x00216c00 disk/file

CORRECT LBA[0]: 0x39e800 | HF: 0x04001274 F1: 0xc079f392 F2: 0x42a8b61c [LOOP: 1 WRITE: 474 SECTOR: 0 / NUM: 1024
CORRECT IO Time: 2023-06-23 00:02:28

ERROR LBA[283]: 0x39e91b | HF: 0xbfbfbfbf F1: 0xbfbfbfbf F2: 0xbfbfbfbf [LOOP: 3217014719 WRITE: 3217014719 SECTOR
INCONSISTENT IO Time: -1077950677--1077952576--1077952577 -1077952577:-1077952577:-1077952577

CORRECT LBA[367]: 0x39e96f | HF: 0x04001274 F1: 0xc079f392 F2: 0x42a8b61c [LOOP: 1 WRITE: 474 SECTOR: 367 / NUM: 1
CORRECT IO Time: 2023-06-23 00:02:28
```

---checktime校验

```
BUG 002[3]: [Thread: 4] VERIFY SECTOR[283] DIFFER: 84 | CORRECT LBA[0]: 0x39e800 | ERROR LBA[283]: 0x39e91b | filename: lba.raw
LBA: 0x39e91b | thread_num: 10, bitmap: 0 0 0 0 1 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x39c000 | HF: 0x04001274 F1: 0xc079f392 F2: 0x42a8b61c [LOOP: 1 WRITE: 490 SECTOR: 0 / NUM: 1024 / ZERO: 98]
LAST WRITE IO Time: 2023-06-23 00:02:29

Buffer addr: 0xfffefad80000 | Physical addr: 0x222ca40000

LBA LIST:
LBA[ 0]: 0x0039c000 0x00736c00 0x001d1000 0x006eac00 0x00073400 0x00152800 0x00721400 0x00374400 0x00438c00 0x006bbc00
LBA[10]: 0x003ca800 0x000c7000 0x00449c00 0x0015d000 0x00276800 0x0065a000 0x0010a400 0x0043d800 0x0076c400 0x00140000
LBA[20]: 0x0054c000 0x00223800 0x006b7400 0x00221000 0x007fc400 0x000ef400 0x00568400 0x00663000 0x001c3c00 0x00732800
LBA[30]: 0x004ed000 0x00759800 0x00571000 0x0073b800 0x001b5800 0x0049d400 0x005f4800 0x00115400 0x0061ac00 0x003df400
LBA[40]: 0x0073e800 0x000e4400 0x002d1c00 0x00133c00 0x00310800 0x000c4400 0x004d5400 0x00428800 0x00391000 0x0030c800
LBA[50]: 0x005ae400 0x007cec00 0x00153400 0x003a6000 0x00359000 0x002f6c00 0x00374c00 0x001af400 0x004f4400 0x006f9800
LBA[60]: 0x00526c00 0x00346c00 0x007e3800 0x0073cc00 0x003a9800 0x00232c00 0x00062800 0x007e7800 0x0036bc00 0x00314400
LBA[70]: 0x0001b400 0x000a5400 0x000bb800 0x001b1800 0x00143400 0x007a4400 0x00434c00 0x00146c00 0x0066b400 0x005fa800
LBA[80]: 0x0017a400 0x0058e800 0x000a5800 0x0073c800 0x00552400 0x00375400 0x00791400 0x006ed800 0x003f3400 0x003e4800
LBA[90]: 0x00749400 0x00673000 0x004be800 0x00609c00 0x001f1000 0x00277800 0x00477400 0x001b1c00 0x003c1800 0x00244800

PRE LBA LIST:
LBA[ 0]: 0x003ee000 0x0017d000 0x0076c000 0x0068dc00 0x00541000 0x006a5000 0x004c3c00 0x0057c800 0x003fc400 0x00285800

----------------------------------------------------------------------------------------------------------------------

BATCH VERIFY:
LBA: 0x39e800 | HF: 0x04001274 F1: 0xc079f392 F2: 0x42a8b61c [LOOP: 1 WRITE: 474 SECTOR: 0 / NUM: 1024 / ZERO: 98]
BATCH VERIFY IO Time: 2023-06-23 00:02:28

Buffer addr: 0xfffefa060000 | Physical addr: 0x221ae70000

LBA LIST:
LBA[ 0]: 0x0039e800 0x00216c00 0x00627400 0x0071b800 0x00667000 0x0079dc00 0x00285800 0x003fc400 0x0057c800 0x004c3c00
LBA[10]: 0x006a5000 0x00541000 0x0068dc00 0x0076c000 0x0017d000 0x003ee000 0x0039c000 0x00736c00 0x001d1000 0x006eac00
LBA[20]: 0x00073400 0x00152800 0x00721400 0x00374400 0x00438c00 0x006bbc00 0x003ca800 0x000c7000 0x00449c00 0x0015d000
LBA[30]: 0x00276800 0x0065a000 0x0010a400 0x0043d800 0x0076c400 0x00140000 0x0054c000 0x00223800 0x006b7400 0x00221000
LBA[40]: 0x007fc400 0x000ef400 0x00568400 0x00663000 0x001c3c00 0x00732800 0x004ed000 0x00759800 0x00571000 0x0073b800
LBA[50]: 0x001b5800 0x0049d400 0x005f4800 0x00115400 0x0061ac00 0x003df400 0x0073e800 0x000e4400 0x002d1c00 0x00133c00
LBA[60]: 0x00310800 0x000c4400 0x004d5400 0x00428800 0x00391000 0x0030c800 0x005ae400 0x007cec00 0x00153400 0x003a6000
LBA[70]: 0x00359000 0x002f6c00 0x00374c00 0x001af400 0x004f4400 0x006f9800 0x00526c00 0x00346c00 0x007e3800 0x0073cc00
LBA[80]: 0x003a9800 0x00232c00 0x00062800 0x007e7800 0x0036bc00 0x00314400 0x0001b400 0x000a5400 0x000bb800 0x001b1800
LBA[90]: 0x00143400 0x007a4400 0x00434c00 0x00146c00 0x0066b400 0x005fa800 0x0017a400 0x0058e800 0x000a5800 0x0073c800

PRE LBA LIST:
LBA[ 0]: 0x00621000 0x0065e400 0x00026000 0x00582800 0x00594400 0x00297400 0x0032a000 0x0079d800 0x00121800 0x0010bc00

----------------------------------------------------------------------------------------------------------------------

DETECT READ ERROR:
LBA: 0x39e91b | HF: 0xbfbfbfbf F1: 0xbfbfbfbf F2: 0xbfbfbfbf [LOOP: 3217014719 WRITE: 3217014719 SECTOR: 49087 / NUM: 49087 / ZERO: 3217014719]
DETECT ERROR IO Time: -1077950677--1077952576--1077952577 -1077952577:-1077952577:-1077952577

Buffer addr: 0xfffefa083600 | Physical addr: 0x22320e3600

LBA LIST:
LBA[ 0]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[10]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[20]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[30]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[40]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[50]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[60]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[70]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[80]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf
LBA[90]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf

PRE LBA LIST:
LBA[ 0]: 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf 0xbfbfbfbf

----------------------------------------------------------------------------------------------------------------------
```

LBA工具基本功能演示

- BUG 002[3]：案例三，inject LIST lba bug

批量校验数据，簇内校验数据出错

BUG 002[3]: [Thread: 6] VERIFY SECTOR[400] DIFFER: 1
| CORRECT LBA[0]: 0x762800 | ERROR LBA[400]: 0x762990
| filename: lba.raw

CORRECT LBA[0]: 0x762800 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0
[LOOP: 1 WRITE: 550 SECTOR: 0 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

ERROR LBA[400]: 0x762990 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0
[LOOP: 1 WRITE: 550 SECTOR: 400 / NUM: 1024 / ZERO: 120]
INCONSISTENT IO Time: 2023-06-25 16:22:21

CORRECT LBA[401]: 0x762991 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0
[LOOP: 1 WRITE: 550 SECTOR: 401 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 6] VERIFY SECTOR[400] DIFFER: 1 | read[550] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x00521800 disk/file
hd_write_verify_dump -c -D -L 0x006dec00 disk/file

CORRECT LBA[0]: 0x762800 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 0 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

ERROR LBA[400]: 0x762990 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 400 / NUM: 1024 / ZERO: 120]
INCONSISTENT IO Time: 2023-06-25 16:22:21

CORRECT LBA[401]: 0x762991 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 401 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21
```

---checktime校验

```
BUG 002[3]: [Thread: 6] VERIFY SECTOR[400] DIFFER: 1 | CORRECT LBA[0]: 0x762800 | ERROR LBA[400]: 0x762990 | filename: lba.raw
LBA: 0x762990 | thread_num: 10, bitmap: 0 0 0 0 0 0 1 0 0 0

LAST CORRECT WRITE:
LBA: 0x6f2400 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 564 SECTOR: 0 / NUM: 1024 / ZERO: 120]
LAST WRITE IO Time: 2023-06-25 16:22:21

Buffer addr: 0xfffeb14d0000 | Physical addr: 0x53a2320000

LBA LIST:
LBA[ 0]: 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000 0x006df400 0x007df800 0x00117800 0x00310c00
LBA[10]: 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000 0x00503c00 0x004cc800 0x001a1c00 0x00560800
LBA[20]: 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800 0x00107000 0x006f7400 0x002dcc00 0x00037800
LBA[30]: 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400 0x0065bc00 0x00449c00 0x00018c00 0x00724000
LBA[40]: 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400 0x00182400 0x004ba800 0x002d5400 0x0004e400
LBA[50]: 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800 0x00366000 0x0079d400 0x006b3c00 0x00489c00
LBA[60]: 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400 0x00509400 0x00590000 0x002ad800 0x002cc400
LBA[70]: 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00 0x002c4400 0x00386000 0x00179c00 0x00421400
LBA[80]: 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00 0x001f3c00 0x0040b800 0x00133800 0x003fac00
LBA[90]: 0x00501000 0x000d3800 0x007c8c00 0x00611400 0x000c7c00 0x006b9800 0x00174000 0x000fe400 0x00386c00 0x006bfc00

PRE LBA LIST:
LBA[ 0]: 0x005b0c00 0x00714400 0x0030d400 0x0021ac00 0x00190c00 0x000d4800 0x0031dc00 0x00449800 0x007a9800 0x006f8c00

----------

BATCH VERIFY:
LBA: 0x762800 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 0 / NUM: 1024 / ZERO: 120]
BATCH VERIFY IO Time: 2023-06-25 16:22:21

Buffer addr: 0xfffeb08f0000 | Physical addr: 0x45028d0000

LBA LIST:
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006f8c00 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000
LBA[20]: 0x006df400 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800
LBA[40]: 0x00107000 0x006f7400 0x002dcc00 0x00037800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x004ba800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00509400 0x00590000 0x002ad800 0x002cc400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00
LBA[90]: 0x002c4400 0x00386000 0x00179c00 0x00421400 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00

PRE LBA LIST:
LBA[ 0]: 0x00521800 0x006a9c00 0x00042c00 0x006c7000 0x00755000 0x00115400 0x005c4800 0x007bf800 0x002e6000 0x002d2400

----------

DETECT READ ERROR:
LBA: 0x762990 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 400 / NUM: 1024 / ZERO: 120]
DETECT ERROR IO Time: 2023-06-25 16:22:21

Buffer addr: 0xfffeb0922000 | Physical addr: 0x53aeff2000

LBA LIST:
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006f8c00 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000
LBA[20]: 0xc6971fb4 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800
LBA[40]: 0x00107000 0x006f7400 0x002dcc00 0x00037800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x004ba800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00509400 0x00590000 0x002ad800 0x002cc400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00
LBA[90]: 0x002c4400 0x00386000 0x00179c00 0x00421400 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00

PRE LBA LIST:
LBA[ 0]: 0x00521800 0x006a9c00 0x00042c00 0x006c7000 0x00755000 0x00115400 0x005c4800 0x007bf800 0x002e6000 0x002d2400

----------

Current Time: 2023-06-25 16:22:21

pause and exit work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 002[4]：案例一，inject SECTOR lba bug

批量校验数据，簇内校验全0扇区数据出错

BUG 002[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54
| CORRECT LBA[0]: 0x7e9800 | ERROR LBA[970]: 0x7e9bca
| filename: lba.raw

CORRECT LBA[0]: 0x7e9800 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23
[LOOP: 1 WRITE: 533 SECTOR: 0 / NUM: 1024 / ZERO: 271]
CORRECT IO Time: 2023-06-23 00:02:29

ERROR LBA[970]: 0x7e9bca | HF: 0xb4b4b4b4 F1: 0xb4b4b4b4 F2: 0xb4b4b4b4
[LOOP: 3031741620 WRITE: 3031741620 SECTOR: 46260 / NUM: 46260 /
ZERO: 3031741620]
INCONSISTENT IO Time: -1263223776--1263225675--1263225676 -
1263225676:-1263225676:-1263225676

NEXT LBA: 0x60b000 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23
[LOOP: 1 WRITE: 534 SECTOR: 0 / NUM: 1024 / ZERO: 271]
NEXT IO Time: 2023-06-23 00:02:29
---runtime校验

```
BUG 007[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54 | read[534] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x006e5c00 disk/file
hd_write_verify_dump -c -D -L 0x0060b000 disk/file

CORRECT LBA[0]: 0x7e9800 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 533 SECTOR: 0 / NUM: 1024 / ZERO: 271]
CORRECT IO Time: 2023-06-23 00:02:29

ERROR LBA[970]: 0x7e9bca | HF: 0xb4b4b4b4 F1: 0xb4b4b4b4 F2: 0xb4b4b4b4 [LOOP: 3031741620 WRITE: 3031741620 SECTOR: 46260 / NU
INCONSISTENT IO Time: -1263223776--1263225675--1263225676 -1263225676:-1263225676:-1263225676

NEXT LBA: 0x60b000 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 534 SECTOR: 0 / NUM: 1024 / ZERO: 271]
NEXT IO Time: 2023-06-23 00:02:29
```

---checktime校验

```
BUG 002[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54 | CORRECT LBA[0]: 0x7e9800 | ERROR LBA[970]: 0x7e9bca | filename: lba.raw
LBA: 0x7e9bca | thread_num: 10, bitmap: 0 1 0 0 0 0 0 0 0 0

LAST CORRECT WRITE:
LBA: 0x717400 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 550 SECTOR: 0 / NUM: 1024 / ZERO: 271]
LAST WRITE IO Time: 2023-06-23 00:02:30

Buffer addr: 0xfffefb020000 | Physical addr: 0x2026e3060000

LBA LIST:
LBA[ 0]: 0x00717400 0x00697800 0x00560000 0x00343000 0x00621400 0x003e8400 0x00010c00 0x002a8800 0x0066e000 0x0037ec00
LBA[10]: 0x00317400 0x00769000 0x00738400 0x000d3400 0x005d4000 0x00444800 0x00510c00 0x005c4400 0x007b1800 0x004de400
LBA[20]: 0x004ab400 0x007c5400 0x007a5800 0x003b3c00 0x000e4000 0x0075fc00 0x0026f800 0x0005a800 0x001a9400 0x0028ac00
LBA[30]: 0x00254000 0x00422800 0x002a3000 0x0016bc00 0x006f9400 0x00048c00 0x004a7000 0x000ba000 0x005ed000 0x00772800
LBA[40]: 0x00793c00 0x00415400 0x006b8800 0x00095c00 0x004ef000 0x006dd400 0x00220c00 0x0012e800 0x000bac00 0x0033b000
LBA[50]: 0x00210400 0x0028cc00 0x000af400 0x0066e800 0x0051d800 0x00170c00 0x00713800 0x0049dc00 0x00735800 0x002ba000
LBA[60]: 0x001f8000 0x001bd400 0x000d3800 0x00505c00 0x00133400 0x0055a000 0x0009a000 0x0022f400 0x001dc400 0x006c0400
LBA[70]: 0x00485800 0x00088400 0x007bec00 0x0053ec00 0x00066c00 0x004b9800 0x003f6000 0x00295800 0x00655400 0x00565400
LBA[80]: 0x0028bc00 0x00539400 0x000d4800 0x003cf800 0x002f7800 0x003d0c00 0x00556c00 0x0040d800 0x004f4800 0x004a0c00
LBA[90]: 0x0077ac00 0x00347c00 0x00045c00 0x0060b400 0x0030d000 0x002a4800 0x007d8c00 0x0070c800 0x00219c00 0x00555400

PRE LBA LIST:
LBA[ 0]: 0x000b2000 0x0054fc00 0x000bd800 0x00041800 0x00639c00 0x00247c00 0x00155000 0x005c8800 0x002dbc00 0x0021c800

----------------------------------------------------------------------------------------------------------------------

BATCH VERIFY:
LBA: 0x7e9800 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 533 SECTOR: 0 / NUM: 1024 / ZERO: 271]
RANDOM VERIFY IO Time: 2023-06-23 00:02:29

Buffer addr: 0xfffefa380000 | Physical addr: 0x43e3fe0000

LBA LIST:
LBA[ 0]: 0x007e9800 0x0060b000 0x003ee400 0x007af400 0x0019d400 0x00648400 0x004ffc00 0x0021c800 0x002dbc00 0x005c8800
LBA[10]: 0x00155000 0x00247c00 0x00639c00 0x00041800 0x000bd800 0x0054fc00 0x000b2000 0x00717400 0x00697800 0x00560000
LBA[20]: 0x00343000 0x00621400 0x003e8400 0x00010c00 0x002a8800 0x0066e000 0x0037ec00 0x00317400 0x00769000 0x00738400
LBA[30]: 0x000d3400 0x005d4000 0x00444800 0x00510c00 0x005c4400 0x007b1800 0x004de400 0x004ab400 0x007c5400 0x007a5800
LBA[40]: 0x003b3c00 0x000e4000 0x0075fc00 0x0026f800 0x0005a800 0x001a9400 0x0028ac00 0x00254000 0x00422800 0x002a3000
LBA[50]: 0x0016bc00 0x006f9400 0x00048c00 0x004a7000 0x000ba000 0x005ed000 0x00772800 0x00793c00 0x00415400 0x006b8800
LBA[60]: 0x00095c00 0x004ef000 0x006dd400 0x00220c00 0x0012e800 0x000bac00 0x0033b000 0x00210400 0x0028cc00 0x000af400
LBA[70]: 0x0066e800 0x0051d800 0x00170c00 0x00713800 0x0049dc00 0x00735800 0x002ba000 0x001f8000 0x001bd400 0x000d3800
LBA[80]: 0x00505c00 0x00133400 0x0055a000 0x0009a000 0x0022f400 0x001dc400 0x006c0400 0x00485800 0x00088400 0x007bec00
LBA[90]: 0x0053ec00 0x00066c00 0x004b9800 0x003f6000 0x00295800 0x00655400 0x00565400 0x0028bc00 0x00539400 0x000d4800

PRE LBA LIST:
LBA[ 0]: 0x006e5c00 0x005de400 0x006ad400 0x00119800 0x002d3c00 0x0013f000 0x0006a800 0x006d4800 0x00288400 0x0049bc00

----------------------------------------------------------------------------------------------------------------------

DETECT READ ERROR:
LBA: 0x7e9bca | HF: 0xb4b4b4b4 F1: 0xb4b4b4b4 F2: 0xb4b4b4b4 [LOOP: 3031741620 WRITE: 3031741620 SECTOR: 46260 / NUM: 46260 / ZERO: 3031741620]
DETECT ERROR IO Time: -1263223776--1263225675--1263225676 -1263225676:-1263225676:-1263225676

Buffer addr: 0xfffefa3f9400 | Physical addr: 0x45f8669400

LBA LIST:
LBA[ 0]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[10]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[20]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[30]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[40]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[50]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[60]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[70]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[80]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[90]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4

PRE LBA LIST:
LBA[ 0]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4

----------------------------------------------------------------------------------------------------------------------


Current Time: 2023-06-23 00:02:35
```

LBA工具基本功能演示

- BUG 002[4]：案例二，inject LIST lba bug

批量校验数据，簇内校验全0扇区数据出错

BUG 002[4]: [Thread: 6] VERIFY ZERO SECTOR[974] DIFFER: 1
| CORRECT LBA[0]: 0x615c00 | ERROR LBA[974]: 0x615fce
| filename: lba.raw
LBA: 0x615fce | thread_num: 10, bitmap: 0 0 0 0 0 0 1 0 0 0

CORRECT LBA[0]: 0x615c00 | HF: 0x04001276 F1: 0x47877c66 F2: 0x657f8ed0 [LOOP: 1 WRITE: 396 SECTOR: 0 / NUM: 1024 / ZERO: 190]
CORRECT IO Time: 2023-06-26 00:22:21

ERROR LBA[974]: 0x615fce | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[975]: 0x615fcf | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
CORRECT IO Time: 1900-01-00 00:00:00

---runtime校验

```
BUG 007[2] LAST IO?: [Thread: 6] VERIFY ZERO SECTOR[974] DIFFER: 1 | read[396] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x001fa400 disk/file
hd_write_verify_dump -c -D -L 0x007b3000 disk/file

CORRECT LBA[0]: 0x615c00 | HF: 0x04001276 F1: 0x47877c66 F2: 0x657f8ed0 [LOOP: 1 WRITE: 396 SECTOR: 0 / NUM:
CORRECT IO Time: 2023-06-26 00:22:21

ERROR LBA[974]: 0x615fce | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[975]: 0x615fcf | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM:
CORRECT IO Time: 1900-01-00 00:00:00
```

---checktime校验

```
Current Time: 2023-06-26 00:22:23
BUG 002[4]: [Thread: 6] VERIFY ZERO SECTOR[974] DIFFER: 1 | CORRECT LBA[0]: 0x615c00 | ERROR LBA[974]: 0x615fce | filename: lba.raw
LBA: 0x615fce | thread_num: 10, bitmap: 0 0 0 0 0 0 1 0 0 0

LAST CORRECT WRITE:
LBA: 0x5fa800 | HF: 0x04001276 F1: 0x47877c66 F2: 0x657f8ed0 [LOOP: 1 WRITE: 402 SECTOR: 0 / NUM: 1024 / ZERO: 190]
LAST WRITE IO Time: 2023-06-26 00:22:21

Buffer addr: 0xffff75820000 | Physical addr: 0x300af3d0000

LBA LIST:
LBA[ 0]: 0x005fa800 0x0062d000 0x00337000 0x00643800 0x00312800 0x0075c000 0x00080000 0x00323800 0x00486800 0x007d9400
LBA[10]: 0x002cc400 0x0013c400 0x007fac00 0x002a3000 0x007d9c00 0x0012a000 0x006a2000 0x000f7000 0x006fbc00 0x0065a400
LBA[20]: 0x00634000 0x005c2c00 0x0029c000 0x0075f400 0x0078f800 0x0059d400 0x006ed800 0x00336400 0x00762000 0x0025a000
LBA[30]: 0x00058c00 0x001dd800 0x00062c00 0x000f1000 0x003a8000 0x00110400 0x0007b400 0x00380800 0x00698400 0x005c9400
LBA[40]: 0x00299400 0x0014d400 0x002d3c00 0x00331000 0x001d2800 0x00403000 0x00560400 0x0057d800 0x000b3400 0x003ec400
LBA[50]: 0x003a5000 0x00631c00 0x0077f800 0x00218000 0x000d2400 0x002eac00 0x0051f000 0x00200800 0x004c3800 0x00669c00
LBA[60]: 0x00460c00 0x000a4000 0x002aa800 0x000d4400 0x005f0800 0x0057a800 0x001b2400 0x006b8400 0x00599400 0x002ff400
LBA[70]: 0x004b8800 0x006cc800 0x00749800 0x00557000 0x00226400 0x0039cc00 0x00772800 0x00022400 0x002c3800 0x00042400
LBA[80]: 0x0034fc00 0x00394400 0x005a1800 0x007ddc00 0x00632c00 0x0049e400 0x00223c00 0x001bc800 0x0031cc00 0x00638400
LBA[90]: 0x006c0400 0x00597000 0x0022c800 0x001b3000 0x000ba800 0x00007c00 0x00317c00 0x0025f800 0x00413c00 0x00283c00

