

# 存储稳定性测试和数据一致性校验工具和系统

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

-- Y O U P L U S

工欲善其事，必先利其器！

## ➤ 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，为什么还要自己写一个？

A: fio和vdbench等工具，有些场景测试不了，有些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等工具能测试出的问题，LBA工具也能测试出来；

LBA工具能测试出的问题，fio和vdbench等工具不一定能测试出来；

## ➤ 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工具功能丰富、完善并且强大！

(七年+，经历几百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作品)

# 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

---

# LBA工具介绍

- 雉形源于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？

# 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V5.0版本(代码经过七年无数次的迭代、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](https://github.com/zhangyoujia/hd_write_verify))，1.5万行左右C代码。

linux版本LBA工具移植成了windows版本，两个版本使用是一样的，并且数据格式兼容。

1. windows版本lba工具写的数据，可以使用linux版本lba工具进行校验；
2. linux版本lba工具写的数据，可以使用windows版本lba工具进行校验；

windows版本与linux版本检查同一个磁盘的数据，结果一样。

11:04:11

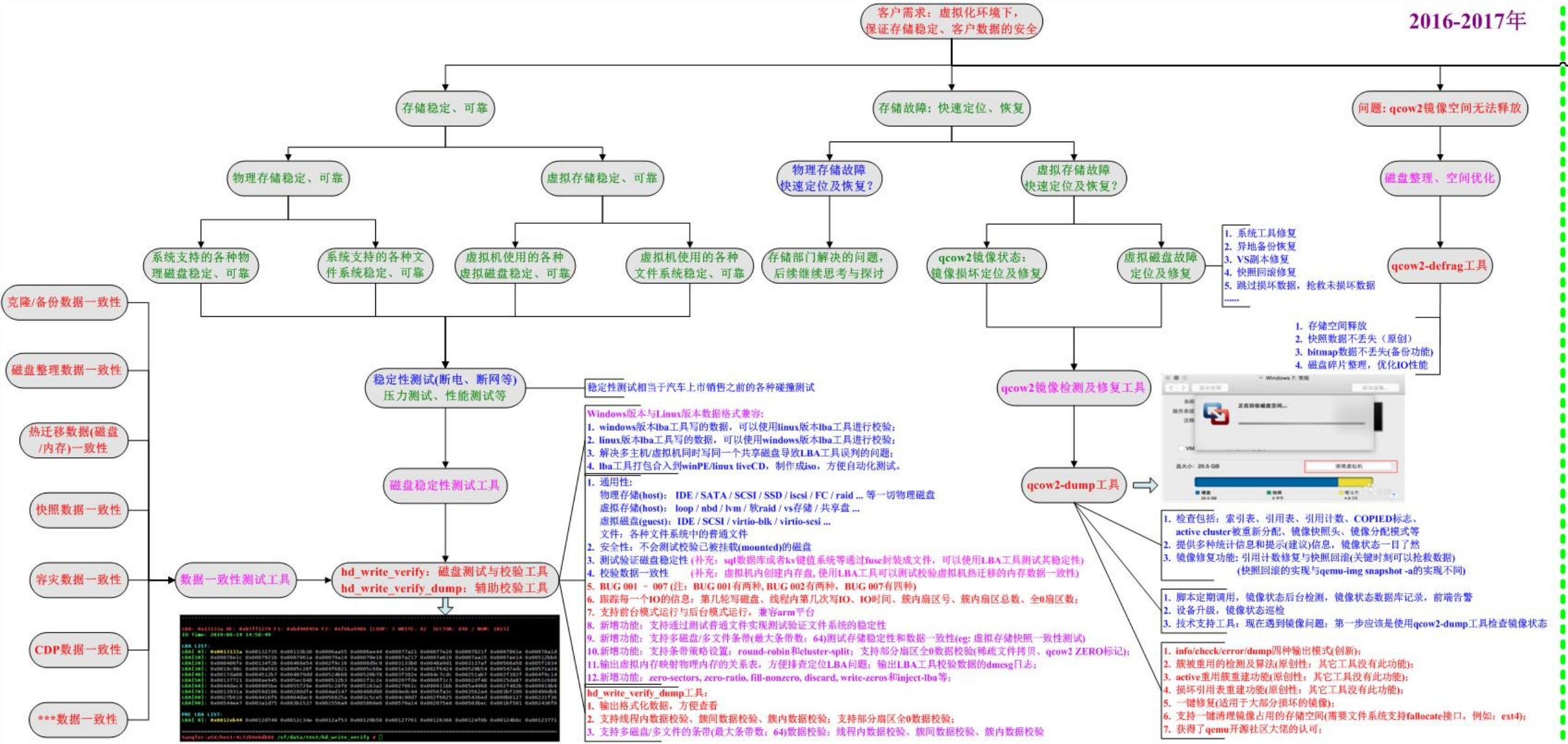
好的，比以前的windows好用很多

11:04:11

hd_write_verify > V4.0 > bin > <a href="#">linux_x86</a> > release				hd_write_verify > V4.0 > bin > <a href="#">linux_arm</a> > release				hd_write_verify > V4.0 > bin > <a href="#">windows</a> > release			
名称	修改日期	类型	大小	名称	修改日期	类型	大小	名称	修改日期	类型	大小
<a href="#">hd_write_verify</a>	2023/7/6 17:33	文件	108 KB	<a href="#">hd_write_verify</a>	2023/7/6 16:20	文件	<a href="#">hd_write_verify.exe</a>	2023/7/7 0:23	应用程序	38 KB	
<a href="#">hd_write_verify_dump</a>	2023/7/6 17:33	文件	43 KB	<a href="#">hd_write_verify_dump</a>	2023/7/6 16:20	文件	<a href="#">hd_write_verify_dump.exe</a>	2023/7/7 0:23	应用程序	18 KB	
<a href="#">md5.txt</a>	2023/7/6 17:34	TXT 文件	1 KB	<a href="#">md5.txt</a>	2023/7/6 16:20	TXT 文件	<a href="#">md5.txt</a>	2023/7/7 0:35	TXT 文件	1 KB	
名称				修改日期	类型	大小	名称				修改日期
<a href="#">hd_write_verify.with.stripe.robin.64K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:00	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.128K.整体校验+批量校验.V10.iso</a>	2023/7/7 0:59	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.256K.整体校验+批量校验.V10.iso</a>	2023/7/7 0:57	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.split.64K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:15	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.512K.整体校验+批量校验.V10.iso</a>	2023/7/7 0:55	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.1M.整体校验+批量校验.V10.iso</a>	2023/7/7 0:54	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.split.128K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:14	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.mem_lba.with.stripe.robin.1M.x86_64.V10.iso</a>	2023/7/7 0:52	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.mem_disk_lba.with.stripe.robin.1M.x86_64.V10.iso</a>	2023/7/7 0:51	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.split.256K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:13	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.8K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:02	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.4K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:04	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.split.512K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:08	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.2K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:05	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.1K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:06	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.split.1M.整体校验+批量校验.V10.iso</a>	2023/7/7 1:06	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.500K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:07	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.100K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:08	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.robin.4K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:04	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.200K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:09	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.1000K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:10	光盘映像文件	66,154 KB
<a href="#">hd_write_verify.with.stripe.robin.8K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:02	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.1500K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:11	光盘映像文件	66,154 KB	<a href="#">hd_write_verify.with.stripe.robin.2000K.整体校验+批量校验.V10.iso</a>	2023/7/7 1:12	光盘映像文件	66,154 KB