PRE LBA LIST:
LBA[ 0]: 0x003e1c00 0x00381c00 0x0050e000 0x0072fc00 0x007b3000 0x00615c00 0x001fa400 0x004d5800 0x004a4400 0x00322800

----------

BATCH VERIFY:
LBA: 0x615c00 | HF: 0x04001276 F1: 0x47877c66 F2: 0x657f8ed0 [LOOP: 1 WRITE: 396 SECTOR: 0 / NUM: 1024 / ZERO: 190]
RANDOM VERIFY IO Time: 2023-06-26 00:22:21

Buffer addr: 0xffff74d80000 | Physical addr: 0x300deb20000

LBA LIST:
LBA[ 0]: 0x00615c00 0x007b3000 0x0072fc00 0x0050e000 0x00381c00 0x003e1c00 0x005fa800 0x0062d000 0x00337000 0x00643800
LBA[10]: 0x00312800 0x0075c000 0x00080000 0x00323800 0x00486800 0x007d9400 0x002cc400 0x0013c400 0x007fac00 0x002a3000
LBA[20]: 0x007d9c00 0x0012a000 0x006a2000 0x000f7000 0x006fbc00 0x0065a400 0x00634000 0x005c2c00 0x0029c000 0x0075f400
LBA[30]: 0x0078f800 0x0059d400 0x006ed800 0x00336400 0x00762000 0x0025a000 0x00058c00 0x001dd800 0x00062c00 0x000f1000
LBA[40]: 0x003a8000 0x00110400 0x0007b400 0x00380800 0x00698400 0x005c9400 0x00299400 0x0014d400 0x002d3c00 0x00331000
LBA[50]: 0x001d2800 0x00403000 0x00560400 0x0057d800 0x000b3400 0x003ec400 0x003a5000 0x00631c00 0x0077f800 0x00218000
LBA[60]: 0x000d2400 0x002eac00 0x0051f000 0x00200800 0x004c3800 0x00669c00 0x00460c00 0x000a4000 0x002aa800 0x000d4400
LBA[70]: 0x005f0800 0x0057a800 0x001b2400 0x006b8400 0x00599400 0x002ff400 0x004b8800 0x006cc800 0x00749800 0x00557000
LBA[80]: 0x00226400 0x0039cc00 0x00772800 0x00022400 0x002c3800 0x00042400 0x0034fc00 0x00394400 0x005a1800 0x007ddc00
LBA[90]: 0x00632c00 0x0049e400 0x00223c00 0x001bc800 0x0031cc00 0x00638400 0x006c0400 0x00597000 0x0022c800 0x001b3000

PRE LBA LIST:
LBA[ 0]: 0x001fa400 0x004d5800 0x004a4400 0x00322800 0x00215800 0x004c8000 0x00107000 0x00028000 0x00663000 0x005ad800

----------

DETECT READ ERROR:
LBA: 0x615fce | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
DETECT ERROR IO Time: 1900-01-00 00:00:00

Buffer addr: 0xffff74df9c00 | Physical addr: 0x3032a459c00

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0xb1031988 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

----------

Current Time: 2023-06-26 00:22:23

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 003

PRE_LIST数据 或者 LBA_LIST数据校验出错

BUG 003: [Thread: 3] WRITE PART SECTOR | read[83] | filename: lba.raw

CORRECT LBA: 0xb62000 | HF: 0x04001273 F1: 0xa1b5ec90 F2: 0x31cedf50 [LOOP: 1 WRITE: 82 SECTOR: 0 / NUM: 1024 / ZERO: 31]
CORRECT LBA IO Time: 2023-06-12 17:11:53

ERROR LBA: 0xed4000 | HF: 0x04001273 F1: 0xa1b5ec90 F2: 0x31cedf50 [LOOP: 1 WRITE: 83 SECTOR: 0 / NUM: 1024 / ZERO: 31]
ERROR LBA IO Time: 2023-06-12 17:12:03

---runtime校验 & checktime校验

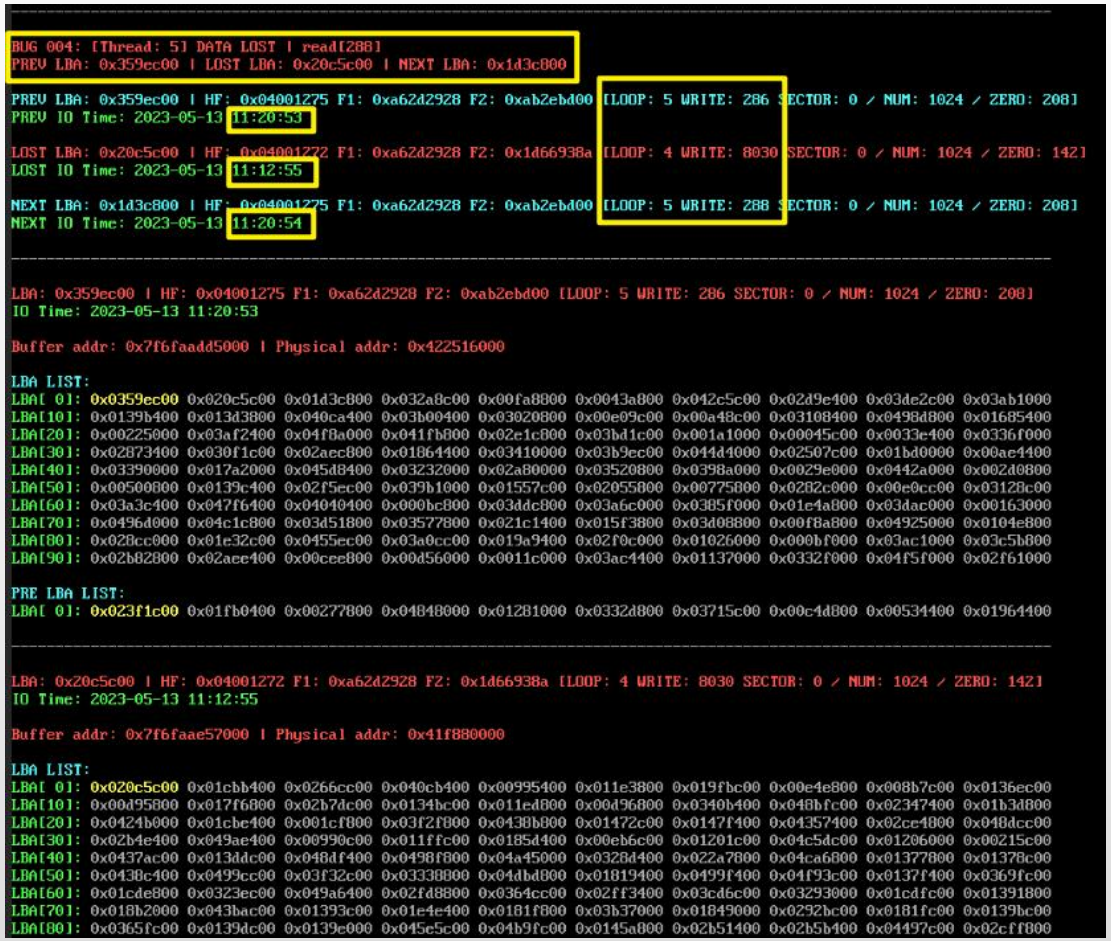
```
BUG 003: [Thread: 3] WRITE PART SECTOR | read[83] | filename: lba.raw

CORRECT LBA: 0xb62000 | HF: 0x04001273 F1: 0xa1b5ec90 F2: 0x31cedf50 [LOOP: 1 WRITE: 82 SECTOR: 0 / NUM: 1024 / ZERO: 31]
CORRECT LBA IO Time: 2023-06-12 17:11:53

ERROR LBA: 0xed4000 | HF: 0x04001273 F1: 0xa1b5ec90 F2: 0x31cedf50 [LOOP: 1 WRITE: 83 SECTOR: 0 / NUM: 1024 / ZERO: 31]
ERROR LBA IO Time: 2023-06-12 17:12:03

LBA: 0xb62000 | HF: 0x04001273 F1: 0xa1b5ec90 F2: 0x31cedf50 [LOOP: 1 WRITE: 82 SECTOR: 0 / NUM: 1024 / ZERO: 31]
IO Time: 2023-06-12 17:11:53

Buffer addr: 0xffff806f0000 | Physical addr: 0x502025c0000

LBA LIST:
LBA[ 0]: 0x00b62000 0x00ed4000 0x00ed3000 0x00122800 0x00ff2000 0x00d8a400 0x01090000 0x003ea000 0x01018000 0x00741c00
LBA[10]: 0x00f00400 0x00d2c800 0x00220c00 0x0042a000 0x00af7c00 0x0096b000 0x00b62c00 0x005c3000 0x00371400 0x01280000
LBA[20]: 0x00b0c000 0x00bbd000 0x00f9e400 0x00e77c00 0x0113d000 0x008e4800 0x0114f000 0x00c80000 0x0051d000 0x003cf000
LBA[30]: 0x0104c800 0x00e77000 0x001df000 0x003dac00 0x00da4c00 0x00509000 0x00647000 0x002c9000 0x004a0400 0x003ddc00
LBA[40]: 0x00830c00 0x00f50800 0x0107c000 0x01046400 0x00ae5800 0x00beb800 0x00440000 0x008b0000 0x0103bc00 0x000ee000
LBA[50]: 0x01037c00 0x00221c00 0x00f43c00 0x0004ac00 0x0139c000 0x002dc000 0x011dd000 0x00153800 0x00d3f000 0x00670000
LBA[60]: 0x0005d400 0x009ed000 0x00e08000 0x0060dc00 0x00a49000 0x00806c00 0x010c1c00 0x002eb000 0x00be8400 0x00eb4000
LBA[70]: 0x00d6e000 0x00297000 0x01316800 0x009b0800 0x00ac1000 0x00a15800 0x00208000 0x005e0000 0x008af800 0x00e02000
LBA[80]: 0x00716400 0x00ba7800 0x011f8000 0x010b9800 0x013b0000 0x00c5e000 0x00648800 0x013f6c00 0x013b8000 0x0062a800
LBA[90]: 0x00bbcc00 0x000b5400 0x0091c800 0x009bdc00 0x00af0800 0x00bbb000 0x00bafc00 0x00872800 0x00506c00 0x008a6800

PRE LBA LIST:
LBA[ 0]: 0x00082c00 0x0138b000 0x00bb0800 0x0031b400 0x00453000 0x00466000 0x00c3d800 0x006db000 0x00631800 0x00932000

LBA: 0xed4000 | HF: 0x04001273 F1: 0xa1b5ec90 F2: 0x31cedf50 [LOOP: 1 WRITE: 83 SECTOR: 0 / NUM: 1024 / ZERO: 31]
IO Time: 2023-06-12 17:12:03

Buffer addr: 0xffff80790000 | Physical addr: 0x58046a70000

LBA LIST:
LBA[ 0]: 0x00ed4000 0x00ed3000 0x00122800 0x00ff2000 0x00d8a400 0x01090000 0x003ea000 0x01018000 0x00741c00 0x00f00400
LBA[10]: 0x00d2c800 0x00220c00 0x0042a000 0x00af7c00 0x0096b000 0x00b62c00 0x005c3000 0x00371400 0x01280000 0x00b0c000
LBA[20]: 0x00bbd000 0x00f9e400 0x00e77c00 0x0113d000 0x008e4800 0x0114f000 0x00c80000 0x0051d000 0x003cf000 0x0104c800
LBA[30]: 0x00e77000 0x001df000 0x003dac00 0x00da4c00 0x00509000 0x00647000 0x002c9000 0x004a0400 0x003ddc00 0x00830c00
LBA[40]: 0x00f50800 0x0107c000 0x01046400 0x00ae5800 0x00beb800 0x00440000 0x008b0000 0x0103bc00 0x000ee000 0x01037c00
LBA[50]: 0x00221c00 0x00f43c00 0x0004ac00 0x0139c000 0x002dc000 0x011dd000 0x00153800 0x00d3f000 0x00670000 0x0005d400
LBA[60]: 0x009ed000 0x00e08000 0x0060dc00 0x00a49000 0x00806c00 0x010c1c00 0x002eb000 0x00be8400 0x00eb4000 0x00d6e000
LBA[70]: 0x00297000 0x01316800 0x009b0800 0x00ac1000 0x00a15800 0x00208000 0x005e0000 0x008af800 0x00e02000 0x00716400
LBA[80]: 0xe8ab4210 0x011f8000 0x010b9800 0x013b0000 0x00c5e000 0x00648800 0x013f6c00 0x013b8000 0x0062a800 0x00bbcc00
LBA[90]: 0x000b5400 0x0091c800 0x009bdc00 0x00af0800 0x00bbb000 0x00bafc00 0x00872800 0x00506c00 0x008a6800 0x00c93800

PRE LBA LIST:
LBA[ 0]: 0x00b62000 0x00082c00 0x0138b000 0x00bb0800 0x0031b400 0x00453000 0x00466000 0x00c3d800 0x006db000 0x00631800

Current Time: 2023-06-23 17:35:38
```

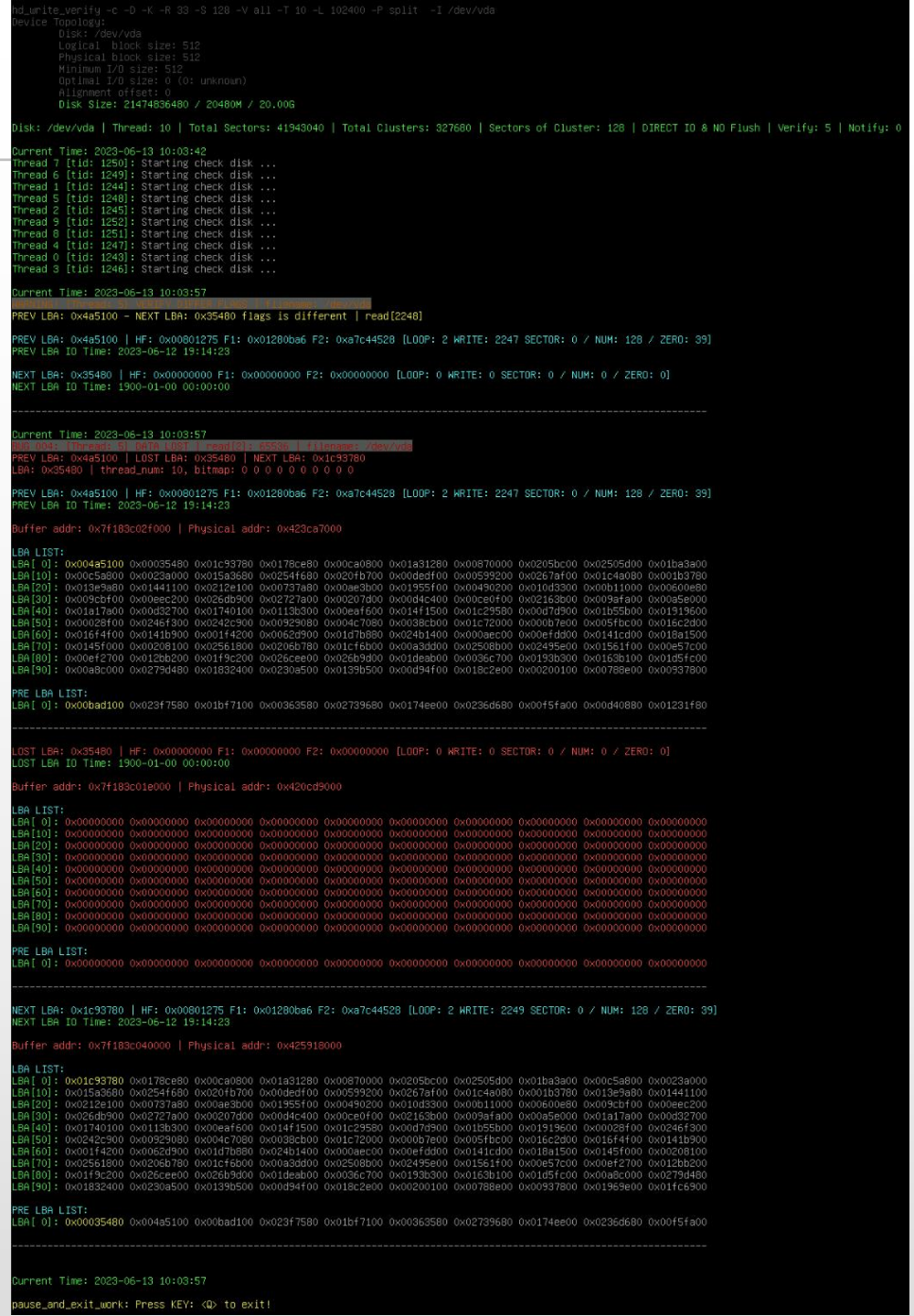
LBA工具基本功能演示

- BUG 004: 簇间数据校验，中间簇数据丢失

BUG 004 [Thread: 5] DATA LOST
PREV LBA: 0x4a5100 | LOST LBA: 0x35480 | NEXT LBA: 0x1c93780
第二轮测试，5号线程第2248次写IO，LBA地址：0x35480的一簇64K数据丢失

BUG 004 [Thread: 5] DATA LOST
PREV LBA: 0x359ec00 | LOST LBA: 0x20c5c00 | NEXT LBA: 0x1d3c800
第五轮测试，5号线程第287次写IO，LBA地址：0x20c5c00的一簇512K数据丢失





LBA工具基本功能演示

- BUG 005：inject FLAG lba bug

整体校验数据，首簇数据FLAG出错

BUG 005: [Thread: 0] VERIFY HD_FLAG ERROR | HF: 0x0400ffff[0x04001270] | filename: test.raw
LBA: 0x0 | thread_num: 10, bitmap: 1 0 0 0 0 0 0 0 0 0

Line: 2572 | HF: 0x4001270 != hd_flag: 0x400ffff, sector_num: 1024

LBA: 0x0 | HF: 0x0400ffff F1: 0xba0b9c1f F2: 0xd239bb83 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 127]
IO Time: 2023-07-09 23:47:39

---runtime校验

```
Line: 2572 | HF: 0x4001270 != hd_flag: 0x400ffff, sector_num: 1024

LBA: 0x0 | HF: 0x0400ffff F1: 0xba0b9c1f F2: 0xd239bb83 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 127]
IO Time: 2023-07-09 23:47:39

Buffer addr: 0x30910000 | Physical addr: 0x1814f0e0000

LBA LIST:
LBA[ 0]: 0x00000000 0x0034f800 0x00746400 0x00116000 0x001ca400 0x0046dc00 0x003ca800 0x004b7800 0x007bf800 0x000ef000
LBA[10]: 0x00585c00 0x0006c000 0x0033e000 0x002e5c00 0x00361c00 0x00664000 0x00444000 0x005b0000 0x00064800 0x000e0c00
LBA[20]: 0x00260800 0x000e9800 0x003c4000 0x003bc000 0x0036bc00 0x0052c000 0x0048d000 0x004e0400 0x0039cc00 0x00560000
LBA[30]: 0x00718000 0x00090000 0x007c9000 0x003f6000 0x004fb000 0x000b4800 0x00695000 0x00376000 0x005d5800 0x00480000
LBA[40]: 0x00345000 0x00634000 0x0006a800 0x00236000 0x006e9000 0x00353c00 0x000ae400 0x00420400 0x0028a400 0x0006a000
LBA[50]: 0x0024f800 0x007ca400 0x00220000 0x005c5800 0x0032cc00 0x00379800 0x00518400 0x005c4000 0x0013b000 0x0032c000
LBA[60]: 0x00140000 0x006e9400 0x000b6000 0x00227c00 0x00367800 0x006e4800 0x00646000 0x000b0800 0x001f8800 0x005e3800
LBA[70]: 0x00565800 0x00616000 0x007a3000 0x00497800 0x005cd400 0x001b9000 0x00758000 0x00318000 0x002b1400 0x00612000
LBA[80]: 0x007b1000 0x00170400 0x0064d800 0x0024fc00 0x007bc000 0x0013b400 0x0048bc00 0x00155000 0x007df000 0x007a7800
LBA[90]: 0x003cb800 0x00031c00 0x00658800 0x00300000 0x00197c00 0x004d0000 0x002e1400 0x001bd400 0x007a6c00 0x006dc000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
```

---checktime校验

```
Starting write file:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-09 23:47:39
0, 379 | 0, 384 | 0, 386 | 0, 391 | 0, 370 | 0, 368 | 0, 378 | 0, 380 | 0, 370 | 0, 372 |

Current Time: 2023-07-09 23:48:39
Thread 9 [tid: 360580]: Starting check file ...
Thread 5 [tid: 360576]: Starting check file ...
Thread 4 [tid: 360575]: Starting check file ...
Thread 3 [tid: 360574]: Starting check file ...
Thread 6 [tid: 360577]: Starting check file ...
Thread 7 [tid: 360578]: Starting check file ...
Thread 1 [tid: 360572]: Starting check file ...
Thread 8 [tid: 360579]: Starting check file ...
Thread 0 [tid: 360571]: Starting check file ...
Thread 2 [tid: 360573]: Starting check file ...

Current Time: 2023-07-09 23:48:39
BUG 005: [Thread: 0] VERIFY HD_FLAG ERROR | HF: 0x0400ffff[0x04001270] | filename: test.raw
LBA: 0x0 | thread_num: 10, bitmap: 1 0 0 0 0 0 0 0 0 0

LBA[0]: 0x0 | HF: 0x0400ffff F1: 0xba0b9c1f F2: 0xd239bb83 [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 127]
LBA[0] IO Time: 2023-07-09 23:47:39

Buffer addr: 0xffff98af0000 | Physical addr: 0x5033ba10000

LBA LIST:
LBA[ 0]: 0x00000000 0x0034f800 0x00746400 0x00116000 0x001ca400 0x0046dc00 0x003ca800 0x004b7800 0x007bf800 0x000ef000
LBA[10]: 0x00585c00 0x0006c000 0x0033e000 0x002e5c00 0x00361c00 0x00664000 0x00444000 0x005b0000 0x00064800 0x000e0c00
LBA[20]: 0x00260800 0x000e9800 0x003c4000 0x003bc000 0x0036bc00 0x0052c000 0x0048d000 0x004e0400 0x0039cc00 0x00560000
LBA[30]: 0x00718000 0x00090000 0x007c9000 0x003f6000 0x004fb000 0x000b4800 0x00695000 0x00376000 0x005d5800 0x00480000
LBA[40]: 0x00345000 0x00634000 0x0006a800 0x00236000 0x006e9000 0x00353c00 0x000ae400 0x00420400 0x0028a400 0x0006a000
LBA[50]: 0x0024f800 0x007ca400 0x00220000 0x005c5800 0x0032cc00 0x00379800 0x00518400 0x005c4000 0x0013b000 0x0032c000
LBA[60]: 0x00140000 0x006e9400 0x000b6000 0x00227c00 0x00367800 0x006e4800 0x00646000 0x000b0800 0x001f8800 0x005e3800
LBA[70]: 0x00565800 0x00616000 0x007a3000 0x00497800 0x005cd400 0x001b9000 0x00758000 0x00318000 0x002b1400 0x00612000
LBA[80]: 0x007b1000 0x00170400 0x0064d800 0x0024fc00 0x007bc000 0x0013b400 0x0048bc00 0x00155000 0x007df000 0x007a7800
LBA[90]: 0x003cb800 0x00031c00 0x00658800 0x00300000 0x00197c00 0x004d0000 0x002e1400 0x001bd400 0x007a6c00 0x006dc000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

LBA[1]: 0x1 | HF: 0x04001270 F1: 0xba0b9c1f F2: 0xd239bb83 [LOOP: 1 WRITE: 1 SECTOR: 1 / NUM: 1024 / ZERO: 127]
LBA[1] IO Time: 2023-07-09 23:47:39

Buffer addr: 0xffff98af0200 | Physical addr: 0x5033ba10200

LBA LIST:
LBA[ 0]: 0x00000000 0x0034f800 0x00746400 0x00116000 0x001ca400 0x0046dc00 0x003ca800 0x004b7800 0x007bf800 0x000ef000
LBA[10]: 0x00585c00 0x0006c000 0x0033e000 0x002e5c00 0x00361c00 0x00664000 0x00444000 0x005b0000 0x00064800 0x000e0c00
LBA[20]: 0x00260800 0x000e9800 0x003c4000 0x003bc000 0x0036bc00 0x0052c000 0x0048d000 0x004e0400 0x0039cc00 0x00560000
LBA[30]: 0x00718000 0x00090000 0x007c9000 0x003f6000 0x004fb000 0x000b4800 0x00695000 0x00376000 0x005d5800 0x00480000
LBA[40]: 0x00345000 0x00634000 0x0006a800 0x00236000 0x006e9000 0x00353c00 0x000ae400 0x00420400 0x0028a400 0x0006a000
LBA[50]: 0x0024f800 0x007ca400 0x00220000 0x005c5800 0x0032cc00 0x00379800 0x00518400 0x005c4000 0x0013b000 0x0032c000
LBA[60]: 0x00140000 0x006e9400 0x000b6000 0x00227c00 0x00367800 0x006e4800 0x00646000 0x000b0800 0x001f8800 0x005e3800
LBA[70]: 0x00565800 0x00616000 0x007a3000 0x00497800 0x005cd400 0x001b9000 0x00758000 0x00318000 0x002b1400 0x00612000
LBA[80]: 0x007b1000 0x00170400 0x0064d800 0x0024fc00 0x007bc000 0x0013b400 0x0048bc00 0x00155000 0x007df000 0x007a7800
LBA[90]: 0x003cb800 0x00031c00 0x00658800 0x00300000 0x00197c00 0x004d0000 0x002e1400 0x001bd400 0x007a6c00 0x006dc000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Current Time: 2023-07-09 23:48:39

pause_and_exit_work: Press KEY: <Q> to exit!
```

LBA工具基本功能演示

- BUG 006: inject SECTOR lba bug

簇间FLAGS数据校验出错

BUG 006: [Thread: 5] VERIFY DIFFER FLAGS | read[248], write[709] | filename: lba.raw
LBA: 0xc2c00 | thread_num：10, bitmap: 0 0 0 0 0 1 0 0 0 0

---runtime校验

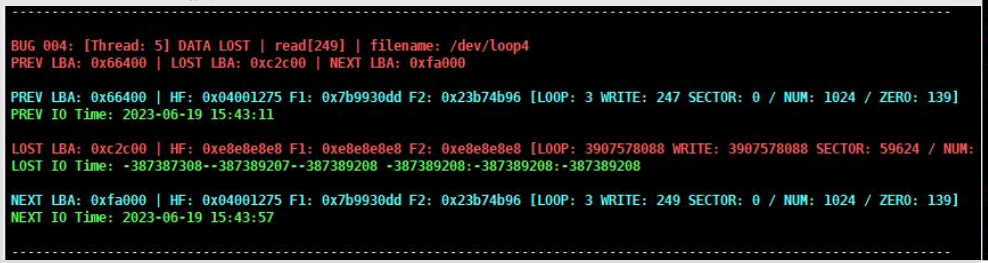
BUG 004: [Thread: 5] DATA LOST | read[249] | filename: /dev/loop4
PREV LBA: 0x66400 | LOST LBA: 0xc2c00 | NEXT LBA: 0xfa000

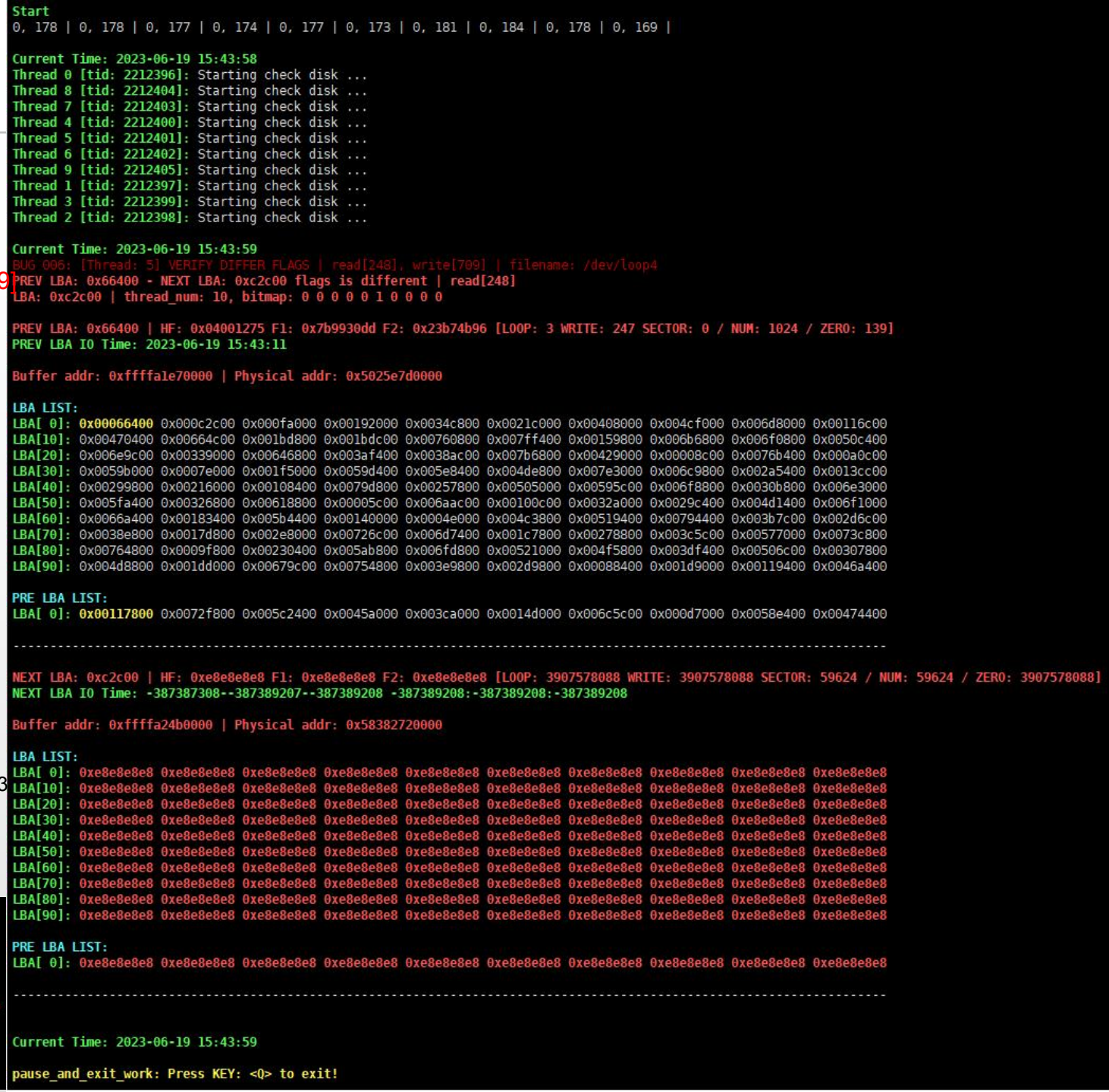
PREV LBA: 0x66400 | HF: 0x04001275 F1: 0x7b9930dd F2: 0x23b74b96 [LOOP: 3 WRITE: 247 SECTOR: 0 / NUM: 1024 / ZERO: 139]
PREV IO Time: 2023-06-19 15:43:11

LOST LBA: 0xc2c00 | HF: 0xe8e8e8e8 F1: 0xe8e8e8e8 F2: 0xe8e8e8e8 [LOOP: 3907578088 WRITE: 3907578088 SECTOR: 59624 / NUM: 59624 / ZERO: 3907578088]
LOST IO Time: -387387308--387389207--387389208 -387389208

NEXT LBA: 0xfa000 | HF: 0x04001275 F1: 0x7b9930dd F2: 0x23b74b96 [LOOP: 3 WRITE: 249 SECTOR: 0 / NUM: 1024 / ZERO: 139]
NEXT IO Time: 2023-06-19 15:43:57

---checktime校验





LBA工具基本功能演示

- BUG 007[1]：案例一，inject SECTOR lba bug

非零扇区：中间扇区的数据与前后扇区的数据不一致

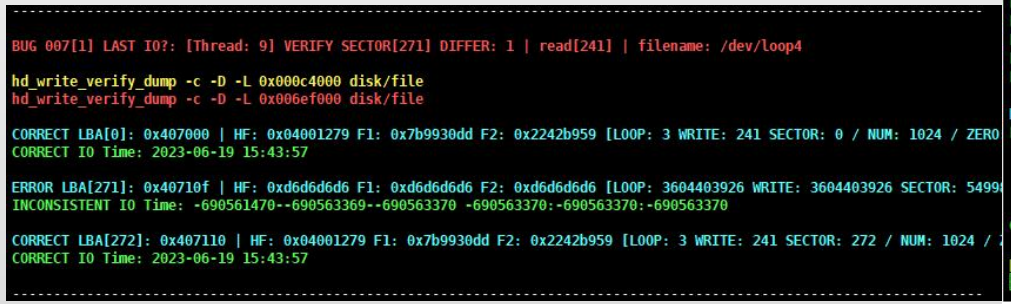
BUG 007[1] : [Thread: 9] VERIFY SECTOR[271] DIFFER: 1 | read[241] | filename: /dev/loop4

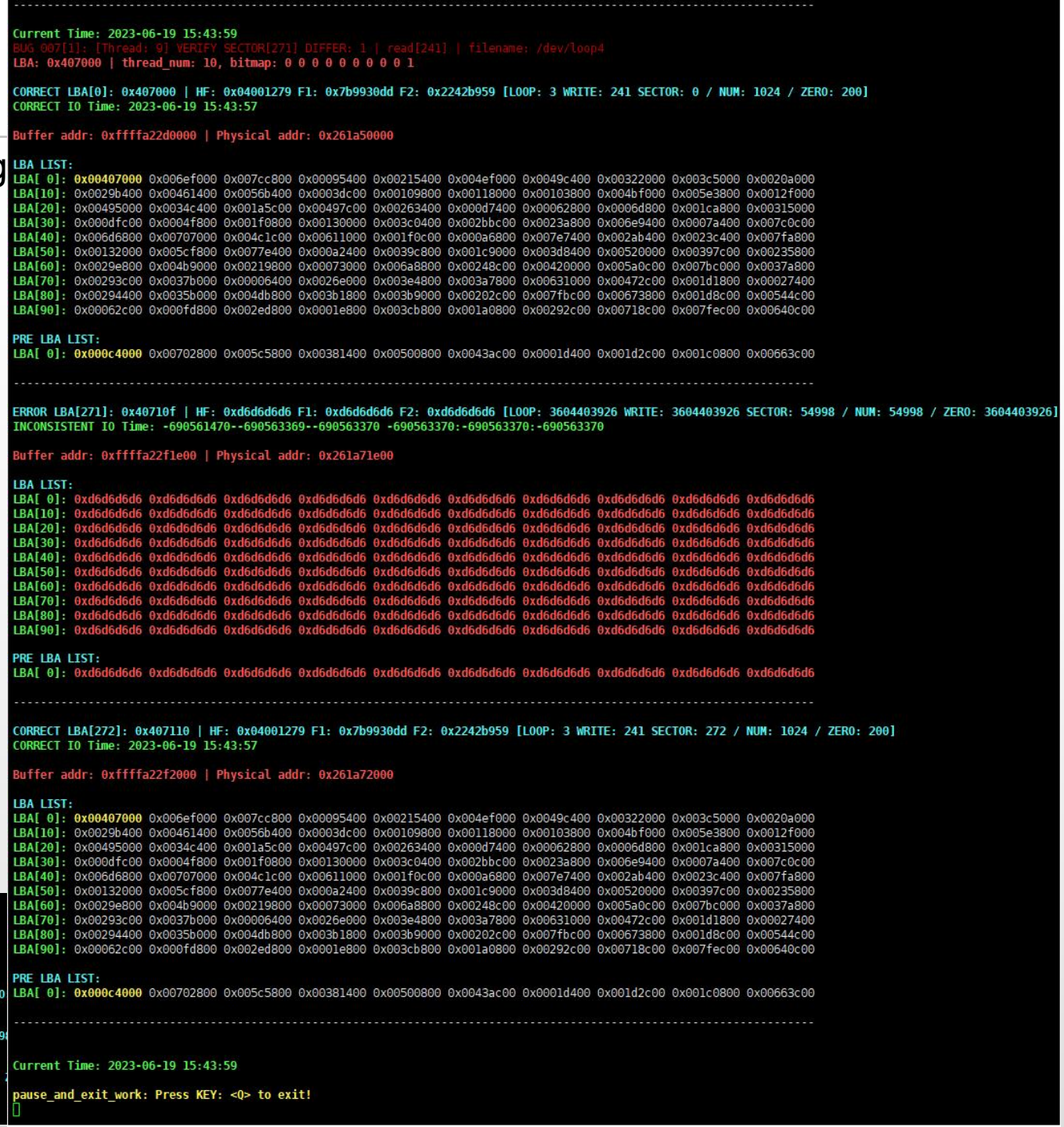
CORRECT LBA[0]: 0x407000 | HF: 0x04001279 F1: 0x7b9930dd F2: 0x2242b959 [LOOP: 3 WRITE: 241 SECTOR: 0 / NUM: 1024 / ZERO: 200]
CORRECT IO Time: 2023-06-19 15:43:57

ERROR LBA[271]: 0x40710f | HF: 0xd6d6d6d6 F1: 0xd6d6d6d6 F2: 0xd6d6d6d6 [LOOP: 3604403926 WRITE: 3604403926 SECTOR: 54998 / NUM: 54998 / ZERO: 3604403926]
INCONSISTENT IO Time: -690561470--690563369--690563370 -690563370:-690563370:-690563370

CORRECT LBA[272]: 0x407110 | HF: 0x04001279 F1: 0x7b9930dd F2: 0x2242b959 [LOOP: 3 WRITE: 241 SECTOR: 272 / NUM: 1024 / ZERO: 200]
CORRECT IO Time: 2023-06-19 15:43:57

---runtime校验 & checktime校验





LBA工具基本功能演示

- BUG 007[1]：案例二，inject LIST lba bug

非零扇区：中间扇区的数据与前后扇区的数据不一致

BUG 007[1] : [Thread: 6] VERIFY SECTOR[400] DIFFER: 1 | read[550] | filename: lba.raw

CORRECT LBA[0]: 0x762800 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 0 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

ERROR LBA[400]: 0x762990 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 400 / NUM: 1024 / ZERO: 120]
INCONSISTENT IO Time: 2023-06-25 16:22:21

CORRECT LBA[401]: 0x762991 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 401 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

---checktime校验

```
BUG 002[3]: [Thread: 6] VERIFY SECTOR[400] DIFFER: 1 | CORRECT LBA[0]: 0x762800 | ERROR LBA[400]: 0x762990 | filename: lba.raw
LBA: 0x762990 | thread_num: 10, bitmap: 0 0 0 0 0 0 1 0 0 0

LAST CORRECT WRITE:
LBA: 0x6f2400 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 564 SECTOR: 0 / NUM: 1024 / ZERO: 120]
LAST WRITE IO Time: 2023-06-25 16:22:21

Buffer addr: 0xfffeb14d0000 | Physical addr: 0x53a2320000
```

---runtime校验

```
BUG 007[1] LAST IO?: [Thread: 6] VERIFY SECTOR[400] DIFFER: 1 | read[550] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x00521800 disk/file
hd_write_verify_dump -c -D -L 0x006dec00 disk/file

CORRECT LBA[0]: 0x762800 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 0 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

ERROR LBA[400]: 0x762990 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 400 / NUM: 1024 / ZERO: 120]
INCONSISTENT IO Time: 2023-06-25 16:22:21

CORRECT LBA[401]: 0x762991 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 401 / NUM: 1024 / ZERO: 120]
CORRECT IO Time: 2023-06-25 16:22:21

----------------------------------------------------------------------------------------------------------------------------

LBA: 0x76298f | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 399 / NUM: 1024 / ZERO: 120]
IO Time: 2023-06-25 16:22:21

Buffer addr: 0xffff46081e00 | Physical addr: 0x5345cf1e00

LBA LIST:
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006f8c00 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000
LBA[20]: 0x006df400 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800
LBA[40]: 0x00107000 0x006f7400 0x002dcc00 0x00037800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x004ba800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00509400 0x00590000 0x002ad800 0x002cc400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00
LBA[90]: 0x002c4400 0x00386000 0x00179c00 0x00421400 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00

PRE LBA LIST:
LBA[ 0]: 0x00521800 0x006a9c00 0x00042c00 0x006c7000 0x00755000 0x00115400 0x005c4800 0x007bf800 0x002e6000 0x002d2400

----------------------------------------------------------------------------------------------------------------------------

LBA: 0x762990 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 400 / NUM: 1024 / ZERO: 120]
IO Time: 2023-06-25 16:22:21

Buffer addr: 0xffff46082000 | Physical addr: 0x5345cf2000

LBA LIST:
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006f8c00 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000
LBA[20]: 0xc6971fb4 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800
LBA[40]: 0x00107000 0x006f7400 0x002dcc00 0x00037800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x004ba800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00509400 0x00590000 0x002ad800 0x002cc400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00
LBA[90]: 0x002c4400 0x00386000 0x00179c00 0x00421400 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00

PRE LBA LIST:
LBA[ 0]: 0x00521800 0x006a9c00 0x00042c00 0x006c7000 0x00755000 0x00115400 0x005c4800 0x007bf800 0x002e6000 0x002d2400

----------------------------------------------------------------------------------------------------------------------------

LBA: 0x762991 | HF: 0x04001276 F1: 0x65bbb47d F2: 0xda812af0 [LOOP: 1 WRITE: 550 SECTOR: 401 / NUM: 1024 / ZERO: 120]
IO Time: 2023-06-25 16:22:21

Buffer addr: 0xffff46082200 | Physical addr: 0x5345cf2200

LBA LIST:
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006f8c00 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000
LBA[20]: 0x006df400 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800
LBA[40]: 0x00107000 0x006f7400 0x002dcc00 0x00037800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x004ba800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00509400 0x00590000 0x002ad800 0x002cc400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00
LBA[90]: 0x002c4400 0x00386000 0x00179c00 0x00421400 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00

PRE LBA LIST:
LBA[ 0]: 0x00521800 0x006a9c00 0x00042c00 0x006c7000 0x00755000 0x00115400 0x005c4800 0x007bf800 0x002e6000 0x002d2400

============================================================================================================================

Current Time: 2023-06-25 16:49:11
```

LBA工具基本功能演示

- BUG 007[2]：案例一，inject SECTOR lba bug

全零扇区：中间扇区的数据与前后扇区的数据不一致

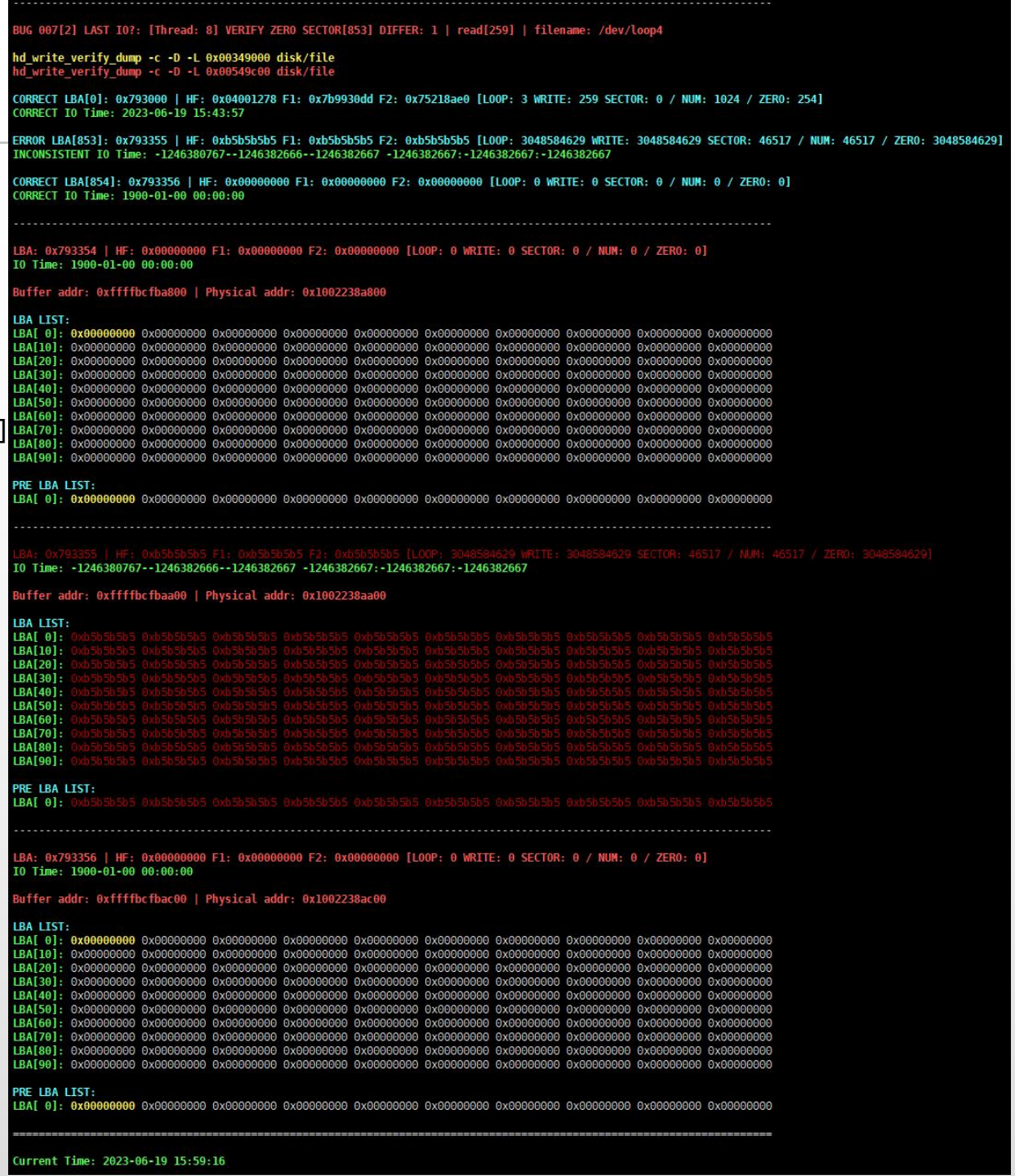
BUG 007[2] : [Thread: 8] VERIFY ZERO SECTOR[853] DIFFER: 1 | read[259] | filename: /dev/loop4

CORRECT LBA[0]: 0x793000 | HF: 0x04001278 F1: 0x7b9930dd F2: 0x75218ae0 [LOOP: 3 WRITE: 259 SECTOR: 0 / NUM: 1024 / ZERO: 254]
CORRECT IO Time: 2023-06-19 15:43:57

ERROR LBA[853]: 0x793355 | HF: 0xb5b5b5b5 F1: 0xb5b5b5b5 F2: 0xb5b5b5b5 [LOOP: 3048584629 WRITE: 3048584629 SECTOR: 46517 / NUM: 46517 / ZERO: 3048584629]
INCONSISTENT IO Time: -1246380767--1246382666--1246382667 -1246382667:-1246382667:-1246382667

CORRECT LBA[854]: 0x793356 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
CORRECT IO Time: 1900-01-00 00:00:00