# LBA工具介绍：功能简图

2016-2017年



# 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；

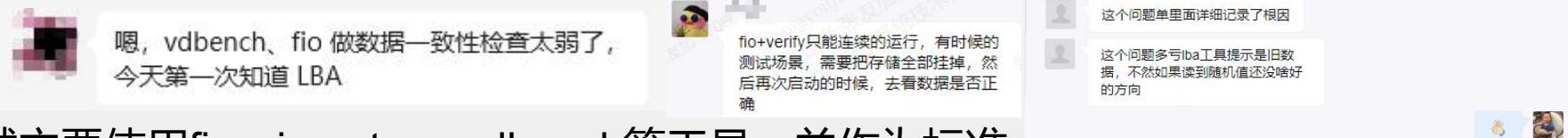
```
/*
 * HD_WRITE_VERIFY & HD_WRITE_VERIFY_DUMP v23.06
 *
 * Copyright (c) 2016-2023 YOUPLUS.
 *
 * Author: YOUPLUS <zhang_youjia@126.com>
 */

/*
 * -----
 * | 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-----| |-----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

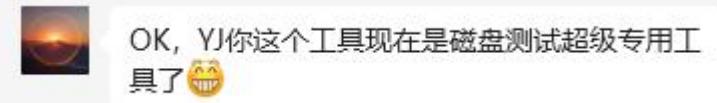
- 与Intel的刘长鹏等交流：对SPDK做块设备稳定性测试和校验数据一致性校验，主要使用fio等工具的verify功能。
- 与华为、华三、腾讯、字节跳动等的研发技术交流、面试交流，测试存储稳定性和校验数据一致性使用的也是一般的工具与方法。
- 参与云产品送检公安部认证，测试存储稳定性和校验数据一致性也没有标准工具，测试过程中推荐使用了LBA工具。
- 存储部门目前测试存储稳定性使用的是fio/vdbench的verify功能，反馈使用时有限制，使用fio/vdbench等工具测试出来的自研存储数据不一致问题不多，但使用LBA工具却测试出了很多数据不一致的问题(后续举例说明)。



- 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、拔物理盘、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。

The screenshot shows a software interface for managing defects. The main window displays a table of issues related to LBA (Logical Block Addressing). The columns include: 缺陷编号 (Defect ID), 状态 (Status), 指派给 (Assigned To), 严重性 (Severity), 发现人 (Reporter), 发现时间 (Discovery Time), 概要 (Summary), 4. 所属模块 (Module), 1. 发现版本和阶 (Version and Phase), Fix日期 (Fix Date), 修改Bug审核 (Review Bug), 老版本存在 (Old Version Exist), and 修复状态 (Status). A search bar at the top contains the text '\*lba\*'. The bottom status bar indicates 'Defect 1 of 406' and 'Server Time: 08:23 PM 06-25-19'.

缺陷编号	状态	指派给	严重性	发现人	发现时间	概要	4. 所属模块	1. 发现版本和阶	Fix日期	修改Bug审核	老版本存在
4011	Closed		4-Urgent		2016-6-25	【数据损坏】【asv】【可靠性自动化】：asv, adesk版本断电出现读错读缓存			2016-6-28		存在
4141	Open		4-Urgent		2016-6-29	【数据损坏】【可靠性自动化】执行可靠性自动化网络故障用例，两台主数据可靠性					存在
4385	Closed		2-Medium		2016-7-5	【自动化测试】有bad文件，替换主机失败，虚拟机出LBA[kvm] [xhza] 其他			2016-8-6		存在
4548	Closed		4-Urgent		2016-7-6	【数据丢失】【可靠性自动化】【adesk】：执行可靠性断电故障后，出历史遗留			2016-7-8		存在
4634	Closed		4-Urgent		2016-7-11	【数据丢失】【asv】：跑私网故障之后，出现读出错误LBA问题	数据可靠性		2016-7-13		存在
4673	halt_2		3-High		2016-7-12	【数据丢失】【adesk】：正在运行iso虚拟机，注入断电故障之后，历史遗留					存在
4727	Closed		4-Urgent		2016-7-12	【数据丢失】【asv】【可靠性自动化】：出现断电故障和断电故障之后数据可靠性			2016-7-12		存在
4828	Closed		4-Urgent		2016-7-14	【数据丢失】【asv】【可靠性自动化】：跑私网案例，迁移出现数据丢失数据可靠性			2016-7-21		存在
4924	Closed		4-Urgent		2016-7-18	【数据丢失】【asv】【可靠性自动化】：一个主机数据盘用尽，跑私网数据可靠性			2016-7-22		存在
5044	Closed		4-Urgent		2016-7-20	【数据丢失】【asv】【可靠性自动化】：跑私网故障，迁移时出LBA问题数据可靠性			2016-7-29		存在
5045	Closed_Dup		4-Urgent		2016-7-20	【数据丢失】【asv】【可靠性自动化】：执行频繁拔插磁盘案例出现读数据可靠性					
5143	Closed		4-Urgent		2016-7-20	【数据损坏】【可靠性自动化】【asv】：执行两主机频繁私网交互断开数据可靠性			2016-8-1		存在
5522	Closed		4-Urgent		2016-7-28	【LBA】【数据损坏】【LBA】进行磁盘拔插故障案例出现LBA	数据可靠性		2016-8-26		存在
5547	New		4-Urgent		2016-7-30	【数据丢失】【VMP】【可靠性自动化】：跑网络故障后出现数据丢失LE	数据可靠性				