---checktime校验



LBA工具基本功能演示

- BUG 007[2]：案例二，inject LIST lba bug

全零扇区：中间扇区的数据与前后扇区的数据不一致

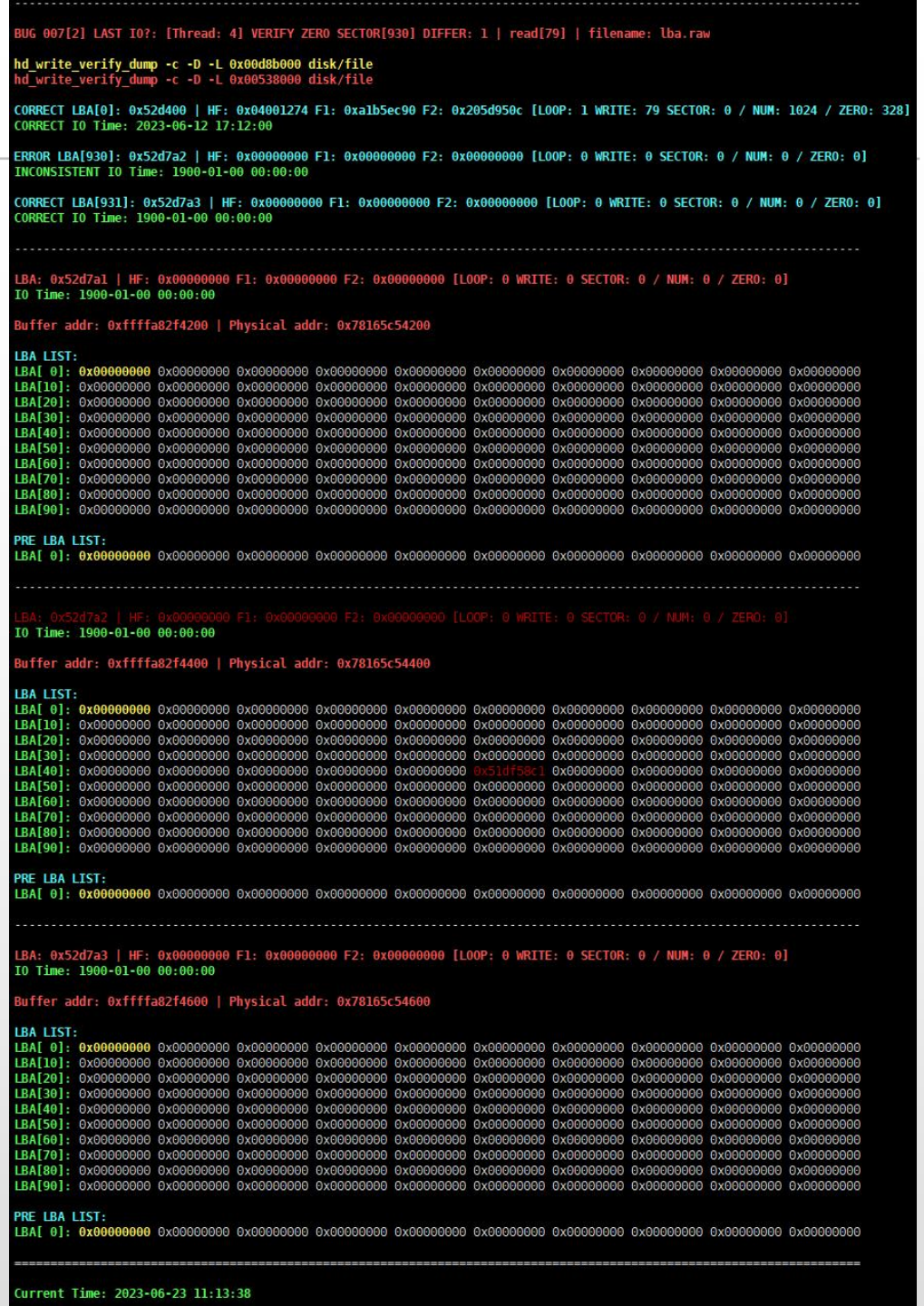
BUG 007[2] : [Thread: 4] VERIFY ZERO SECTOR[930] DIFFER: 1 | read[79] | filename: lba.raw

CORRECT LBA[0]: 0x52d400 | HF: 0x04001274 F1: 0xa1b5ec90 F2: 0x205d950c [LOOP: 1 WRITE: 79 SECTOR: 0 / NUM: 1024 / ZERO: 328]
CORRECT IO Time: 2023-06-12 17:12:00

ERROR LBA[930]: 0x52d7a2 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
INCONSISTENT IO Time: 1900-01-00 00:00:00

CORRECT LBA[931]: 0x52d7a3 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
CORRECT IO Time: 1900-01-00 00:00:00

---runtime校验 & checktime校验



LBA工具基本功能演示

- BUG 007[3]：inject SECTOR lba bug

前后簇数据正确，中间簇非零扇区数据校验出错

BUG 007[3]: [Thread: 7] VERIFY SECTOR[310] DIFFER: 11 | read[498] | filename: lba.raw

CORRECT LBA[0]: 0x5ad000 | HF: 0x04001277 F1: 0xc079f392 F2: 0x662da7b8 [LOOP: 1 WRITE: 498 SECTOR: 0 / NUM: 1024 / ZERO: 224]
CORRECT IO Time: 2023-06-23 00:02:28

ERROR LBA[310]: 0x5ad136 | HF: 0x19191919 F1: 0x19191919 F2: 0x19191919 [LOOP: 421075225 WRITE: 421075225 SECTOR: 6425 / NUM: 6425 / ZERO: 421075225]
INCONSISTENT IO Time: 421077125-421075226-421075225 421075225:421075225:421075225

CORRECT LBA[321]: 0x5ad141 | HF: 0x04001277 F1: 0xc079f392 F2: 0x662da7b8 [LOOP: 1 WRITE: 498 SECTOR: 321 / NUM: 1024 / ZERO: 224]
CORRECT IO Time: 2023-06-23 00:02:28

---checktime校验

```
Current Time: 2023-06-23 11:01:27
BUG 007[3]: [Thread: 7] VERIFY SECTOR[310] DIFFER: 11 | read[498] | filename: lba.raw
LBA: 0x5ad000 | thread_num: 10, bitmap: 0 0 0 0 0 0 0 0 0 0

CORRECT LBA[0]: 0x5ad000 | HF: 0x04001277 F1: 0xc079f392 F2: 0x662da7b8 [LOOP: 1 WRITE: 498 SECTOR: 0 / NUM: 1024 / ZERO: 224]
CORRECT IO Time: 2023-06-23 00:02:28

Buffer addr: 0xffff024d0000 | Physical addr: 0x420fb10000

LBA LIST:
LBA[ 0]: 0x005ad000 0x0060cc00 0x002f1000 0x00495000 0x0012f000 0x00459c00 0x0012f400 0x00154800 0x00366c00 0x00175800
LBA[10]: 0x0075e800 0x003b1c00 0x0008e800 0x005a6400 0x00099c00 0x00550400 0x00632c00 0x000e1c00 0x0055d800 0x00136000
LBA[20]: 0x00302c00 0x007cbc00 0x00073800 0x00058400 0x00491000 0x003c4c00 0x00787400 0x0067ec00 0x00359800 0x006e7400
LBA[30]: 0x00687800 0x001f5c00 0x004b5800 0x002e3400 0x0073a400 0x0075f400 0x0044cc00 0x007f1000 0x00794800 0x005f9400
LBA[40]: 0x002c9800 0x00505800 0x003d5c00 0x00445000 0x004e9c00 0x00420400 0x00526800 0x001e7800 0x0031bc00 0x0001cc00
LBA[50]: 0x0030b400 0x0052e800 0x001e1000 0x006dd000 0x00539800 0x00422c00 0x00696c00 0x006a0400 0x00542400 0x002b4400
LBA[60]: 0x005be000 0x00289c00 0x00650000 0x000f3400 0x004ca800 0x005f1000 0x00748400 0x00603400 0x005fd400 0x004b0c00
LBA[70]: 0x007d0400 0x00743c00 0x000b2800 0x00419800 0x0076a800 0x00307800 0x0056d800 0x00142c00 0x0047b400 0x007af800
LBA[80]: 0x006cc400 0x003d4800 0x00127c00 0x00585400 0x001a9800 0x004c4c00 0x007de000 0x003b7000 0x0059a400 0x006da000
LBA[90]: 0x00176400 0x00223c00 0x000ddc00 0x00159400 0x002ac800 0x006c4400 0x001e0400 0x006c6800 0x002e9000 0x000d4400

PRE LBA LIST:
LBA[ 0]: 0x005da400 0x0060e800 0x00092400 0x0051f800 0x00433800 0x00289000 0x005b3c00 0x00128c00 0x0022b000 0x00342c00

ERROR LBA[310]: 0x5ad136 | HF: 0x19191919 F1: 0x19191919 F2: 0x19191919 [LOOP: 421075225 WRITE: 421075225 SECTOR: 6425 / NUM: 6425 / ZERO: 421075225]
INCONSISTENT IO Time: 421077125-421075226-421075225 421075225:421075225:421075225

Buffer addr: 0xffff024f6c00 | Physical addr: 0x4490226c00

LBA LIST:
LBA[ 0]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[10]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[20]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[30]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[40]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[50]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[60]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[70]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[80]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919
LBA[90]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919

PRE LBA LIST:
LBA[ 0]: 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919 0x19191919

NEXT LBA: 0x60cc00 | HF: 0x04001277 F1: 0xc079f392 F2: 0x662da7b8 [LOOP: 1 WRITE: 499 SECTOR: 0 / NUM: 1024 / ZERO: 224]
NEXT IO Time: 2023-06-23 00:02:28

Buffer addr: 0xffff018f0000 | Physical addr: 0x48a45f0000

LBA LIST:
LBA[ 0]: 0x0060cc00 0x002f1000 0x00495000 0x0012f000 0x00459c00 0x0012f400 0x00154800 0x00366c00 0x00175800 0x0075e800
LBA[10]: 0x003b1c00 0x0008e800 0x005a6400 0x00099c00 0x00550400 0x00632c00 0x000e1c00 0x0055d800 0x00136000 0x00302c00
LBA[20]: 0x007cbc00 0x00073800 0x00058400 0x00491000 0x003c4c00 0x00787400 0x0067ec00 0x00359800 0x006e7400 0x00687800
LBA[30]: 0x001f5c00 0x004b5800 0x002e3400 0x0073a400 0x0075f400 0x0044cc00 0x007f1000 0x00794800 0x005f9400 0x002c9800
LBA[40]: 0x00505800 0x003d5c00 0x00445000 0x004e9c00 0x00420400 0x00526800 0x001e7800 0x0031bc00 0x0001cc00 0x0030b400
LBA[50]: 0x0052e800 0x001e1000 0x006dd000 0x00539800 0x00422c00 0x00696c00 0x006a0400 0x00542400 0x002b4400 0x005be000
LBA[60]: 0x00289c00 0x00650000 0x000f3400 0x004ca800 0x005f1000 0x00748400 0x00603400 0x005fd400 0x004b0c00 0x007d0400
LBA[70]: 0x00743c00 0x000b2800 0x00419800 0x0076a800 0x00307800 0x0056d800 0x00142c00 0x0047b400 0x007af800 0x006cc400
LBA[80]: 0x003d4800 0x00127c00 0x00585400 0x001a9800 0x004c4c00 0x007de000 0x003b7000 0x0059a400 0x006da000 0x00176400
LBA[90]: 0x00223c00 0x000ddc00 0x00159400 0x002ac800 0x006c4400 0x001e0400 0x006c6800 0x002e9000 0x000d4400 0x0006a400

PRE LBA LIST:
LBA[ 0]: 0x005ad000 0x005da400 0x0060e800 0x00092400 0x0051f800 0x00433800 0x00289000 0x005b3c00 0x00128c00 0x0022b000
```

LBA工具基本功能演示

- BUG 007[4]：inject SECTOR lba bug

前后簇数据正确，中间簇全零扇区数据校验出错

BUG 007[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54 | read[534] | filename: lba.raw

CORRECT LBA[0]: 0x7e9800 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 533 SECTOR: 0 / NUM: 1024 / ZERO: 271]
CORRECT IO Time: 2023-06-23 00:02:29

ERROR LBA[970]: 0x7e9bca | HF: 0xb4b4b4b4 F1: 0xb4b4b4b4 F2: 0xb4b4b4b4 [LOOP: 3031741620 WRITE: 3031741620 SECTOR: 46260 / NUM: 46260 / ZERO: 3031741620]
INCONSISTENT IO Time: -1263223776--1263225675--1263225676 -1263225676:-1263225676:-1263225676

NEXT LBA: 0x60b000 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 534 SECTOR: 0 / NUM: 1024 / ZERO: 271]
NEXT IO Time: 2023-06-23 00:02:29

---checktime校验

```
BUG 007[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54 | read[534] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x006e5c00 disk/file
hd_write_verify_dump -c -D -L 0x0060b000 disk/file

CORRECT LBA[0]: 0x7e9800 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 533 SECTOR: 0 / NUM: 1024 / ZERO: 271]
CORRECT IO Time: 2023-06-23 00:02:29

ERROR LBA[970]: 0x7e9bca | HF: 0xb4b4b4b4 F1: 0xb4b4b4b4 F2: 0xb4b4b4b4 [LOOP: 3031741620 WRITE: 3031741620 SECTOR: 46260 / NUM: 46260 / ZERO: 3031741620]
INCONSISTENT IO Time: -1263223776--1263225675--1263225676 -1263225676:-1263225676:-1263225676

NEXT LBA: 0x60b000 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 534 SECTOR: 0 / NUM: 1024 / ZERO: 271]
NEXT IO Time: 2023-06-23 00:02:29

----------------------------------------------------------------------------------------------------------------------------

LBA: 0x7e9bc9 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
IO Time: 1900-01-00 00:00:00

Buffer addr: 0xffff55489200 | Physical addr: 0x223be19200

LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

----------------------------------------------------------------------------------------------------------------------------

LBA: 0x7e9bca | HF: 0xb4b4b4b4 F1: 0xb4b4b4b4 F2: 0xb4b4b4b4 [LOOP: 3031741620 WRITE: 3031741620 SECTOR: 46260 / NUM: 46260 / ZERO: 3031741620]
IO Time: -1263223776--1263225675--1263225676 -1263225676:-1263225676:-1263225676

Buffer addr: 0xffff55489400 | Physical addr: 0x223be19400

LBA LIST:
LBA[ 0]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[10]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[20]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[30]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[40]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[50]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[60]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[70]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[80]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
LBA[90]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4

PRE LBA LIST:
LBA[ 0]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4

----------------------------------------------------------------------------------------------------------------------------

LBA: 0x60b000 | HF: 0x04001271 F1: 0xc079f392 F2: 0x8d10bb23 [LOOP: 1 WRITE: 534 SECTOR: 0 / NUM: 1024 / ZERO: 271]
IO Time: 2023-06-23 00:02:29

Buffer addr: 0xffff552d0000 | Physical addr: 0x21a2df0000

LBA LIST:
LBA[ 0]: 0x0060b000 0x003ee400 0x007af400 0x0019d400 0x00648400 0x004ffc00 0x0021c800 0x002dbc00 0x005c8800 0x00155000
LBA[10]: 0x00247c00 0x00639c00 0x00041800 0x000bd800 0x0054fc00 0x000b2000 0x00717400 0x00697800 0x00560000 0x00343000
LBA[20]: 0x00621400 0x003e8400 0x00010c00 0x002a8800 0x0066e000 0x0037ec00 0x00317400 0x00769000 0x00738400 0x000d3400
LBA[30]: 0x005d4000 0x00444800 0x00510c00 0x005c4400 0x007b1800 0x004de400 0x004ab400 0x007c5400 0x007a5800 0x003b3c00
LBA[40]: 0x000e4000 0x0075fc00 0x0026f800 0x0005a800 0x001a9400 0x0028ac00 0x00254000 0x00422800 0x002a3000 0x0016bc00
LBA[50]: 0x006f9400 0x00048c00 0x004a7000 0x000ba000 0x005ed000 0x00772800 0x00793c00 0x00415400 0x006b8800 0x00095c00
LBA[60]: 0x004ef000 0x006dd400 0x00220c00 0x0012e800 0x000bac00 0x0033b000 0x00210400 0x0028cc00 0x000af400 0x0066e800
LBA[70]: 0x0051d800 0x00170c00 0x00713800 0x0049dc00 0x00735800 0x002ba000 0x001f8000 0x001bd400 0x000d3800 0x00505c00
LBA[80]: 0x00133400 0x0055a000 0x0009a000 0x0022f400 0x001dc400 0x006c0400 0x00485800 0x00088400 0x007bec00 0x0053ec00
LBA[90]: 0x00066c00 0x004b9800 0x003f6000 0x00295800 0x00655400 0x00565400 0x0028bc00 0x00539400 0x000d4800 0x003cf800

PRE LBA LIST:
LBA[ 0]: 0x007e9800 0x006e5c00 0x005de400 0x006ad400 0x00119800 0x002d3c00 0x0013f000 0x0006a800 0x006d4800 0x00288400

============================================================================================================================

Current Time: 2023-06-23 11:23:18
```

LBA工具基本功能演示：hd_write_verify_dump工具数据校验功能

- 线程数据校验：(-T参数)

(1) 校验2号线程写入的所有数据，第85簇IO与前面簇的数据不一致：BUG 003

(2) 校验2号线程写入的前30簇的数据，所有数据是一致的。

```
[root@localhost lba]# hd_write_verify_dump -c -D -T 2 lba.raw
File Information:
        File:       lba.raw
        Size:       10737418240
        Blocks:     965680
        IO Block:   65536
        Device:     2051
        Inode:      1678094440
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-06-24 12:29:30

================================================================================================================

BUG 003: [Thread: 2] WRITE PART SECTOR | read[85] | filename: lba.raw

CORRECT LBA: 0x11de000 | HF: 0x04001272 F1: 0xa1b5ec90 F2: 0x27c3ef0a [LOOP: 1 WRITE: 84 SECTOR: 0 / NUM: 1024 / ZERO: 246]
CORRECT LBA IO Time: 2023-06-12 17:11:59

ERROR LBA: 0x95d000 | HF: 0x04001272 F1: 0xa1b5ec90 F2: 0x27c3ef0a [LOOP: 1 WRITE: 85 SECTOR: 0 / NUM: 1024 / ZERO: 246]
ERROR LBA IO Time: 2023-06-12 17:12:00

----------------------------------------------------------------------------------------------------------------

LBA: 0x11de000 | HF: 0x04001272 F1: 0xa1b5ec90 F2: 0x27c3ef0a [LOOP: 1 WRITE: 84 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:11:59

Buffer addr: 0xffff92050000 | Physical addr: 0x7024d560000

LBA LIST:
LBA[ 0]: 0x011de000 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000 0x00fb5000 0x011c0c00 0x00d49000 0x00898000
LBA[10]: 0x00bb6c00 0x001cc000 0x00f46400 0x00013c00 0x00192800 0x00e60400 0x0015a800 0x0094c000 0x00db6000 0x00f92c00
LBA[20]: 0x003ec000 0x0081d000 0x000ca800 0x0032a000 0x008d7800 0x010f4800 0x002a8000 0x00ca1000 0x011c4000 0x000cb000
LBA[30]: 0x007e8800 0x001f4000 0x00872400 0x010ce000 0x008c6800 0x011e5000 0x00664000 0x00e34c00 0x00ab5800 0x00768400
LBA[40]: 0x00d77000 0x00901c00 0x00e16000 0x00499000 0x00a80000 0x0007e800 0x00c2a000 0x00217800 0x0074f800 0x00a67800
LBA[50]: 0x00847000 0x012a0400 0x006fb800 0x004c1400 0x004e1000 0x0055cc00 0x00976000 0x00118000 0x010e5c00 0x00370000
LBA[60]: 0x001b1000 0x00f40000 0x000e8c00 0x00f2a000 0x00638000 0x00780000 0x008f6400 0x008c1c00 0x00d08800 0x002a3800
LBA[70]: 0x011afc00 0x00a62000 0x00cf4400 0x005b4000 0x013ed800 0x01195800 0x008c8400 0x00509c00 0x00513800 0x00e16400
LBA[80]: 0x007c9400 0x00462000 0x00088800 0x0095c000 0x00d4bc00 0x0008e000 0x00a9d000 0x01038000 0x00b50000 0x00616400
LBA[90]: 0x00b28000 0x00807800 0x00ff1000 0x011e9000 0x00c61400 0x0135bc00 0x013e2000 0x00316400 0x00c76000 0x00b62400

PRE LBA LIST:
LBA[ 0]: 0x00a30000 0x001cc400 0x00417000 0x00e5c000 0x0105c400 0x01138000 0x01010400 0x0052f000 0x009fa400 0x00ebb000

----------------------------------------------------------------------------------------------------------------

LBA: 0x95d000 | HF: 0x04001272 F1: 0xa1b5ec90 F2: 0x27c3ef0a [LOOP: 1 WRITE: 85 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xffff920f0000 | Physical addr: 0x782b6920000

LBA LIST:
LBA[ 0]: 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000 0x00fb5000 0x011c0c00 0x00d49000 0x00898000 0x00bb6c00
LBA[10]: 0x001cc000 0x00f46400 0x00013c00 0x00192800 0x00e60400 0x0015a800 0x0094c000 0x00db6000 0x00f92c00 0x003ec000
LBA[20]: 0x0081d000 0x000ca800 0x0032a000 0x008d7800 0x010f4800 0x002a8000 0x00ca1000 0x011c4000 0x000cb000 0x007e8800
LBA[30]: 0x001f4000 0x00872400 0x010ce000 0x008c6800 0x011e5000 0x00664000 0x00e34c00 0x00ab5800 0x00768400 0x00d77000
LBA[40]: 0x00901c00 0x00e16000 0x00499000 0x00a80000 0x0007e800 0x00c2a000 0x00217800 0x0074f800 0x00a67800 0x00847000
LBA[50]: 0x012a0400 0x006fb800 0x004c1400 0x004e1000 0x0055cc00 0x00976000 0x00118000 0x010e5c00 0xcld4b822 0x001b1000
LBA[60]: 0x00f40000 0x000e8c00 0x00f2a000 0x00638000 0x00780000 0x008f6400 0x008c1c00 0x00d08800 0x002a3800 0x011afc00
LBA[70]: 0x00a62000 0x00cf4400 0x005b4000 0x013ed800 0x01195800 0x008c8400 0x00509c00 0x00513800 0x00e16400 0x007c9400
LBA[80]: 0x00462000 0x00088800 0x0095c000 0x00d4bc00 0x0008e000 0x00a9d000 0x01038000 0x00b50000 0x00616400 0x00b28000
LBA[90]: 0x00807800 0x00ff1000 0x011e9000 0x00c61400 0x0135bc00 0x013e2000 0x00316400 0x00c76000 0x00b62400 0x0005a400

PRE LBA LIST:
LBA[ 0]: 0x011de000 0x00a30000 0x001cc400 0x00417000 0x00e5c000 0x0105c400 0x01138000 0x01010400 0x0052f000 0x009fa400

================================================================================================================

Current Time: 2023-06-24 12:29:31
```

```
[root@localhost lba]# hd_write_verify_dump -c -D -T 2 -C 30 lba.raw
File Information:
        File:       lba.raw
        Size:       10737418240
        Blocks:     965680
        IO Block:   65536
        Device:     2051
        Inode:      1678094440
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-06-24 12:30:11

================================================================================================================

LBA: 0x800 | HF: 0x04001272 F1: 0xa1b5ec90 F2: 0x27c3ef0a [LOOP: 1 WRITE: 1 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:11:00

Buffer addr: 0x26c20000 | Physical addr: 0x10195ad0000

LBA LIST:
LBA[ 0]: 0x00000800 0x0022a000 0x00efcc00 0x01303000 0x01264000 0x00461c00 0x0034a400 0x010ad000 0x003a2c00 0x0035bc00
LBA[10]: 0x00eb7c00 0x00144000 0x00406800 0x00522000 0x00914000 0x00611800 0x00c78000 0x005f7800 0x00493000 0x00f9bc00
LBA[20]: 0x00395400 0x010fe000 0x0016d000 0x0007fc00 0x00d00000 0x008b0800 0x00be6400 0x001bf000 0x00d10800 0x01168000
LBA[30]: 0x01029000 0x00c74000 0x00b1e400 0x004a0000 0x00cf0c00 0x00a80400 0x00c22000 0x00350000 0x00cf6000 0x01295000
LBA[40]: 0x007fb800 0x00291c00 0x003b7800 0x010f8000 0x00d52000 0x00bca000 0x00534000 0x002ff000 0x006c8000 0x011a1400
LBA[50]: 0x0077b400 0x0001d000 0x012a5800 0x0046f000 0x00e62000 0x009d3c00 0x00aa0000 0x000d3000 0x008eb800 0x000f1800
LBA[60]: 0x011fa000 0x003dc800 0x0007a000 0x008ff000 0x0107c400 0x01100000 0x00f5f000 0x00ccdc00 0x00ac7000 0x00f31000
LBA[70]: 0x00891400 0x004bc000 0x00ba1c00 0x00ebb000 0x009fa400 0x0052f000 0x01010400 0x01138000 0x0105c400 0x00e5c000
LBA[80]: 0x00417000 0x001cc400 0x00a30000 0x011de000 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000 0x00fb5000
LBA[90]: 0x011c0c00 0x00d49000 0x00898000 0x00bb6c00 0x001cc000 0x00f46400 0x00013c00 0x00192800 0x00e60400 0x0015a800

PRE LBA LIST:
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

----------------------------------------------------------------------------------------------------------------

LBA: 0x1168000 | HF: 0x04001272 F1: 0xa1b5ec90 F2: 0x27c3ef0a [LOOP: 1 WRITE: 30 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:11:18

Buffer addr: 0xffffa4a20000 | Physical addr: 0x1032c530000

LBA LIST:
LBA[ 0]: 0x01168000 0x01029000 0x00c74000 0x00b1e400 0x004a0000 0x00cf0c00 0x00a80400 0x00c22000 0x00350000 0x00cf6000
LBA[10]: 0x01295000 0x007fb800 0x00291c00 0x003b7800 0x010f8000 0x00d52000 0x00bca000 0x00534000 0x002ff000 0x006c8000
LBA[20]: 0x011a1400 0x0077b400 0x0001d000 0x012a5800 0x0046f000 0x00e62000 0x009d3c00 0x00aa0000 0x000d3000 0x008eb800
LBA[30]: 0x000f1800 0x011fa000 0x003dc800 0x0007a000 0x008ff000 0x0107c400 0x01100000 0x00f5f000 0x00ccdc00 0x00ac7000
LBA[40]: 0x00f31000 0x00891400 0x004bc000 0x00ba1c00 0x00ebb000 0x009fa400 0x0052f000 0x01010400 0x01138000 0x0105c400
LBA[50]: 0x00e5c000 0x00417000 0x001cc400 0x00a30000 0x011de000 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000
LBA[60]: 0x00fb5000 0x011c0c00 0x00d49000 0x00898000 0x00bb6c00 0x001cc000 0x00f46400 0x00013c00 0x00192800 0x00e60400
LBA[70]: 0x0015a800 0x0094c000 0x00db6000 0x00f92c00 0x003ec000 0x0081d000 0x000ca800 0x0032a000 0x008d7800 0x010f4800
LBA[80]: 0x002a8000 0x00ca1000 0x011c4000 0x000cb000 0x007e8800 0x001f4000 0x00872400 0x010ce000 0x008c6800 0x011e5000
LBA[90]: 0x00664000 0x00e34c00 0x00ab5800 0x00768400 0x00d77000 0x00901c00 0x00e16000 0x00499000 0x00a80000 0x0007e800

PRE LBA LIST:
LBA[ 0]: 0x00d10800 0x001bf000 0x00be6400 0x008b0800 0x00d00000 0x0007fc00 0x0016d000 0x010fe000 0x00395400 0x00f9bc00

================================================================================================================

Current Time: 2023-06-24 12:30:11
```

LBA工具基本功能演示：hd_write_verify_dump工具数据校验功能

- 簇间数据校验：(-L和-C参数)

(1) 校验LBA地址：0x63000开始的之后30簇数据，所有数据是一致的。

(2) 校验LBA地址：0x63000开始的之后所有数据，第82簇IO，簇内偏移234扇区的数据与前后扇区的数据不一致：BUG 007[1]

```
[root@localhost lba]# hd_write_verify_dump -c -D -L 0x63000 -C 50  lba.raw
File Information:
        File:       lba.raw
        Size:       10737418240
        Blocks:     965680
        IO Block:   65536
        Device:     2051
        Inode:      1678094440
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-06-24 13:06:25

================================================================================================================

LBA: 0x63000 | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 21 SECTOR: 0 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:11:12

Buffer addr: 0x14160000 | Physical addr: 0x10076980000

LBA LIST:
LBA[ 0]: 0x00063000 0x013ffc00 0x00c98000 0x0060d800 0x01329c00 0x00da4800 0x00e28400 0x0127c800 0x013bac00 0x00afdc00
LBA[10]: 0x008c8000 0x013bc800 0x00f84c00 0x003b2000 0x003eb400 0x00b05c00 0x004af000 0x0096f800 0x01121800 0x00b7ac00
LBA[20]: 0x00e9d000 0x00160000 0x00483800 0x00c6d000 0x0054d800 0x0058c000 0x00b02000 0x00be0000 0x00690000 0x00314800
LBA[30]: 0x000f7000 0x00e9b000 0x0057e000 0x003df400 0x00f95000 0x00b21800 0x004b2400 0x00acb800 0x00ceb800 0x01057000
LBA[40]: 0x00e00000 0x00804000 0x00c4f800 0x00e20000 0x0114b800 0x00492000 0x0105b000 0x00266800 0x0012cc00 0x008e6400
LBA[50]: 0x006c5c00 0x012ab800 0x010c0800 0x0091c000 0x00f63000 0x005a4400 0x00d40400 0x00b11800 0x00ae1000 0x00673000
LBA[60]: 0x00a05800 0x00f24000 0x00dc4c00 0x0020bc00 0x01341400 0x00ae6800 0x009c6400 0x00fe4000 0x00983800 0x004e2800
LBA[70]: 0x00661800 0x010d4000 0x01298800 0x00233000 0x0037f800 0x013ab800 0x005c3800 0x003e7c00 0x013a6800 0x00e3e000
LBA[80]: 0x006fc000 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x00bec800
LBA[90]: 0x00e69800 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400 0x00d69000

PRE LBA LIST:
LBA[ 0]: 0x0134d000 0x004d0000 0x008d6000 0x0093e000 0x01134000 0x004f0000 0x001d7c00 0x00822800 0x01392000 0x00122000

----------------------------------------------------------------------------------------------------------------

LBA: 0x8e6400 | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 70 SECTOR: 0 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:11:55

Buffer addr: 0xffffb0740000 | Physical addr: 0x100769f0000

LBA LIST:
LBA[ 0]: 0x008e6400 0x006c5c00 0x012ab800 0x010c0800 0x0091c000 0x00f63000 0x005a4400 0x00d40400 0x00b11800 0x00ae1000
LBA[10]: 0x00673000 0x00a05800 0x00f24000 0x00dc4c00 0x0020bc00 0x01341400 0x00ae6800 0x009c6400 0x00fe4000 0x00983800
LBA[20]: 0x004e2800 0x00661800 0x010d4000 0x01298800 0x00233000 0x0037f800 0x013ab800 0x005c3800 0x003e7c00 0x013a6800
LBA[30]: 0x00e3e000 0x006fc000 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000
LBA[40]: 0x00bec800 0x00e69800 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400
LBA[50]: 0x00d69000 0x00fc7800 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00b89800 0x0101bc00 0x00207000 0x00430000
LBA[60]: 0x00c11000 0x00d5d000 0x0057a800 0x00bd3400 0x00753c00 0x007c4c00 0x00ed6800 0x00933000 0x00d77400 0x00258400
LBA[70]: 0x00b7e000 0x00b44000 0x008e2800 0x00958800 0x004f1800 0x00128000 0x01310000 0x0042c000 0x000aac00 0x00539800
LBA[80]: 0x006c4000 0x00375000 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000
LBA[90]: 0x0085c000 0x00e5bc00 0x0025f400 0x00af5800 0x00ca3800 0x00f1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800

PRE LBA LIST:
LBA[ 0]: 0x0012cc00 0x00266800 0x0105b000 0x00492000 0x0114b800 0x00e20000 0x00c4f800 0x00804000 0x00e00000 0x01057000

================================================================================================================

Current Time: 2023-06-24 13:06:25
```

```
[root@localhost lba]# hd_write_verify_dump -c -D -L 0x63000 -C 0  lba.raw
File Information:
        File:       lba.raw
        Size:       10737418240
        Blocks:     965680
        IO Block:   65536
        Device:     2051
        Inode:      1678094440
        Links:      1
        File Size: 10737418240 / 10240M / 10.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-06-24 13:06:17

================================================================================================================

BUG 007[1] LAST IO?: [Thread: 5] VERIFY SECTOR[234] DIFFER: 1 | read[62] | filename: lba.raw

hd_write_verify_dump -c -D -L 0x00a05800 disk/file
hd_write_verify_dump -c -D -L 0x00dc4c00 disk/file

CORRECT LBA[0]: 0xf24000 | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 82 SECTOR: 0 / NUM: 1024 / ZERO: 208]
CORRECT IO Time: 2023-06-12 17:12:00

ERROR LBA[234]: 0xf240ea | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 82 SECTOR: 234 / NUM: 1024 / ZERO: 208]
INCONSISTENT IO Time: 2023-06-12 17:12:00

CORRECT LBA[235]: 0xf240eb | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 82 SECTOR: 235 / NUM: 1024 / ZERO: 208]
CORRECT IO Time: 2023-06-12 17:12:00

----------------------------------------------------------------------------------------------------------------

LBA: 0xf240e9 | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 82 SECTOR: 233 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xffff9327d200 | Physical addr: 0x30325e1d200

LBA LIST:
LBA[ 0]: 0x00f24000 0x00dc4c00 0x0020bc00 0x01341400 0x00ae6800 0x009c6400 0x00fe4000 0x00983800 0x004e2800 0x00661800
LBA[10]: 0x010d4000 0x01298800 0x00233000 0x0037f800 0x013ab800 0x005c3800 0x003e7c00 0x013a6800 0x00e3e000 0x006fc000
LBA[20]: 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x00bec800 0x00e69800
LBA[30]: 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400 0x00d69000 0x00fc7800
LBA[40]: 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00b89800 0x0101bc00 0x00207000 0x00430000 0x00c11000 0x00d5d000
LBA[50]: 0x0057a800 0x00bd3400 0x00753c00 0x007c4c00 0x00ed6800 0x00933000 0x00d77400 0x00258400 0x00b7e000 0x00b44000
LBA[60]: 0x008e2800 0x00958800 0x004f1800 0x00128000 0x01310000 0x0042c000 0x000aac00 0x00539800 0x006c4000 0x00375000
LBA[70]: 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000 0x0085c000 0x00e5bc00
LBA[80]: 0x0025f400 0x00af5800 0x00ca3800 0x00f1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800 0x01083800 0x00ec2800
LBA[90]: 0x009c5800 0x000ba000 0x007aac00 0x005b7c00 0x00478000 0x00476000 0x011f6000 0x0138c000 0x00451000 0x00f91c00

PRE LBA LIST:
LBA[ 0]: 0x00a05800 0x00673000 0x00ae1000 0x00b11800 0x00d40400 0x005a4400 0x00f63000 0x0091c000 0x010c0800 0x012ab800

----------------------------------------------------------------------------------------------------------------

LBA: 0xf240ea | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 82 SECTOR: 234 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xffff9327d400 | Physical addr: 0x30325e1d400

LBA LIST:
LBA[ 0]: 0x00f24000 0x00dc4c00 0x0020bc00 0x01341400 0x00ae6800 0x009c6400 0x00fe4000 0x00983800 0x004e2800 0x00661800
LBA[10]: 0x010d4000 0x01298800 0x00233000 0x0037f800 0x013ab800 0x005c3800 0x003e7c00 0x013a6800 0x00e3e000 0x006fc000
LBA[20]: 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x00bec800 0x00e69800
LBA[30]: 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400 0x00d69000 0x00fc7800
LBA[40]: 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00e9c2c 0x0101bc00 0x00207000 0x00430000 0x00c11000 0x00d5d000
LBA[50]: 0x0057a800 0x00bd3400 0x00753c00 0x007c4c00 0x00ed6800 0x00933000 0x00d77400 0x00258400 0x00b7e000 0x00b44000
LBA[60]: 0x008e2800 0x00958800 0x004f1800 0x00128000 0x01310000 0x0042c000 0x000aac00 0x00539800 0x006c4000 0x00375000
LBA[70]: 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000 0x0085c000 0x00e5bc00
LBA[80]: 0x0025f400 0x00af5800 0x00ca3800 0x00f1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800 0x01083800 0x00ec2800
LBA[90]: 0x009c5800 0x000ba000 0x007aac00 0x005b7c00 0x00478000 0x00476000 0x011f6000 0x0138c000 0x00451000 0x00f91c00

PRE LBA LIST:
LBA[ 0]: 0x00a05800 0x00673000 0x00ae1000 0x00b11800 0x00d40400 0x005a4400 0x00f63000 0x0091c000 0x010c0800 0x012ab800

----------------------------------------------------------------------------------------------------------------

LBA: 0xf240eb | HF: 0x04001275 F1: 0xa1b5ec90 F2: 0x29d3a78e [LOOP: 1 WRITE: 82 SECTOR: 235 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xffff9327d600 | Physical addr: 0x30325e1d600

LBA LIST:
LBA[ 0]: 0x00f24000 0x00dc4c00 0x0020bc00 0x01341400 0x00ae6800 0x009c6400 0x00fe4000 0x00983800 0x004e2800 0x00661800
LBA[10]: 0x010d4000 0x01298800 0x00233000 0x0037f800 0x013ab800 0x005c3800 0x003e7c00 0x013a6800 0x00e3e000 0x006fc000
LBA[20]: 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x00bec800 0x00e69800
LBA[30]: 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400 0x00d69000 0x00fc7800
LBA[40]: 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00b89800 0x0101bc00 0x00207000 0x00430000 0x00c11000 0x00d5d000
LBA[50]: 0x0057a800 0x00bd3400 0x00753c00 0x007c4c00 0x00ed6800 0x00933000 0x00d77400 0x00258400 0x00b7e000 0x00b44000
LBA[60]: 0x008e2800 0x00958800 0x004f1800 0x00128000 0x01310000 0x0042c000 0x000aac00 0x00539800 0x006c4000 0x00375000
LBA[70]: 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000 0x0085c000 0x00e5bc00
LBA[80]: 0x0025f400 0x00af5800 0x00ca3800 0x00f1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800 0x01083800 0x00ec2800
LBA[90]: 0x009c5800 0x000ba000 0x007aac00 0x005b7c00 0x00478000 0x00476000 0x011f6000 0x0138c000 0x00451000 0x00f91c00

PRE LBA LIST:
LBA[ 0]: 0x00a05800 0x00673000 0x00ae1000 0x00b11800 0x00d40400 0x005a4400 0x00f63000 0x0091c000 0x010c0800 0x012ab800

================================================================================================================

Current Time: 2023-06-24 13:06:17
```

LBA工具基本功能演示：hd_write_verify_dump工具数据校验功能

- 簇内数据校验：(-L和-S参数)

```
[root@localhost lba]# hd_write_verify_dump -c -D -L 0xa89ab0 -S 2 lba_test.raw
File Information:
        File:      lba_test.raw
        Size:      10737418240
        Blocks:    965680
        IO Block:  65536
        Device:    2051
        Inode:     1678094440
        Links:     1
        File Size: 10737418240 / 10240M / 10.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-07-12 15:13:18

================================================================================================================

LBA: 0xa89ab0 | HF: 0x04001270 F1: 0xa1b5ec90 F2: 0x29701e70 [LOOP: 1 WRITE: 103 SECTOR: 688 / NUM: 1024 / ZERO: 282]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xffffa1926000 | Physical addr: 0x682a12b6000

LBA LIST:
LBA[ 0]: 0x00a89800 0x0133b000 0x013c0400 0x00bae800 0x00928000 0x007b1000 0x011b7000 0x0068a000 0x00b03000 0x01131000
LBA[10]: 0x00a94800 0x0074b000 0x00b89000 0x01250c00 0x012a3800 0x00fa7400 0x00e86400 0x005d0400 0x00de5800 0x00046400
LBA[20]: 0x00e1ec00 0x001d2000 0x00702400 0x00ce6000 0x006f3000 0x001c5000 0x010ea800 0x00005000 0x009d2000 0x00256800
LBA[30]: 0x00ae5000 0x01078c00 0x001b4800 0x00660c00 0x00d8f000 0x00788000 0x0012f800 0x004e1800 0x0136dc00 0x008f0800
LBA[40]: 0x005d2000 0x00326000 0x01111800 0x00223800 0x013cc800 0x01375800 0x0096b400 0x00735000 0x011cb000 0x00cec000
LBA[50]: 0x00be0800 0x01118c00 0x00f31400 0x002ea000 0x00876000 0x000aa000 0x0071d000 0x00583c00 0x00853000 0x010dbc00
LBA[60]: 0x00cc1800 0x0010c400 0x002b3800 0x00ce8800 0x00e3b000 0x00c50000 0x00444400 0x00456000 0x01211400 0x00ca4800
LBA[70]: 0x00a56400 0x00879400 0x0051ec00 0x00abb000 0x011f0400 0x00a90000 0x002c6400 0x00c8a000 0x00ed3800 0x00164800
LBA[80]: 0x0036e400 0x00902000 0x00234800 0x00196400 0x0133e800 0x0128ac00 0x00550000 0x00cca800 0x00924000 0x00cc1000
LBA[90]: 0x0073e000 0x007efc00 0x013f0800 0x00ab7000 0x00c3f000 0x01303800 0x00fcc000 0x00eba800 0x001ee400 0x00f12000

PRE LBA LIST:
LBA[ 0]: 0x01172000 0x01206000 0x002e8000 0x00810000 0x00a86400 0x0043e000 0x00493800 0x011e4000 0x010e4000 0x00f08c00

----------------------------------------------------------------------------------------------------------------

LBA: 0xa89ab1 | HF: 0x04001270 F1: 0xa1b5ec90 F2: 0x29701e70 [LOOP: 1 WRITE: 103 SECTOR: 689 / NUM: 1024 / ZERO: 282]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xffffa1926200 | Physical addr: 0x682a12b6200

LBA LIST:
LBA[ 0]: 0x00a89800 0x0133b000 0x013c0400 0x00bae800 0x00928000 0x007b1000 0x011b7000 0x0068a000 0x00b03000 0x01131000
LBA[10]: 0x00a94800 0x0074b000 0x00b89000 0x01250c00 0x012a3800 0x00fa7400 0x00e86400 0x005d0400 0x00de5800 0x00046400
LBA[20]: 0x00e1ec00 0x001d2000 0x00702400 0x00ce6000 0x006f3000 0x001c5000 0x010ea800 0x00005000 0x009d2000 0x00256800
LBA[30]: 0x00ae5000 0x01078c00 0x001b4800 0x00660c00 0x00d8f000 0x00788000 0x0012f800 0x004e1800 0x0136dc00 0x008f0800
LBA[40]: 0x005d2000 0x00326000 0x01111800 0x00223800 0x013cc800 0x01375800 0x0096b400 0x00735000 0x011cb000 0x00cec000
LBA[50]: 0x00be0800 0x01118c00 0x00f31400 0x002ea000 0x00876000 0x000aa000 0x0071d000 0x00583c00 0x00853000 0x010dbc00
LBA[60]: 0x00cc1800 0x0010c400 0x002b3800 0x00ce8800 0x00e3b000 0x00c50000 0x00444400 0x00456000 0x01211400 0x00ca4800
LBA[70]: 0x00a56400 0x00879400 0x0051ec00 0x00abb000 0x011f0400 0x00a90000 0x002c6400 0x00c8a000 0x00ed3800 0x00164800
LBA[80]: 0x0036e400 0x00902000 0x00234800 0x00196400 0x0133e800 0x0128ac00 0x00550000 0x00cca800 0x00924000 0x00cc1000
LBA[90]: 0x0073e000 0x007efc00 0x013f0800 0x00ab7000 0x00c3f000 0x01303800 0xcf00eb7c 0x00eba800 0x001ee400 0x00f12000

PRE LBA LIST:
LBA[ 0]: 0x01172000 0x01206000 0x002e8000 0x00810000 0x00a86400 0x0043e000 0x00493800 0x011e4000 0x010e4000 0x00f08c00

================================================================================================================

Current Time: 2023-07-12 15:13:18

[root@localhost lba]#
```

```
[root@localhost lba]# hd_write_verify_dump -c -D -L 0x649418 -S 2 lba.raw
File Information:
        File:      lba.raw
        Size:      4294967296
        Blocks:    7375232
        IO Block:  65536
        Device:    2051
        Inode:     1678094460
        Links:     1
        File Size: 4294967296 / 4096M / 4.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-07-12 15:38:30

================================================================================================================

LBA: 0x649418 | HF: 0x04001277 F1: 0x0892947a F2: 0x9b232b74 [LOOP: 1 WRITE: 629 SECTOR: 24 / NUM: 1024 / ZERO: 4]
IO Time: 2023-07-12 15:37:04

Buffer addr: 0xffff9d133000 | Physical addr: 0x182b4613000

LBA LIST:
LBA[ 0]: 0x00649400 0x000ec800 0x00557c00 0x005cfc00 0x003c6800 0x0017e400 0x00285400 0x00158c00 0x0058f000 0x0018b400
LBA[10]: 0x007bb000 0x001bcc00 0x0075ec00 0x00397c00 0x0064c400 0x007e0c00 0x00259800 0x0015a000 0x005fb800 0x004f5c00
LBA[20]: 0x0057d800 0x00674400 0x00399800 0x0002b400 0x00308800 0x004f6000 0x001f7c00 0x0016a800 0x00638800 0x00409c00
LBA[30]: 0x0040b400 0x005e5000 0x00699000 0x0020c400 0x000fa800 0x007bb400 0x0016ac00 0x0077e400 0x002a8c00 0x00274400
LBA[40]: 0x000d4000 0x0002d000 0x00249400 0x001bf800 0x0040b800 0x002lec00 0x0033e000 0x005d0400 0x003b8800 0x0072f800
LBA[50]: 0x00389800 0x00202400 0x0017ec00 0x007f6800 0x0021f400 0x002f9400 0x00795c00 0x003bf000 0x005c4c00 0x002d4400
LBA[60]: 0x00191c00 0x0049d000 0x001d6400 0x005ca400 0x00333c00 0x004df400 0x0064bc00 0x005eb800 0x0076b800 0x002da400
LBA[70]: 0x000db800 0x0030a000 0x007db800 0x00212800 0x002ec400 0x000b5800 0x003f1400 0x004fd400 0x0066a000 0x007dd400
LBA[80]: 0x0064cc00 0x003d9c00 0x007de800 0x0038cc00 0x0034e000 0x001dfc00 0x001f0800 0x0064d800 0x001f8400 0x001af400
LBA[90]: 0x005f3400 0x004c5400 0x003c4800 0x0059b000 0x0057a800 0x0019ec00 0x006cac00 0x00789800 0x001cf800 0x00263400

PRE LBA LIST:
LBA[ 0]: 0x0002a000 0x00740c00 0x0055ec00 0x00039400 0x007d4c00 0x0071b000 0x007adc00 0x005e3400 0x0013d000 0x0034fc00

----------------------------------------------------------------------------------------------------------------

LBA: 0x649419 | HF: 0x3a3a3a3a F1: 0x3a3a3a3a F2: 0x3a3a3a3a [LOOP: 976894522 WRITE: 976894522 SECTOR: 14906 / NUM: 149
IO Time: 976896422-976894523-976894522 976894522:976894522:976894522

Buffer addr: 0xffff9d133200 | Physical addr: 0x182b4613200

LBA LIST:
LBA[ 0]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[10]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[20]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[30]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[40]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[50]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[60]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[70]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[80]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[90]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a

PRE LBA LIST:
LBA[ 0]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a

================================================================================================================

Current Time: 2023-07-12 15:38:30

[root@localhost lba]#
```