5621	Closed		4-Urgent		2016-8-2	【LBA】【数据损坏】【LBA】【可靠性自动化】仲裁版本跑可靠性副本仲裁					
5671	New		4-Urgent		2016-8-3	【数据丢失】执行可靠性自动化网络故障用例，出现虚拟机LBA	其他		2016-9-2		否
5804	Closed		4-Urgent		2016-8-5	【LBA】【数据损坏】【LBA】【可靠性自动化】虚拟机正在迁移的是时候数据可靠性			2016-8-26		存在
6107	Closed		4-Urgent		2016-8-12	【LBA】【数据损坏】【LBA】【可靠性自动化】跑可靠性自动化网络故障副本仲裁			2016-8-26		否
6217	Closed_Dup		4-Urgent		2016-8-13	【数据损坏】【可靠性自动化】跑可靠性自动化网络故障案例，出现LBA副本仲裁					
6310	Closed		4-Urgent		2016-8-15	【LBA】【数据损坏】【文件损坏】oracle虚拟机循环进行开机和关机操作其他			2016-8-26		否
6377	Closed		4-Urgent		2016-8-17	【LBA】【数据损坏】【文件损坏】oracle RAC 虚拟机数据损坏无法ope其他			2016-8-26		否
6577	Closed		2-Medium		2016-8-20	【升级包】【脚本更新】mysqldump打错误包的脚本，不能通过升级包升级			2016-8-23		存在
6594	Closed		4-Urgent		2016-8-20	【LBA】【数据损坏】【文件损坏】共享盘是非DirectIO，如果断电会老功能			2016-8-23		存在
6992	Closed		4-Urgent		2016-8-27	【LBA】【数据损坏】【LBA】【可靠性自动化】可靠性BVT案例，出现LE	其他		2016-8-30		存在
7099	Closed		4-Urgent		2016-8-30	【LBA】【数据损坏】【文件损坏】【可靠性自动化】跑案例，虚拟机锁副本仲裁			2016-9-3		否
7146	Closed		4-Urgent		2016-8-31	【LBA】【数据损坏】【LBA】【可靠性自动化】跑主机断电故障，出现LE	其他		2016-9-8		存在
7189	Closed		4-Urgent		2016-9-1	【LBA】【数据损坏】【LBA】【可靠性自动化】执行随机故障案例，出现副本仲裁			2016-9-6		否
7282	Closed		4-Urgent		2016-9-2	【LBA】【数据损坏】【LBA】【可靠性自动化】执行随机故障出现LBA	副本仲裁		2016-9-6		否
7342	Won't Fix		4-Urgent		2016-9-3	【数据损坏】【LBA】【2主机】【可靠性自动化】两主机执行新的磁盘或其他					存在
7563	Rejected		4-Urgent		2016-9-7	【数据损坏】【NFS解耦】【可靠性自动化】：跑磁盘拔插案例，出现数NFS解耦					
7754	Rejected		4-Urgent		2016-9-12	【紧急必须项】【数据损坏】【920beta】【vs2.3】【可靠性自动化】共享盘UI					
7852	Closed		4-Urgent		2016-9-13	【数据损坏】【LBA】【可靠性自动化】磁盘拔插中克隆的虚拟机5台有其他			2016-9-19		存在
7934	Closed_Dup		4-Urgent		2016-9-17	镜像文件引用计数错误，多磁盘虚拟机出LBA-1cg回归	副本仲裁				
7947	Rejected		4-Urgent		2016-9-17	【紧急必须项】【数据损坏】【920beta】【可靠性自动化】linux版本和其他					
8194	Closed		2-Medium		2016-9-22	外置存储环境multipath_blacklist_callback.log日志非常频繁【回归】外置存储			2016-9-27		否
8290	Closed		4-Urgent		2016-9-24	【LBA】【数据损坏】【LBA】【可靠性自动化】【可靠性跟踪】克隆操作其他			2016-10-5		存在
8444	Closed		4-Urgent		2016-9-28	【LBA】【数据损坏】【镜像损坏】【HCl5.2】测试中心环境，告警日志虚拟机			2016-10-6		存在
9815	Closed		2-Medium		2016-11-1	【vmp5.0】vmp5.0跑功能自动化，出lba	其他		2016-11-7		否

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

版本稳定性运行日报 (7.1) ( 稳定运行6天；无新增AB类问题 )					
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			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， 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 件并挂载点目录下又创建2个qcow2文件，因此每个存储上 （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 影像损坏问题。
80711	虚拟机bitmap损坏 虚拟机仅执行开关机操作。			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 开关操作。主机执行过强制宕机、关机 操作。			分析中	//6.19 L2表对应的簇重用了，与快照对应的簇重用。 需要确认镜像损坏的时间点，目前根据虚拟机的镜像文 件和簇重用信息，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，判断关机条件是

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工具磁盘稳定性自动化测试：(winPE的iso运行后自动测试)