LBA工具基本功能演示：hd_write_verify_dump工具数据校验功能

- dump指定LBA的数据：(-L和-S参数)

```
[root@localhost lba]# hd_write_verify_dump -c -D -L 0xc5800 -S 2 lba.raw
File Information:
        File:       lba.raw
        Size:       4294967296
        Blocks:     7375232
        IO Block:   65536
        Device:     2051
        Inode:      1678094460
        Links:      1
        File Size: 4294967296 / 4096M / 4.00G

        Sector_Per_Cluster: 1024

Current Time: 2023-07-12 16:58:29

================================================================================================================

LBA: 0xc5800 | HF: 0x04001275 F1: 0x0892947a F2: 0x0e0fab86 [LOOP: 1 WRITE: 629 SECTOR: 0 / NUM: 1024 / ZERO: 262]
IO Time: 2023-07-12 15:37:03

Buffer addr: 0xffff92ad0000 | Physical addr: 0x50181300000

LBA LIST:
LBA[ 0]: 0x000c5800 0x001acc00 0x00246800 0x00739800 0x00202000 0x0037c400 0x006d7000 0x00175c00 0x0017e800 0x00688400
LBA[10]: 0x00588800 0x0045c000 0x007f2400 0x00091c00 0x002b8800 0x00088800 0x003b1800 0x00273000 0x0015a400 0x001b7c00
LBA[20]: 0x004c9c00 0x00211c00 0x0023f400 0x000a3800 0x006ee800 0x00409800 0x006c8c00 0x0069d800 0x0035e000 0x0070a000
LBA[30]: 0x00092800 0x00370000 0x00344c00 0x005ec800 0x001c5400 0x004ca000 0x001c3400 0x001c3c00 0x000a7000 0x00486c00
LBA[40]: 0x0008fc00 0x003ab000 0x004fd800 0x006c9400 0x004e0800 0x0058dc00 0x00059400 0x0053ac00 0x0037cc00 0x00683800
LBA[50]: 0x00369800 0x004b3c00 0x00454400 0x003ab800 0x003c3400 0x0036a000 0x00194800 0x00076c00 0x003c3c00 0x00455400
LBA[60]: 0x003c7400 0x004e6c00 0x003f0400 0x00766400 0x00730c00 0x0043fc00 0x0063ac00 0x00687400 0x00261800 0x0068f800
LBA[70]: 0x00732000 0x007dcc00 0x005eec00 0x00753400 0x00207800 0x005ef800 0x006cb800 0x00353c00 0x00313800 0x00233800
LBA[80]: 0x001ce000 0x002e7400 0x0003dc00 0x00578400 0x0073a800 0x00797800 0x0038f000 0x006da800 0x003e1800 0x00797c00
LBA[90]: 0x0009d000 0x00776000 0x00249c00 0x00733c00 0x0062c000 0x00042400 0x001e9c00 0x005b4800 0x005f3c00 0x00655c00

PRE LBA LIST:
LBA[ 0]: 0x00307000 0x001b0c00 0x001d5400 0x0056f000 0x00224c00 0x0004fc00 0x0068ec00 0x0049ac00 0x00727c00 0x003a9c00

----------------------------------------------------------------------------------------------------------------

LBA: 0xc5801 | HF: 0x04001275 F1: 0x0892947a F2: 0x0e0fab86 [LOOP: 1 WRITE: 629 SECTOR: 1 / NUM: 1024 / ZERO: 262]
IO Time: 2023-07-12 15:37:03

Buffer addr: 0xffff92ad0200 | Physical addr: 0x50181300200

LBA LIST:
LBA[ 0]: 0x000c5800 0x001acc00 0x00246800 0x00739800 0x00202000 0x0037c400 0x006d7000 0x00175c00 0x0017e800 0x00688400
LBA[10]: 0x00588800 0x0045c000 0x007f2400 0x00091c00 0x002b8800 0x00088800 0x003b1800 0x00273000 0x0015a400 0x001b7c00
LBA[20]: 0x004c9c00 0x00211c00 0x0023f400 0x000a3800 0x006ee800 0x00409800 0x006c8c00 0x0069d800 0x0035e000 0x0070a000
LBA[30]: 0x00092800 0x00370000 0x00344c00 0x005ec800 0x001c5400 0x004ca000 0x001c3400 0x001c3c00 0x000a7000 0x00486c00
LBA[40]: 0x0008fc00 0x003ab000 0x004fd800 0x006c9400 0x004e0800 0x0058dc00 0x00059400 0x0053ac00 0x0037cc00 0x00683800
LBA[50]: 0x00369800 0x004b3c00 0x00454400 0x003ab800 0x003c3400 0x0036a000 0x00194800 0x00076c00 0x003c3c00 0x00455400
LBA[60]: 0x003c7400 0x004e6c00 0x003f0400 0x00766400 0x00730c00 0x0043fc00 0x0063ac00 0x00687400 0x00261800 0x0068f800
LBA[70]: 0x00732000 0x007dcc00 0x005eec00 0x00753400 0x00207800 0x005ef800 0x006cb800 0x00353c00 0x00313800 0x00233800
LBA[80]: 0x001ce000 0x002e7400 0x0003dc00 0x00578400 0x0073a800 0x00797800 0x0038f000 0x006da800 0x003e1800 0x00797c00
LBA[90]: 0x0009d000 0x00776000 0x00249c00 0x00733c00 0x0062c000 0x00042400 0x001e9c00 0x005b4800 0x005f3c00 0x00655c00

PRE LBA LIST:
LBA[ 0]: 0x00307000 0x001b0c00 0x001d5400 0x0056f000 0x00224c00 0x0004fc00 0x0068ec00 0x0049ac00 0x00727c00 0x003a9c00

================================================================================================================

Current Time: 2023-07-12 16:58:29

[root@localhost lba]# hd_write_verify_dump -c -D -L 0xc5800 -S 1024 lba.raw > dump.log
[root@localhost lba]#
```

dump.log

目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

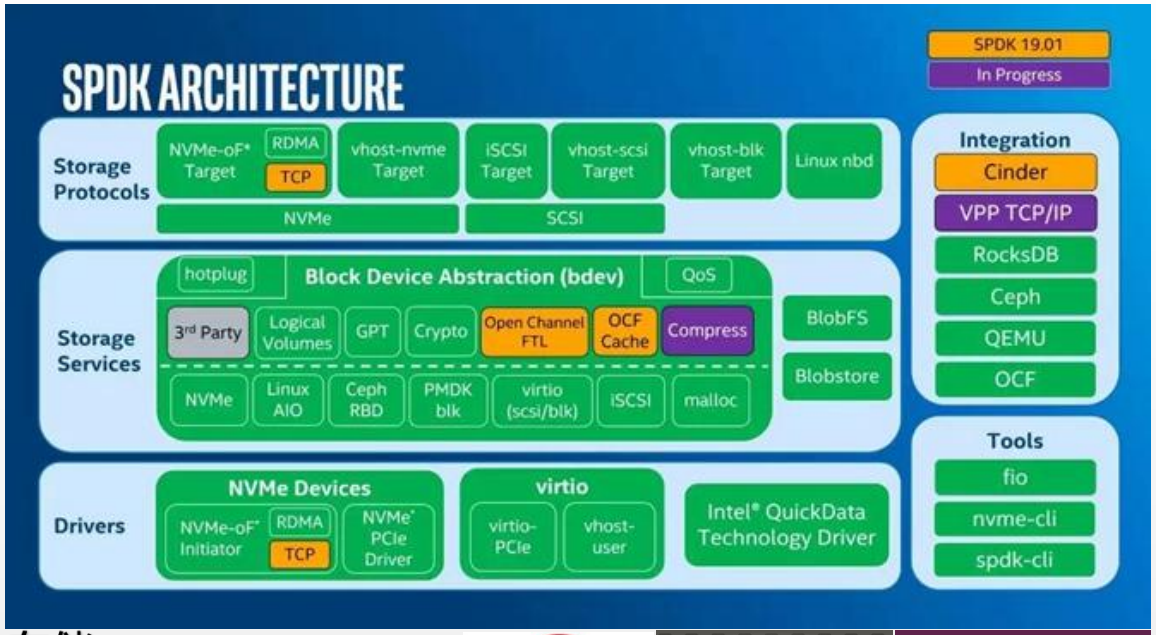
04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

LBA工具典型应用场景：存储、OS

- 各种类型存储及文件系统的稳定性、可靠性测试

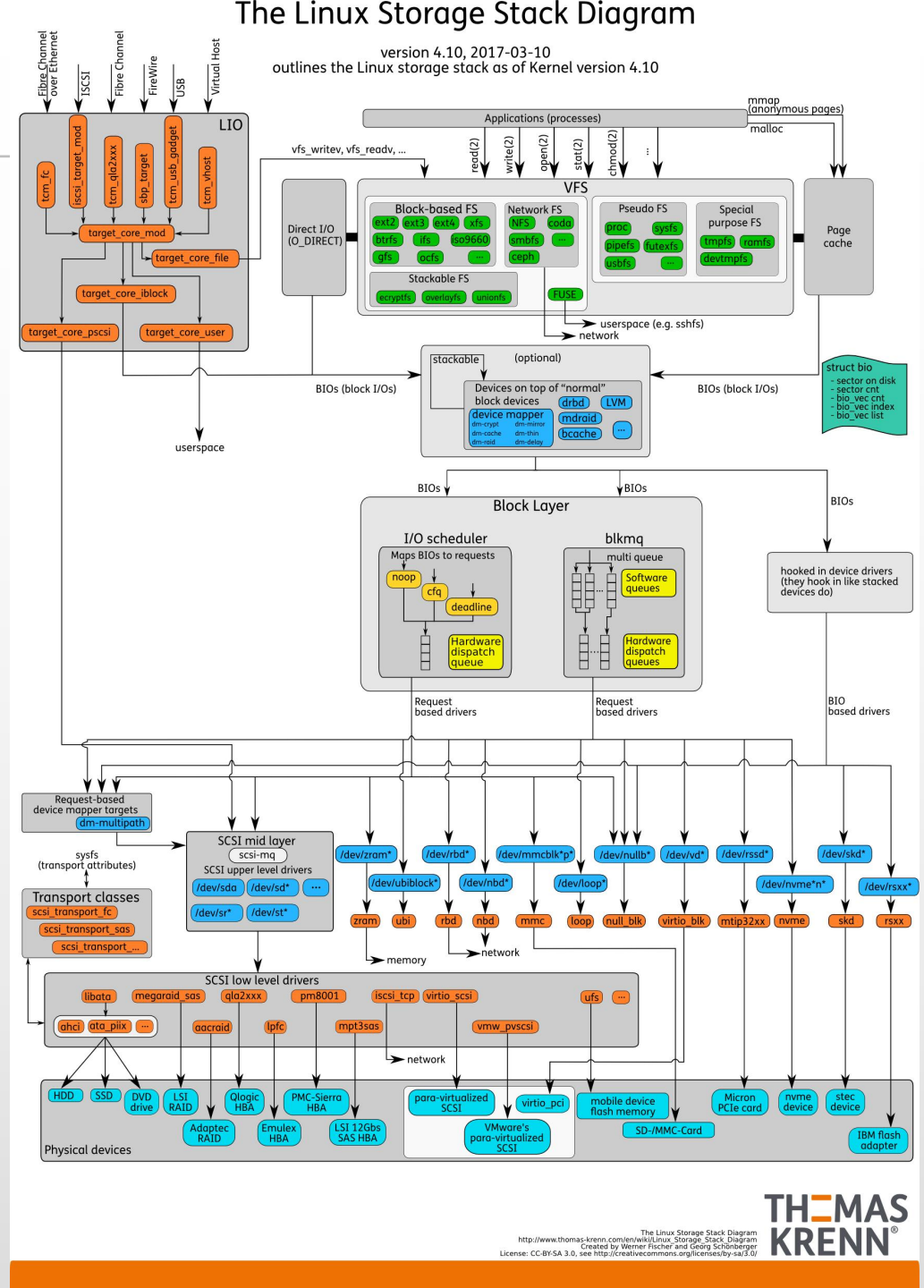


存储：

1. 内核存储协议栈
2. spdk
3. ceph
4. glusterfs
5. linstor
6. longhorn
7. ipsan/fcsan/multipath/nas/.....

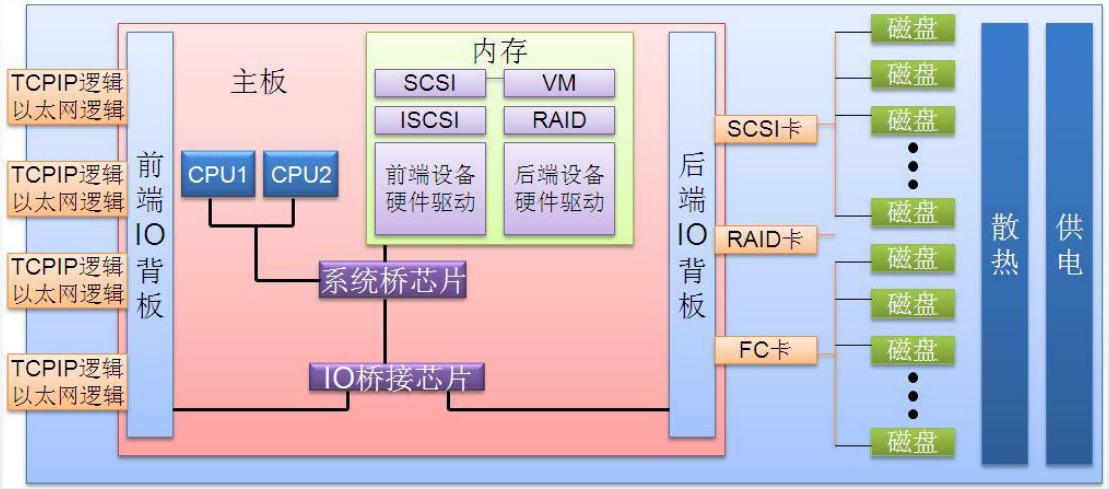
文件系统：

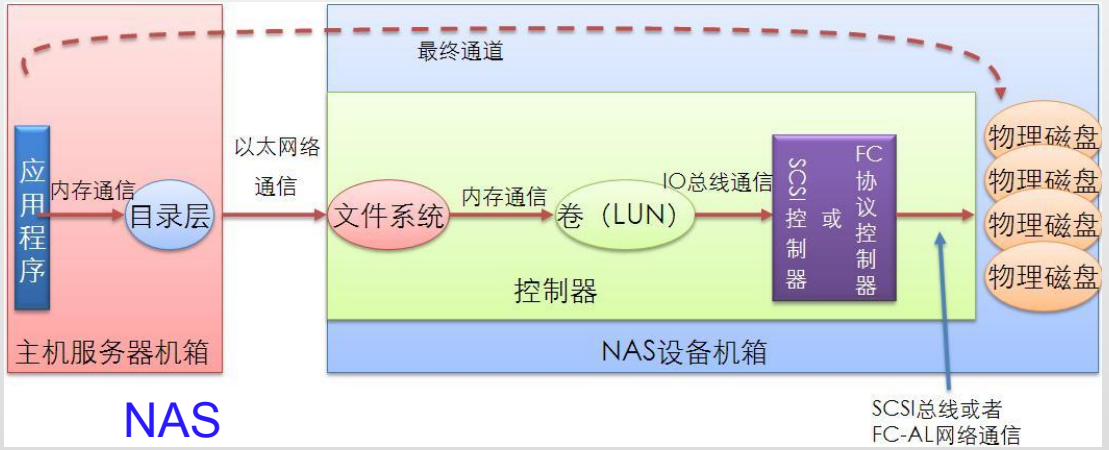
1. 本地文件系统：ext2/ext3/ext4/xfs/.....
2. 分布式文件系统：ocfs2/gfs2/.....
3. 内存文件系统: tmpfs/.....

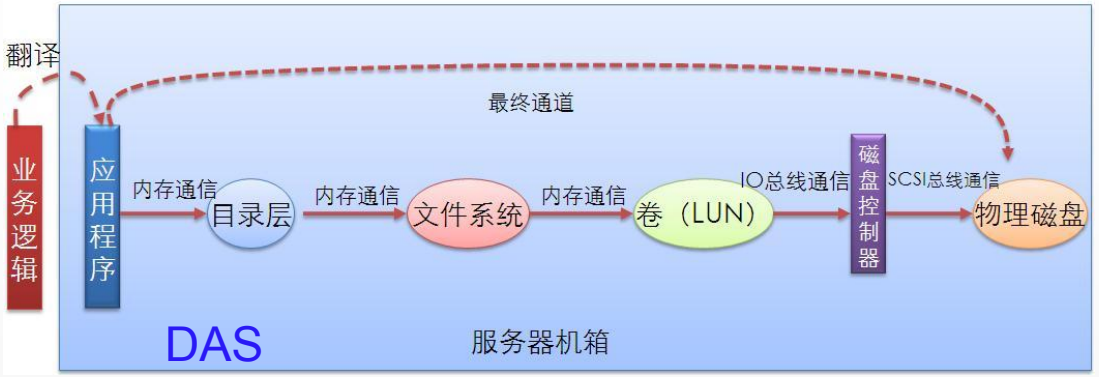


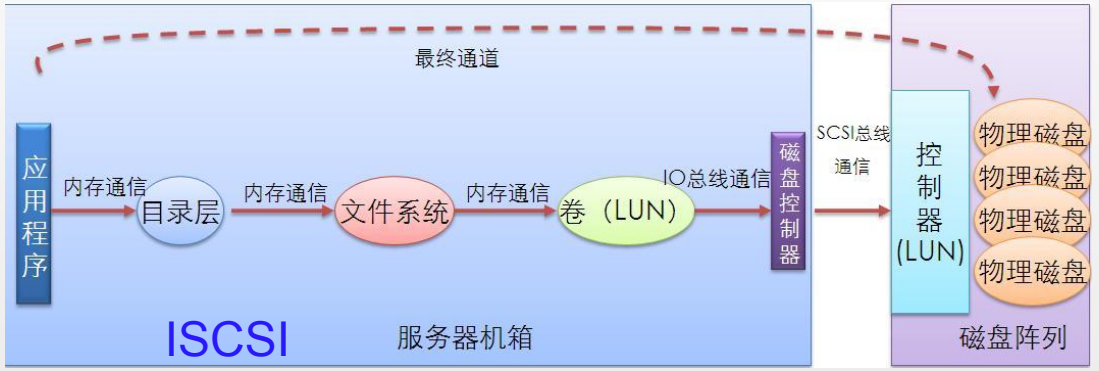
LBA工具典型应用场景：存储、OS

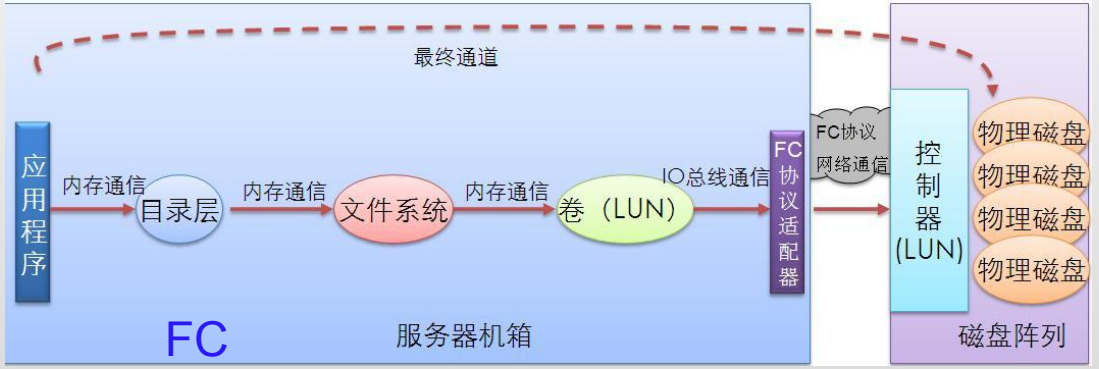
1. 磁盘硬件(磁盘坏道等--badblocks)
2. 间接测试校验存储压缩功能(例如：qcow2压缩等)
3. 间接测试校验存储加密功能(例如：qcow2加密、磁盘luks加密等)





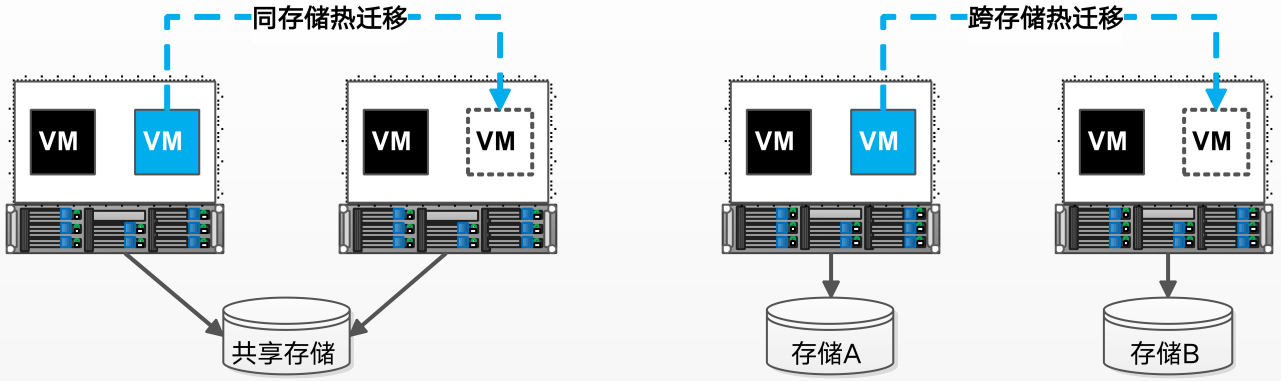






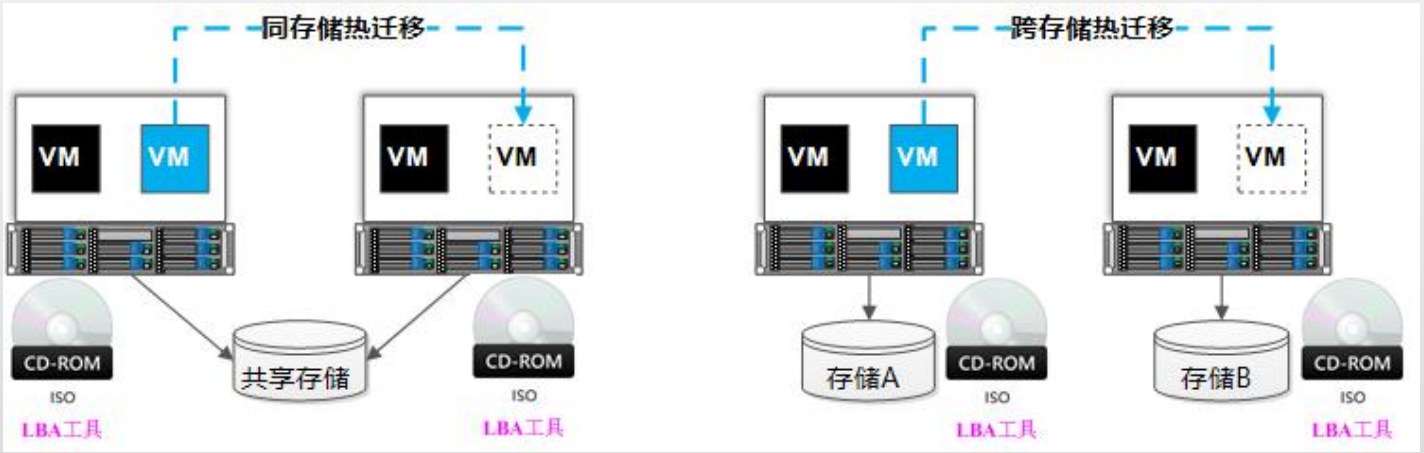
LBA工具典型应用场景：云计算

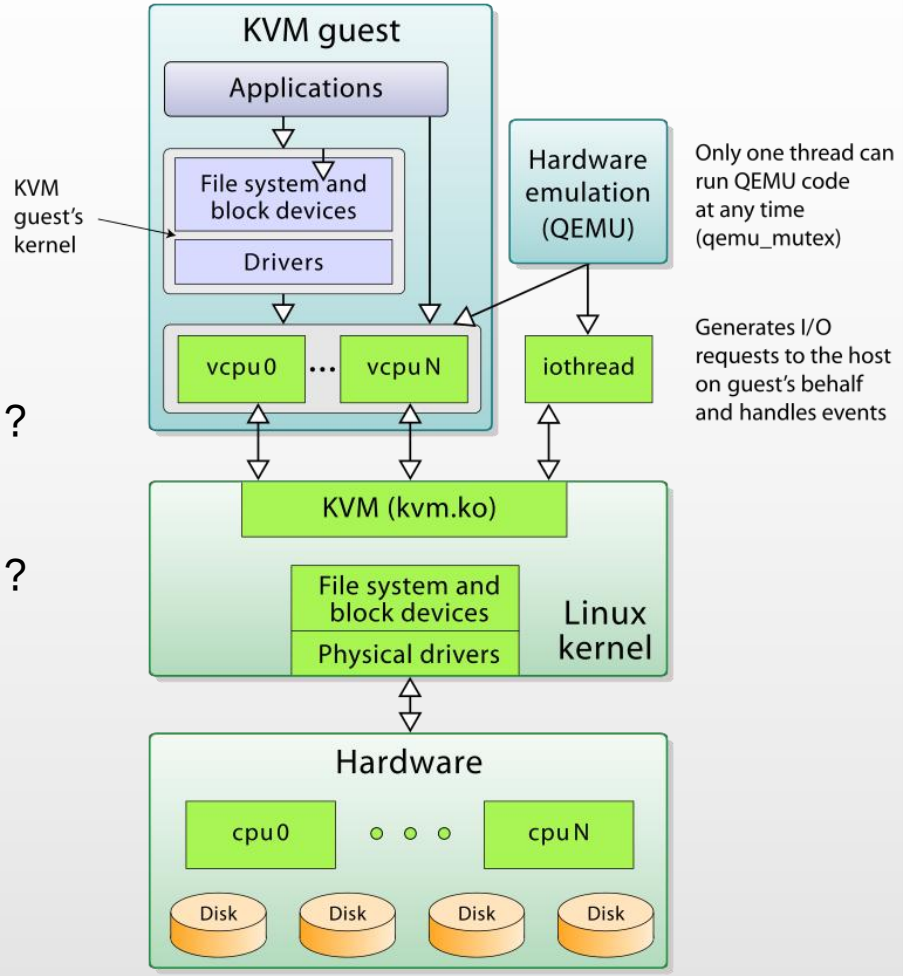
- 虚拟机热迁移存储/内存数据一致性校验



(1) 虚拟机热迁移，怎么验证内存数据迁移到目标端虚拟机后，数据没有出错？
(2) 虚拟机热迁移，怎么验证内存数据迁移，源端虚拟机没有漏迁数据？

(3) 虚拟机热迁移，怎么验证磁盘数据迁移到目标端虚拟机后，数据没有出错？
(4) 虚拟机热迁移，怎么验证磁盘数据迁移，源端虚拟机没有漏迁数据？





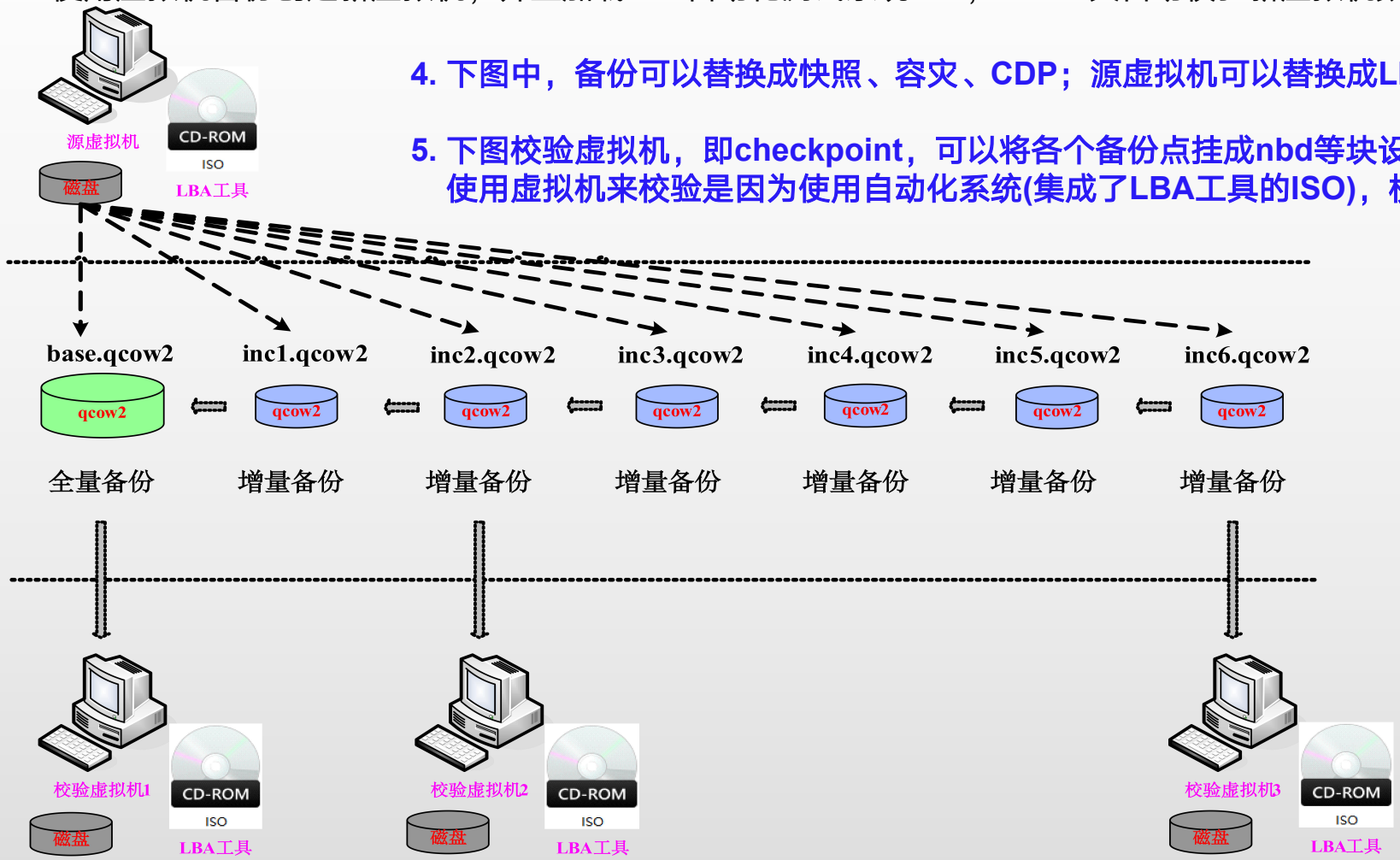
LBA工具典型应用场景：快照、备份、容灾、CDP

- 校验快照、备份、容灾、CDP等数据一致性校验

1. 源虚拟机配置LBA自动化测试系统ISO和数据盘，LBA工具写磁盘模拟业务系统的IO数据；
2. 定期对虚拟机进行全量备份和增量备份；
3. 使用虚拟机备份创建新虚拟机，并且加载LBA自动化测试系统ISO，LBA工具自动校验新虚拟机数据盘中的数据一致性；

4. 下图中，备份可以替换成快照、容灾、CDP；源虚拟机可以替换成LBA工具测试存储(比如：存储支持快照)；

5. 下图校验虚拟机，即checkpoint，可以将各个备份点挂成nbd等块设备，直接使用LBA工具校验数据一致性；使用虚拟机来校验是因为使用自动化系统(集成了LBA工具的ISO)，校验checkpoint的数据一致性更方便。



目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

存储稳定性测试与数据一致性校验自动化测试系统演示

- 自动化测试系统ISO，包括：linux_x86 / linux_arm / winPE 三种版本

名称	修改日期
hd_write_verify.with.stripe.split.64K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:41
hd_write_verify.with.stripe.split.128K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:41
hd_write_verify.with.stripe.split.256K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:40
hd_write_verify.with.stripe.split.512K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:40
hd_write_verify.with.stripe.split.1M.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:40
hd_write_verify.with.stripe.robin.1K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:39
hd_write_verify.with.stripe.robin.4K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:39
hd_write_verify.with.stripe.robin.8K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:38
hd_write_verify.with.stripe.robin.16K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:38
hd_write_verify.with.stripe.robin.64K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:38
hd_write_verify.with.stripe.robin.128K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:37
hd_write_verify.with.stripe.robin.256K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:37
hd_write_verify.with.stripe.robin.512K.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:37
hd_write_verify.with.stripe.robin.1M.整体校验+批量校验.x86_64.V10.iso	2024/3/22 0:37
hd_write_verify.disk_lba_check_only.with.stripe.split.x86_64.V10.iso	2024/3/22 0:36
hd_write_verify.disk_lba_check_only.with.stripe.robin.x86_64.V10.iso	2024/3/22 0:36
hd_write_verify.mem_lba.with.stripe.robin.1M.x86_64.V10.iso	2024/3/22 0:36
hd_write_verify.mem_disk_lba.with.stripe.robin.1M.x86_64.V10.iso	2024/3/22 0:36

名称	修改日期	大小
hd_write_verify.with.stripe.split.64K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:56	64,486 KB
hd_write_verify.with.stripe.split.128K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:55	64,486 KB
hd_write_verify.with.stripe.split.256K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:54	64,486 KB
hd_write_verify.with.stripe.split.512K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:53	64,486 KB
hd_write_verify.with.stripe.split.1M.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:51	64,486 KB
hd_write_verify.with.stripe.robin.1K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:51	64,486 KB
hd_write_verify.with.stripe.robin.4K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:49	64,486 KB
hd_write_verify.with.stripe.robin.8K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:48	64,486 KB
hd_write_verify.with.stripe.robin.16K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:47	64,486 KB
hd_write_verify.with.stripe.robin.64K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:46	64,486 KB
hd_write_verify.with.stripe.robin.128K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:45	64,486 KB
hd_write_verify.with.stripe.robin.256K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:44	64,486 KB
hd_write_verify.with.stripe.robin.512K.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:43	64,486 KB
hd_write_verify.with.stripe.robin.1M.整体校验+批量校验.aarch64.V10.iso	2024/3/25 0:41	64,486 KB
hd_write_verify.disk_lba_check_only.with.stripe.split.aarch64.V10.iso	2024/3/25 0:40	64,486 KB
hd_write_verify.disk_lba_check_only.with.stripe.robin.aarch64.V10.iso	2024/3/25 0:39	64,488 KB
hd_write_verify.mem_lba.with.stripe.robin.1M.aarch64.V10.iso	2024/3/25 0:38	64,486 KB
hd_write_verify.mem_disk_lba.with.stripe.robin.1M.aarch64.V10.iso	2024/3/25 0:35	64,490 KB

存储稳定性测试与数据一致性校验自动化测试系统演示

1.虚拟机添加hd_write_verify.mem_disk_lba.with.stripe.robin.1M.x86_64.V2510.iso，作为启动盘；
2.添加数据盘，最多可以添加64个数据盘（一般只需要添加一个数据盘进行测试即可）；
3.运行虚拟机，稳定性自动化测试就跑起来了。（可以对虚拟机注入各种故障：断电，断网，存储IO错误等）；
4.如果虚拟机内部运行的LBA工具，报出了“BUG”关键字输出（鲜红色），即测试出了LBA问题（出现数据一致性错误）；

BUG 001[1]: [Thread: 1] RANDOM VERIFY DIFFER FLAGS | ERROR LBA: 0x794400

BUG 001[2]: [Thread: 3] RANDOM VERIFY DIFFER LBA_LIST | ERROR LBA: 0x28c00

BUG 001[3]: [Thread: 1] BATCH VERIFY DIFFER FLAGS | ERROR LBA: 0x4f5000

BUG 001[4]: [Thread: 9] BATCH VERIFY DIFFER LBA_LIST | ERROR LBA: 0x404800

BUG 002[1]: [Thread: 5] RANDOM VERIFY SECTOR[647] DIFFER: 1
| CORRECT LBA[0]: 0x63f400 | ERROR LBA[647]: 0x63f687 | filename: lba.raw

BUG 002[2]: [Thread: 2] RANDOM VERIFY ZERO SECTOR[63] DIFFER: 1
| CORRECT LBA[0]: 0x57ef00 | ERROR LBA[63]: 0x57ef3f | filename: lba.raw

BUG 002[3]: [Thread: 3] VERIFY SECTOR[306] DIFFER: 1
| CORRECT LBA[0]: 0xb2a000 | ERROR LBA[306]: 0xb2a132 | filename: lba.raw

BUG 002[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54
| CORRECT LBA[0]: 0x7e9800 | ERROR LBA[970]: 0x7e9bca

BUG 003: [Thread: 3] WRITE PART SECTOR | read[83]

BUG 004 [Thread: 5] DATA LOST
PREV LBA: 0x4a5100 | LOST LBA: 0x35480 | NEXT LBA: 0x1c93780

BUG 005: [Thread: 0] VERIFY HD_FLAG ERROR | HF: 0x0400ffff[0x04001270]

BUG 006: [Thread: 5] VERIFY DIFFER FLAGS | read[248], write[709]

BUG 007[1] : [Thread: 9] VERIFY SECTOR[271] DIFFER: 1 | read[241]

BUG 007[2] : [Thread: 8] VERIFY ZERO SECTOR[853] DIFFER: 1 | read[259]

BUG 007[3]: [Thread: 7] VERIFY SECTOR[310] DIFFER: 11 | read[498]

BUG 007[4]: [Thread: 1] VERIFY ZERO SECTOR[970] DIFFER: 54 | read[534]

存储稳定性测试与数据一致性校验自动化测试系统演示

(1) 磁盘数据一致性测试与校验 (默认: alt + F1)

(2) 内存数据一致性测试与校验 (tty2: alt + F2)

(3) iostat查看IO性能及精准限速(比较稳定的负载) (tty3: alt + F3, iostat -x 3)

(4) 查看内存变脏速度(适用于虚拟机热迁移场景) (tty4: alt + F4, tailf /var/log/mem_dirty_speed.log)

(5) 自动化测试简单演示(**录屏**) (查看控制台信息: shift + pageup向上翻页, shift + pagedown向下翻页)

LBA_iso.gif

存储稳定性测试与数据一致性校验自动化测试系统演示

- 磁盘数据一致性测试与校验

1. 多磁盘条带测试：校验磁盘组快照、备份等数据一致性

2. 虚拟机热迁移：磁盘数据一致性测试

```
-------------------------------------------------

       Welcome to the YOUPLUS's LBA TESTING SYSTEM

       https://github.com/zhangyoujia/

-------------------------------------------------


YOUPLUS login: root (automatic login)
echo never > /sys/kernel/mm/transparent_hugepage/enabled

hd_write_verify -c -D -K -R 33 -w on -S 2048 -V all -T 10 -L 102400 -P robin -I /dev/vda -I /dev/vdb
Device Topology:
        Disk: /dev/vda
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 10737418240 / 10240M / 10.00G

Device Topology:
        Disk: /dev/vdb
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 10737418240 / 10240M / 10.00G

Disk: stripe | Thread: 10 | Total Sectors: 83886080 | Total Clusters: 20480 | Sectors of Cluster: 2048 | DIRECT IO & NO Flush | Verify: 6 | Notify: 0

Current Time: 2023-07-10 13:35:10
Thread 4 [tid: 1052]: Starting check disk ...
Thread 9 [tid: 1057]: Starting check disk ...
Thread 7 [tid: 1055]: Starting check disk ...
Thread 1 [tid: 1049]: Starting check disk ...
Thread 0 [tid: 1047]: Starting check disk ...
Thread 3 [tid: 1051]: Starting check disk ...
Thread 5 [tid: 1053]: Starting check disk ...
Thread 8 [tid: 1056]: Starting check disk ...
Thread 2 [tid: 1050]: Starting check disk ...
Thread 6 [tid: 1054]: Starting check disk ...

-------------------------------------------------------------------------------------------------------------------

Starting write disk:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-10 13:35:10
2283, 1629 | 2288, 1635 | 2280, 1633 | 2326, 1656 | 2303, 1643 | 2296, 1637 | 2267, 1645 | 2281, 1631 | 2267, 1645 | 2273, 1640 |
```

存储稳定性测试与数据一致性校验自动化测试系统演示

- iostat查看IO性能及精准限速(比较稳定的负载)

```
-----------------------------------------------------

       Welcome to the YOUPLUS's LBA TESTING SYSTEM

       https://github.com/zhangyoujia/

-----------------------------------------------------



YOUPLUS login: root (automatic login)
Last login: Mon Jul 10 13:35:10 GMT-8 2023 on tty2
root/YOUPLUS /var/iso/tools # lsblk
NAME  MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop1   7:1    0   3.2G  0 loop
vdb   253:16   0    10G  0 disk
sr0    11:0    1  64.6M  0 rom  /var/iso
vda   253:0    0    10G  0 disk
root/YOUPLUS /var/iso/tools # iostat -x 3
Linux 4.18.0 (YOUPLUS)  07/10/23        _x86_64_        (2 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           4.20    0.00   17.05   36.29    0.36   42.11

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00   38.61   41.95 39073.19 26445.41  1626.40     0.79   11.82   14.84    9.03   7.49  60.38
vdb               0.00     0.00   20.14   23.18 20153.62 14441.90  1597.42     0.37   10.51   14.65    6.91   9.37  40.58
sr0               0.00     0.00    0.36    0.00     1.34     0.00     7.40     0.00    0.40    0.40    0.00   1.01   0.04
loop1             0.00     0.00  217.16  264.27 110050.83 83074.29   802.29     0.25    1.50    1.46    1.52   0.88  42.24

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5.30    0.00   12.58   41.72    0.17   40.23

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00   34.00   33.67 34816.00 22186.67  1684.81     0.61   10.85   17.30    4.34   8.93  60.40
vdb               0.00     0.00   25.33   31.67 25941.33 19235.67  1585.16     0.43    9.31   16.03    3.94   8.09  46.13
sr0               0.00     0.00    0.33    0.00     0.67     0.00     4.00     0.00    1.00    1.00    0.00   8.00   0.27
loop1             0.00     0.00  200.00  332.00 102400.00 102400.00   769.92     0.10    0.96    0.93    0.98   0.68  36.13

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           3.20    0.00   13.13   36.70    0.34   46.63

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00   40.33   20.00 41301.33 11946.67  1765.13     0.53   10.78   14.02    4.23   7.73  46.67
vdb               0.00     0.00   33.33   25.33 34133.33 15018.67  1675.64     0.42    9.19   13.82    3.11   8.45  49.60
sr0               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
loop1             0.00     0.00  200.00  319.33 102400.00 102400.00   788.70     0.12    1.04    1.01    1.05   0.70  36.27
```

存储稳定性测试与数据一致性校验自动化测试系统演示

- 内存数据一致性测试与校验
  1. 虚拟机热迁移：内存数据一致性测试(tmpfs + loop设备)

```
Welcome to the YOUPLUS's LBA TESTING SYSTEM

https://github.com/zhangyoujia/
```

```
YOUPLUS login: root (automatic login)
Last login: Mon Jul 10 13:35:10 GMT-8 2023 on tty2
root/YOUPLUS /var/iso/tools # lsblk
NAME  MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop1   7:1    0   3.2G  0 loop
vdb   253:16   0    10G  0 disk
sr0    11:0    1  64.6M  0 rom  /var/iso
vda   253:0    0    10G  0 disk
root/YOUPLUS /var/iso/tools #
root/YOUPLUS /var/iso/tools # mount
rootfs on / type rootfs (rw,size=1995196k,nr_inodes=498799)
tmpfs on /run type tmpfs (rw,nosuid,noexec,relatime,size=163
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relati
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev type tmpfs (rw,relatime,size=32768k,mode=755)
/dev/sr0 on /var/iso type iso9660 (ro,relatime,norock,check=
tmpfs on /var/mem type tmpfs (rw,relatime,size=3329024k)
root/YOUPLUS /var/iso/tools #
root/YOUPLUS /var/iso/tools # losetup -a
/dev/loop1: [0018]:19412 (/var/mem/mem_test.raw)
root/YOUPLUS /var/iso/tools #
```

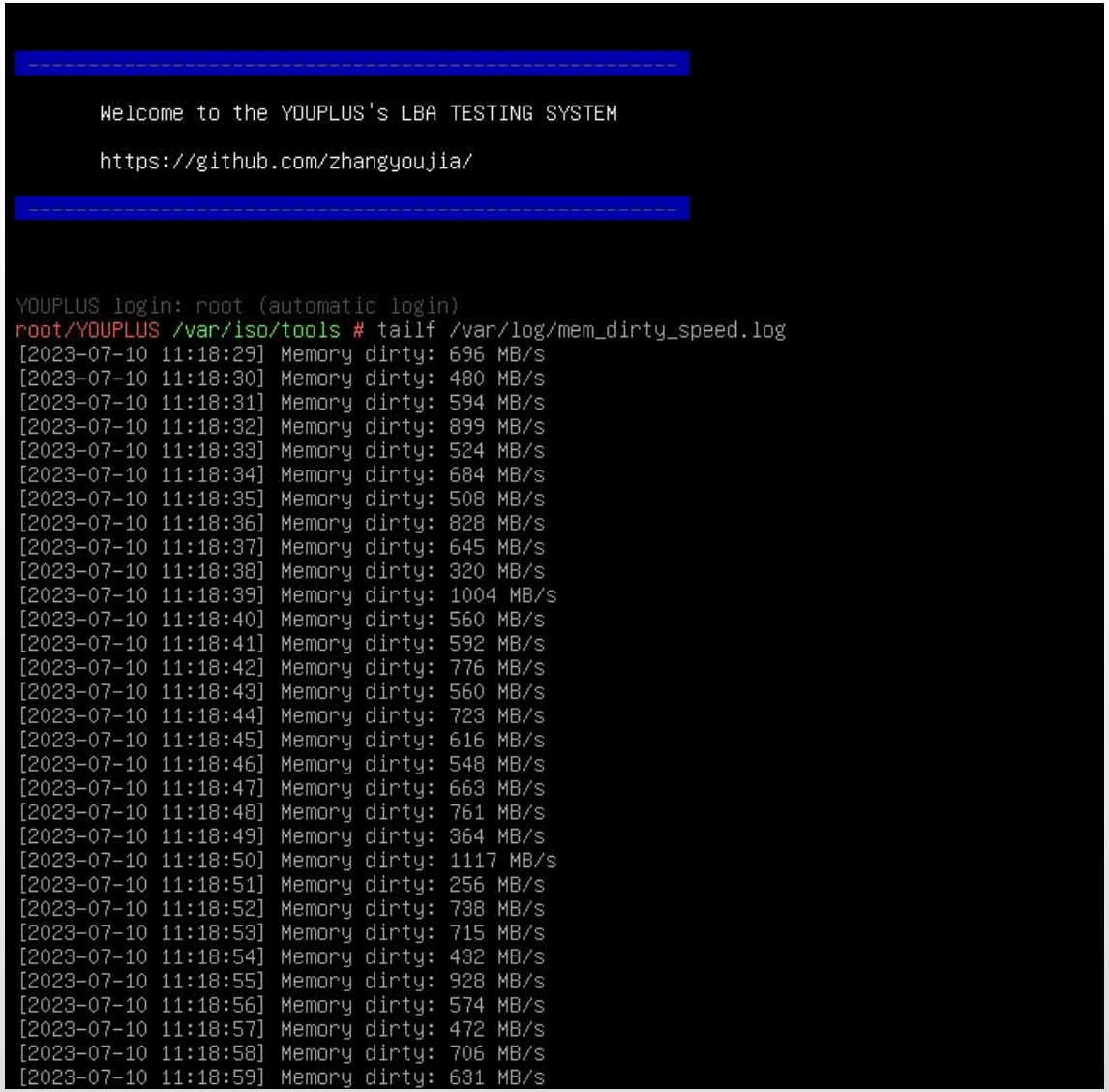
```
Welcome to the YOUPLUS's LBA TESTING SYSTEM

https://github.com/zhangyoujia/
```

```
YOUPLUS login: root (automatic login)
Last login: Mon Jul 10 14:06:03 GMT-8 2023 on tty1
hd_write_verify -c -D -K -R 33 -w on -S 1024 -V once -T 10 -L 204800 /dev/loop1
Device Topology:
        Disk: /dev/loop1
        Logical  block size: 512
        Physical block size: 512
        Minimum I/O size: 512
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 0
        Disk Size: 3387949056 / 3231M / 3.16G

Disk: /dev/loop1 | Thread: 10 | Total Sectors: 6617088 | Total Clusters: 6462 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush | Verify: 2 | Notify: 0

Current Time: 2023-07-10 14:06:05
Thread 5 [tid: 1122]: Starting check disk ...
Thread 6 [tid: 1123]: Starting check disk ...
Thread 1 [tid: 1118]: Starting check disk ...
Thread 9 [tid: 1126]: Starting check disk ...
Thread 4 [tid: 1121]: Starting check disk ...
Thread 2 [tid: 1119]: Starting check disk ...
Thread 3 [tid: 1120]: Starting check disk ...
Thread 8 [tid: 1125]: Starting check disk ...
Thread 0 [tid: 1117]: Starting check disk ...
Thread 7 [tid: 1124]: Starting check disk ...

-------------------------------------------------------------------------------------------------------------------------

Starting write disk:  Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit

Loop 1: .
Current Time: 2023-07-10 14:06:06
371, 298 | 369, 301 | 372, 299 | 375, 302 | 388, 309 | 378, 303 | 376, 301 | 372, 298 | 380, 304 | 362, 308 |

Current Time: 2023-07-10 14:06:39
Thread 8 [tid: 1125]: Starting check disk ...
Thread 4 [tid: 1121]: Starting check disk ...
Thread 9 [tid: 1126]: Starting check disk ...
Thread 2 [tid: 1119]: Starting check disk ...
Thread 5 [tid: 1122]: Starting check disk ...
Thread 6 [tid: 1123]: Starting check disk ...
```