```
ca管理员: X:\windows\system32\cmd.exe
ca管理员: X:\windows\system32\cmd.exe -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 0: 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_0: 0x405000 MF: 0x00001276 F1: 0xb9ac75e0 F2: 0xb19e577e [LOOP: 6 WRITE: 4 SECTOR: 0 / NUM: 1024]
LBA_1 IO Time: 2019-01-04 10:51:13
LBA_1: 0x405400 MF: 0x00001276 F1: 0xb9ac75e0 F2: 0xb19e577e [LOOP: 6 WRITE: 296 SECTOR: 0 / NUM: 1024]

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: 0x405e00

PREV LBA: 0x405e00 | DATA | LBA: 0x405000 MF: 0x00001276 F1: 0xb9ac75e0 F2: 0xb19e577e [LOOP: 6 WRITE: 4 SECTOR: 0 / NUM: 1024]
PREV IO Time: 2019-01-04 10:43:59

LOST LBA: 0x405400 | DATA | LBA: 0x405400 MF: 0x00001276 F1: 0xb9ac75e0 F2: 0xb19e577e [LOOP: 6 WRITE: 296 SECTOR: 0 / NUM: 1024]
LBA: 0x405400 | thread_count: 10, bitmap: 0 0 0 0 0 0 0 0 0 0
LOST IO Time: 2019-01-04 10:51:13

NEXT LBA: 0x405e00 | DATA | LBA: 0x405e00 MF: 0x00001276 F1: 0xb9ac75e0 F2: 0xb19e577e [LOOP: 6 WRITE: 4 SECTOR: 0 / NUM: 1024]
```

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
RGPUUsed=465, ErrorCount=80
ResuedBlock=54
```

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

# 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/rdhms/easdbs/easdbs/incident/incdir_36804/easdbs_ora_31240_i36804.t
Wed Sep 04 08:02:27 2019
Process monitoring performing id:[cdmp_20190904080227]
Errors in file '/home/oracle/app/oracle/diag/rdhms/easdbs/easdbs/incident/incdir_36803/easdbs_ora_31240_i36803.trc':
ORA-00354: 日志损坏接近最近的日志块
ORA-00353: 日志损坏最近的日志块
ORA-00312: 联机日志 4 残留 1: '/home/oracle/app/oracle/oradata/easdbs/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/EAS_D_GXJT_STANDARD02.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/rdhms/easdbs/easdbs/trace/easdbs_ora_31240.trc' (incident=36805):
ORA-00354: 日志损坏最近的日志块
ORA-00334: 日志损坏最近的日志块
ORA-00334: 归档日志 1: '/home/oracle/app/oracle/oradata/easdbs/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/EAS_D_GXJT_STANDARD02.ORA'
Incident details in: /home/oracle/app/oracle/diag/rdhms/easdbs/easdbs/incident/incdir_36805/easdbs_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的是 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