存储稳定性测试与数据一致性校验自动化测试系统演示

- 查看内存变脏速度 (适用于虚拟机热迁移场景)

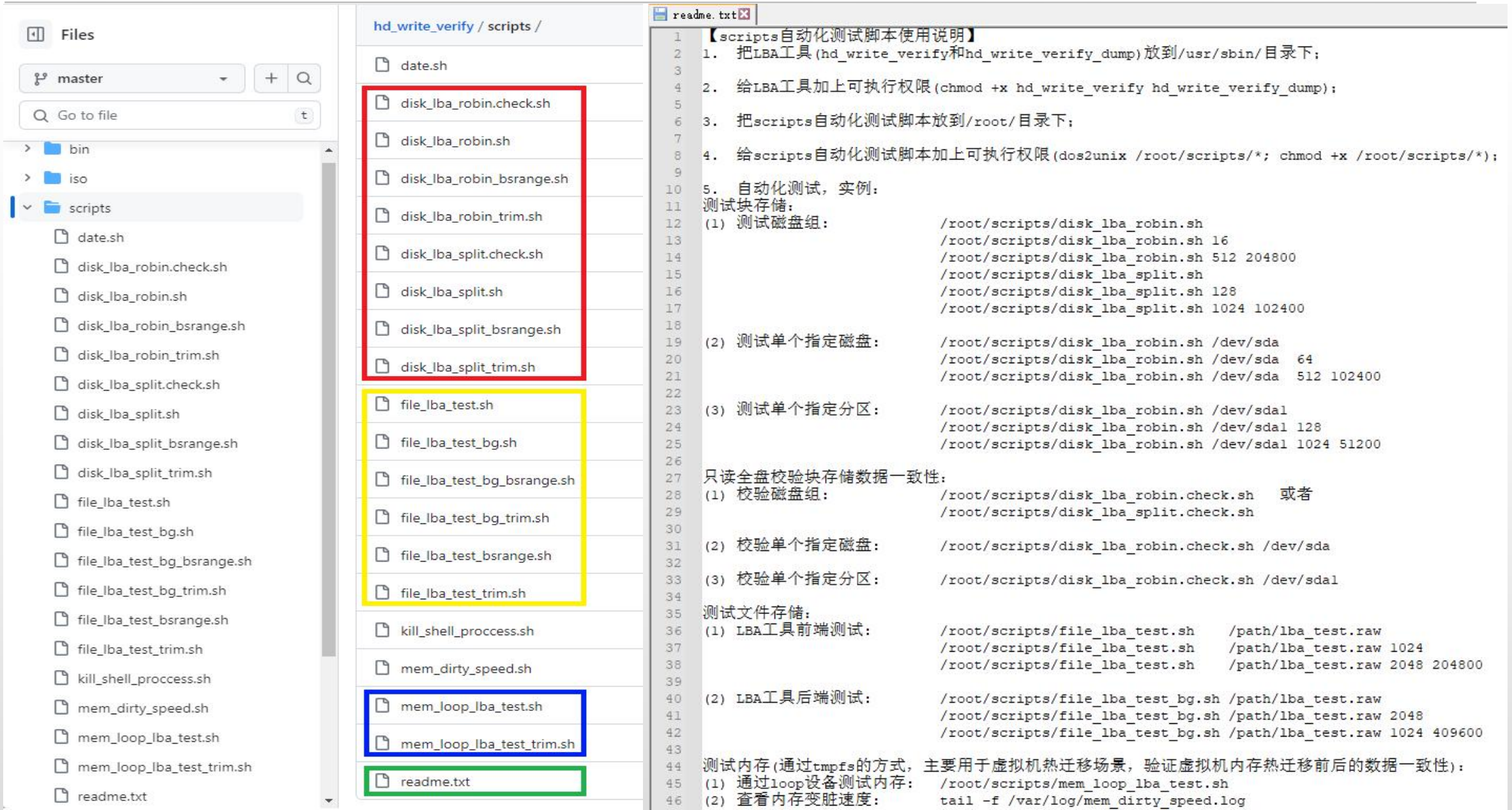
  虚拟机热迁移性能优化：multifd / dirty-ring / 内存压缩 / VCPU节流策略等



存储稳定性测试与数据一致性校验自动化测试系统演示

- 自动化测试简单演示(**录屏**)

存储稳定性测试与数据一致性校验自动化测试脚本



物理存储稳定性测试与数据一致性校验方案

pynvme

2015年Intel开源了SPDK，为数据中心存储提供一系列软件支持，其中包括了NVMe驱动。SPDK的整体架构非常清晰，NVMe驱动模块的接口完整，实现了大部分NVMe设备的功能。SPDK也经过数据中心的实践检验，有稳定性的保障。但另一方面，SPDK毕竟是以数据中心存储系统为设计目标，会缺少一些SSD测试所需要的功能，譬如：

1. SPDK采用轮询，不支持中断；
2. SPDK只提供C语言的接口，直接在SPDK上面开发测试程序的成本高，并且容易引入bug；
3. 没有提供cmdlog等专门用于测试和调试的功能，测试过程中遇到问题不容易分析；
4. NVMe的SQ和CQ被绑定为1:1的Qpair；
5. 只支持NVMe协议。

但是作为一个Intel的开源项目，SPDK的文档和代码非常规范，模块之间高度解耦。我们以SPDK的NVMe驱动为起点，开发了NVMe设备的测试驱动。这就是pynvme的缘起。

在SPDK的基础上，我们在pynvme中实现了这些功能：

1. 支持MSIx中断：通过软件实现的中断控制器，提供中断相关的测试接口；
2. 通过Cython的封装，为SPDK的NVMe驱动提供了Python接口，抽象出controller, namespace, qpair, buffer, ioworker等对象和方法；
3. 实现了cmdlog：将NVMe测试驱动最近发送的若干条NVMe command及其completion结构记录到cmdlog中；
4. 提供通用化的命令接口，可以发送任何NVMe command，包括VS command，甚至是不符合NVMe协议要求的命令；
5. 结合pytest及其fixture，进一步简化测试脚本的开发和执行；
6. 通过vscode及其插件，在IDE中开发和调试测试脚本，并能查看测试过程中寄存器、Qpair以及cmdlog的数据；
7. 通过S3/standby模式实现SSD设备的电源控制，无需任何额外设备就可以在普通PC上实现部分上下电测试；
8. 实现ioworker，以多进程的方式对同一个SSD设备进行并发读写操作，加大测试压力；
9. 自动计算、记录、校验每个LBA的CRC，实现数据完整性的测试；
10. 自动在每个LBA数据中插入LBA地址等信息，以发现其他数据一致性问题；
11. 可以读写PCI配置空间，以及BAR空间。

pynvme实现了众多NVMe SSD测试所需的功能，并通过Python提供一致的脚本接口。就像美味的煎饼果子，用煎饼包裹各种不同的食材，呈现统一而诱人的风味。pynvme就是专为NVMe SSD测试而生的煎饼大侠！:)

https://github-wiki-see.page/m/pynvme/pynvme/wiki/初探pynvme

https://github-wiki-see.page/m/pynvme/pynvme/wiki/实战pynvme

SSD盘可以参考pynvme测试方案：

1.LBA工具侧重于左下图红框中的三项测试功能；

2.LBA工具支持磁盘组(多盘条带)测试；(见前页)

```
disk_lba_robin.sh                （搜索所有空闲盘，以robin条带策略，IO大小：1MB 测试）
disk_lba_robin.sh 16             （同上，IO大小：8KB 测试）
disk_lba_robin.sh 512 204800     （同上，IO大小：256KB 测试，同时限速：200MB/s）

disk_lba_split.sh                （搜索所有空闲盘，以split条带策略，IO大小：1MB 测试）
disk_lba_split.sh 128            （同上，IO大小：64KB 测试）
disk_lba_split.sh 1024 102400    （同上，IO大小：512KB 测试，同时限速：100MB/s）
```

3.LBA工具支持全盘数据的trim-all测试和每个IO的discard测试等高级功能；

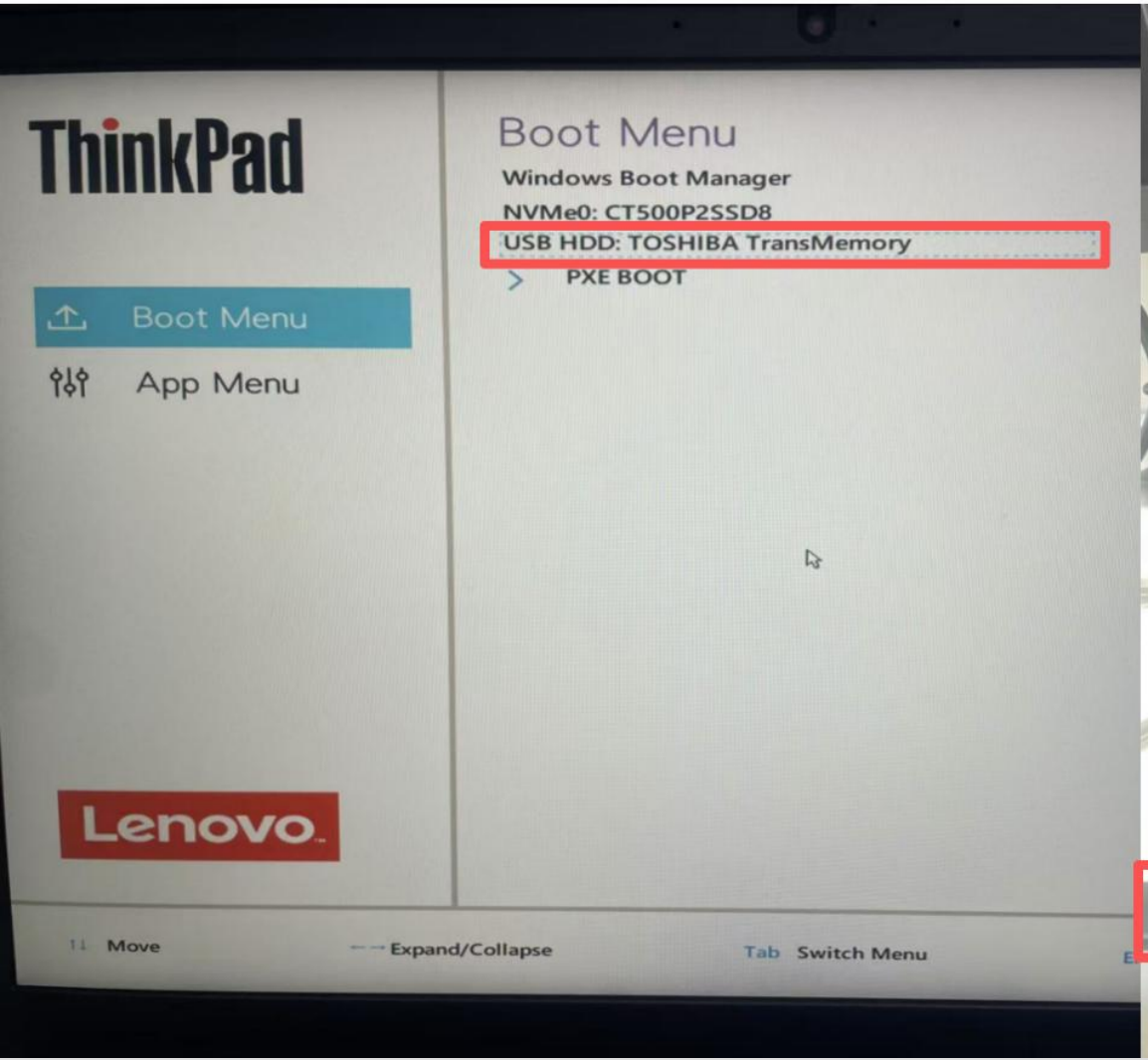
4.LBA工具支持bsrange测试；

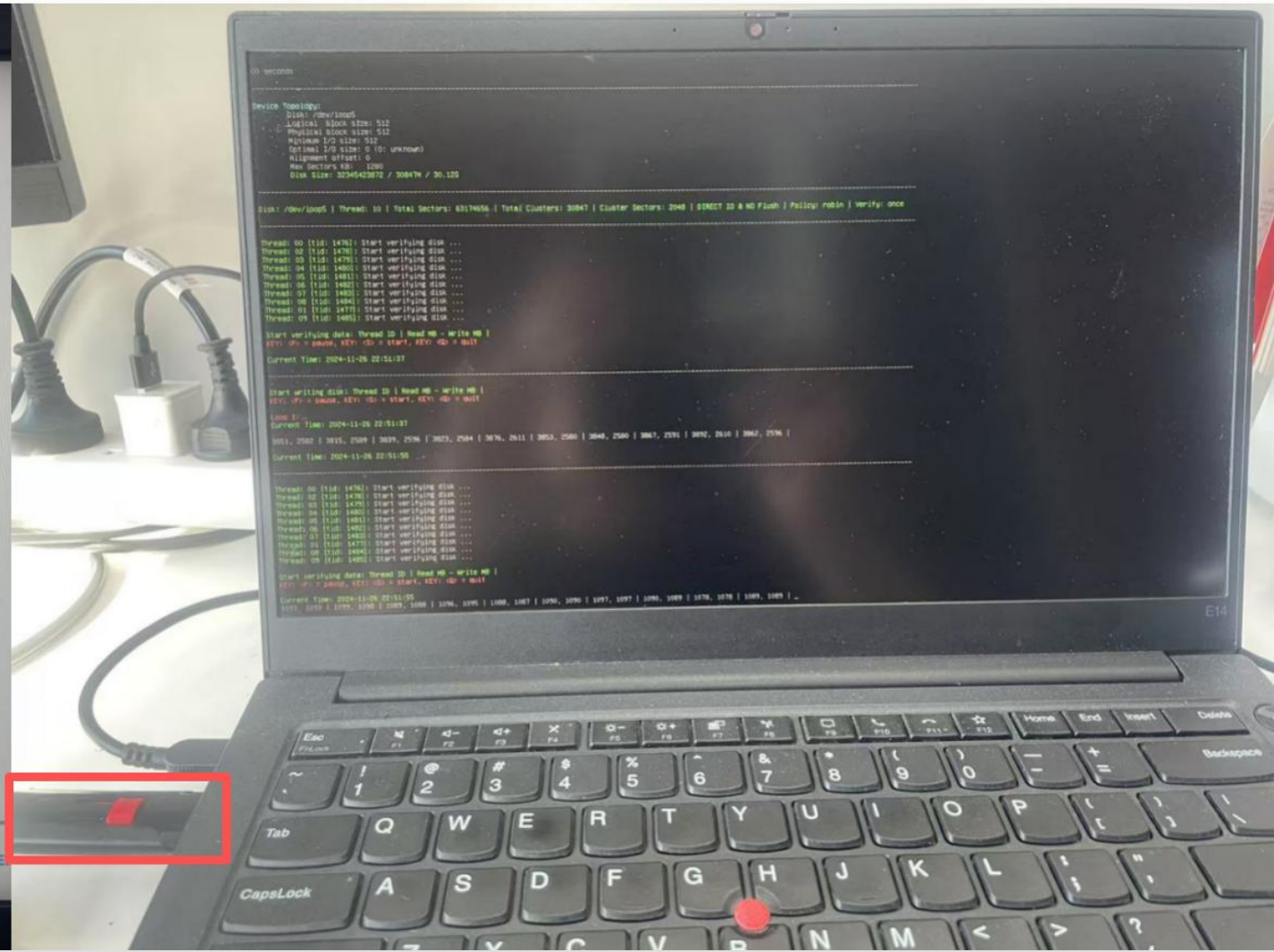
```
[root@linux lba]# hd_write_verify_dump -c -D -L 0 /dev/sdb
-------------------------------------------------------------

Device Topology:
        Disk: /dev/sdb
        Logical  block size: 512
        Physical block size: 4096
        Minimum I/O size: 262144
        Optimal I/O size: 0 (0: unknown)
        Alignment offset: 262144
        Max Sectors KB:   256
        Disk Size: 479559942144 / 457344M / 446.62G
```

右图：512e的ssd盘

物理存储稳定性测试与数据一致性校验方案

1. 用百度网盘中的镜像：hd_write_verify.disk_mem_lba.可用于烧录U盘livecd.x86_64.iso 烧录到U盘制作启动盘；
2. 应用场景：物理服务器，不需要安装操作系统，插上U盘，直接引导到存储稳定性测试与数据一致性校验系统，测试验证服务器的磁盘、内存是否有问题---非常方便；





文件存储稳定性测试与数据一致性校验方案

Curve 文件存储：百亿级文件支撑

■ 技术/案例分享 ■ 原理解读

如何支撑百亿级文件

Curve 文件系统的重要特点之一就是适用于海量文件存储，那么 Curve 文件系统如何保证可以支撑百亿级规模？如何保证在百亿级规模下的性能？从理论上来看：

- 规模方面，Curve文件存储的元数据集群，每个节点存储一定范围的 inode(比如1~10000)和 dentry，如果文件数量增多，可以进行存储节点的扩充，所以理论上规模是没有上限的。
- 性能方面，当文件数量很多时，对于单个文件的操作是没有什么差别的，但对于一些需要元数据的聚合操作会出现性能问题，比如 du (计算当前文件系统的容量), ls (获取目录下所有文件信息)等操作，需要做一定的优化来保障性能。

那实际上 Curve 文件系统的表现如何呢？

首先介绍一下文件系统的几款通用测试工具。

1. pjdfstest[1]: posix 兼容性测试。有3600+个回归测试用例，覆盖 chmod, chown, link, mkdir, mkfifo, open, rename, rmdir, symlink, truncate, unlink 等
2. mdtest[2]: 元数据性能测试。对文件或者目录进行 open/stat/close 等操作，并返回报告
3. vdbench[3]: 数据一致性测试。Vdbench 是 Oracle 编写的一款应用广泛的存储性能测试工具，既支持块设备的性能测试，也支持文件系统性能测试，在做随机写的一致性测试很方便，能实时检查出哪一个扇区出现了数据不一致
4. fio[4]: 数据性能测试。

Curve 文件系统从v2.3版本以后提供了单独压测元数据集群的方式（数据集群一般使用 Curve 块存储和 S3，所以直接对这些组件进行性能测试即可）。

1. 通过 CurveAdm[5] 搭建文件系统，在准备客户端配置文件 client.yaml[6] 时新增配置项: s3.fakeS3=true[7]。
2. 使用 mdtest，vdbench，ImagerNet数据集[8]作为数据源，测试大小文件混合场景下文件系统的稳定性和性能。

https://ask.opencurve.io/t/topic/115

可以参考网易OpenCurve测试方案：

1.使用pjdfstest测试兼容性；

2.使用mdtest测试元数据；

3.使用LBA工具测试 & 校验普通文件数据一致性和文件系统稳定性；
(替代截图中的vdbench)

可以用自动化脚本同时测试多个文件：

```
file_lba_test.sh    /path/lba_test1.raw
file_lba_test.sh    /path/lba_test2.raw
.....
file_lba_test.sh    /path/lba_testN.raw
```

目 录

01 LBA工具简介

02 LBA工具实现原理

03 LBA工具使用说明及基本功能演示

04 LBA工具典型应用场景

05 存储稳定性测试与数据一致性校验自动化测试系统演示

06 展望

展望

1. PPT篇幅有限，很多内容没有深入或者涉及，后续考虑录屏，现场演示和详细讲解LBA工具各项功能；

2. 制作多平台、集成LBA工具并可自动化测试的容器镜像；

3. 使用QT等制作图形界面版本LBA工具；

4. 开发基于spdk版本的LBA工具；

5. 对外推广：向qemu、ceph、spdk等开源社区和国内存储(OS)、云计算、备份/容灾等厂商推荐使用/试用；

6. 期望能成为存储(OS)、云计算、备份/容灾等产品稳定性测试与数据一致性校验首选工具或标准；

7. 期望能成为存储(OS)、云计算、备份/容灾等产品稳定性与数据一致性认证场景中使用的首选工具或标准；

8. 欢迎试用与合作(**培训授课、顾问咨询、LBA工具及自动化测试系统商业使用授权 + 技术支持**)

github:
https://github.com/zhangyoujia/hd_write_verify

gitee:
https://gitee.com/youplus2024/hd_write_verify

腾讯会议录屏:
https://cloud.tencent.com/developer/video/78756

百度网盘:
https://pan.baidu.com/s/1hxM1idYhY1Z3UgqprXif6Q

https://www.bilibili.com/video/BV1aX4y1L78a

提取码: LBA1

墨菲安全

zhangyoujia，你好：

久等了，您的项目检测报告已生成，点击或复制链接查看：

https://www.murphysec.com/console/report/1702199936857980928/1702199936933478400?allow=1&f=m

报告概览：

项目名称： **zhangyoujia/hd_write_verify**		2023-09-14 13:57:45
依赖开源组件数： 0	存在安全漏洞数： 0	高危及以上漏洞： 0

暂未发现相关安全风险，点击报告中的安全徽章，获取放置在您的项目主页，彰显项目安全可靠：

点击快速检测，检测您其它项目的安全风险：

快速检测

Q & A

微信1群已超200人，无法再生成新的二维码，有意请添加微信！

Thanks