➤ 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】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--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】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】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】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

```
Current Time: 2023-04-05 10:50:50
BUG 09711]: Thread: 0] VERIFY_SECTOR(0) DIFFER: 4 | read(120358)
LBA: 0x1f2e400 | thread_num: 10, bitmap: 1 0 0 0 0 0 0 0 0 0

CORRECT LBA[0]: 0x1f2e400 | HF: 0x00001270 F1: 0x7fa81dc9 F2: 0xc58591e4 ILOOP: 3 WRITE: 120358 STRIPE: 0 LBA: 0xf97240 SECTOR: 0 / NUM: 128 / ZERO: 421
CORRECT IO Time: 2023-04-04 12:19:30

Buffer addr: 0x7f898003c000 | Physical addr: 0x228c83000

LBA LIST:
LBA[ 0]: 0x01f2e400 0x08f31980 0x02496780 0x08629280 0x048e7580 0x07932500 0x07f6f700 0x072f5880 0x00dd5380 0x05765400
LBA[10]: 0x05243a00 0x05d01400 0x02955d00 0x021f6f80 0x0643a580 0x03e7d480 0x089df400 0x09f99200 0x08a8bd80 0x01157680
LBA[20]: 0x064d6f80 0x0038fa00 0x048f5480 0x07db5300 0x09fdff00 0x08696e00 0x08dfd380 0x078d5500 0x08207380 0x019e5a80
LBA[30]: 0x038dcc00 0x0095ef00 0x06a9ec00 0x03668880 0x00e7b500 0x042bf600 0x01328e80 0x04b0fc00 0x058cccd00 0x0100ed80
LBA[40]: 0x025c5500 0x05e84480 0x0456e580 0x013bf80 0x02980f00 0x0126f300 0x04e37980 0x00f45f00 0x07525780 0x08c1b280
LBA[50]: 0x05142280 0x00ac5b80 0x06163f80 0x01265e80 0x047df280 0x07984c80 0x00f2b200 0x08a5bb00 0x06875d00 0x01da3f80
LBA[60]: 0x065a3200 0x06645f80 0x0454b880 0x0639ab80 0x07a32b00 0x02e7fb80 0x073e7280 0x049d2e00 0x0082ef80 0x07ea3280
LBA[70]: 0x03ffd80 0x09b31f00 0x00531b80 0x04219f80 0x04e1db80 0x078d5580 0x09a15d00 0x06f47c00 0x06938780 0x0056be80
LBA[80]: 0x09194d00 0x07d5d580 0x0588f300 0x081c4f00 0x02002580 0x0380fd80 0x07d5d680 0x07d25e80 0x08c59500 0x08153980
LBA[90]: 0x05f3e580 0x005eae80 0x0656a780 0x00b74580 0x08a44d80 0x04dca800 0x0006fc00 0x010a4a80 0x0408af80 0x09235200

PRE LBA LIST:
LBA[ 0]: 0x09218e80 0x08d70980 0x040cda80 0x05c92480 0x07ac2b80 0x044c4980 0x06b30d80 0x07938f80 0x01697380 0x07a7af00

ERROR LBA[8]: 0x1f2e488 | HF: 0x00001270 F1: 0x7fa81dc9 F2: 0xc58591e4 ILOOP: 3 WRITE: 120358 STRIPE: 0 LBA: 0xf97248 SECTOR: 12 / NUM: 128 / ZERO: 421
INCORRECT IO Time: 2023-04-04 12:19:30

Buffer addr: 0x7f898003d000 | Physical addr: 0x21f324000

LBA LIST:
LBA[ 0]: 0x01f2e400 0x08f31980 0x02496780 0x08629280 0x048e7580 0x07932500 0x07f6f700 0x072f5880 0x00dd5380 0x05765400
LBA[10]: 0x05243a00 0x05d01400 0x02955d00 0x021f6f80 0x0643a580 0x03e7d480 0x089df400 0x09f99200 0x08a8bd80 0x01157680
LBA[20]: 0x064d6f80 0x0038fa00 0x048f5480 0x07db5300 0x09fdff00 0x08696e00 0x08dfd380 0x078d5500 0x08207380 0x019e5a80
LBA[30]: 0x038dcc00 0x0095ef00 0x06a9ec00 0x03668880 0x00e7b500 0x042bf600 0x01328e80 0x04b0fc00 0x058cccd00 0x0100ed80
LBA[40]: 0x025c5500 0x05e84480 0x0456e580 0x013bf80 0x02980f00 0x0126f300 0x04e37980 0x00f45f00 0x07525780 0x08c1b280
LBA[50]: 0x05142280 0x00ac5b80 0x06163f80 0x01265e80 0x047df280 0x07984c80 0x00f2b200 0x08a5bb00 0x06875d00 0x01da3f80
LBA[60]: 0x065a3200 0x06645f80 0x0454b880 0x0639ab80 0x07a32b00 0x02e7fb80 0x073e7280 0x049d2e00 0x0082ef80 0x07ea3280
LBA[70]: 0x03ffd80 0x09b31f00 0x00531b80 0x04219f80 0x04e1db80 0x078d5580 0x09a15d00 0x06f47c00 0x06938780 0x0056be80
LBA[80]: 0x09194d00 0x07d5d580 0x0588f300 0x081c4f00 0x02002580 0x0380fd80 0x07d5d680 0x07d25e80 0x08c59500 0x08153980
LBA[90]: 0x05f3e580 0x005eae80 0x0656a780 0x00b74580 0x08a44d80 0x04dca800 0x0006fc00 0x010a4a80 0x0408af80 0x09235200

PRE LBA LIST:
LBA[ 0]: 0x09218e80 0x08d70980 0x040cda80 0x05c92480 0x07ac2b80 0x044c4980 0x06b30d80 0x07938f80 0x01697380 0x07a7af00

CORRECT LBA[12]: 0x1f2e48c | HF: 0x00001270 F1: 0x7fa81dc9 F2: 0xc58591e4 ILOOP: 3 WRITE: 120358 STRIPE: 0 LBA: 0xf9724c SECTOR: 12 / NUM: 128 / ZERO: 421
CORRECT IO Time: 2023-04-04 12:19:30

Buffer addr: 0x7f898003d800 | Physical addr: 0x21f324800
```

# 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o log 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: 0x2c1c082 | HF: 0x00001279 F1: 0x010742B3 F2: 0xe177ddc0 LLOOP: 5 WRITE: 16138 STRIPE: 1 LBA: 0x160e002 SECTOR: 2 × NUM: 128 × ZERO
IO Time: 2023-04-03 17:12:32

Buffer addr: 0x22ba400 | Physical addr: 0x42457b400

LBA LIST:
LBA[ 0]: 0x02c1c080 0x0664d600 0x06903980 0x0830e280 0x0281c600 0x010df880 0x04164c80 0x06b7b000 0x046e6780 0x00e18300
LBA[10]: 0x092f1500 0x0122b600 0x0513a000 0x07ae8180 0x03e21e00 0x06800200 0x0007c100 0x07c60100 0x041d4000 0x06fe4000
LBA[20]: 0x09909980 0x07cd2200 0x09e57580 0x092eed00 0x021ce980 0x0327fd00 0x035a4900 0x09a79f80 0x022a5880 0x071df900
LBA[30]: 0x06c40c80 0x07b20480 0x07dc0280 0x0655f980 0x07017d00 0x07d67400 0x047d1e00 0x01bd900 0x09d52b00 0x02f5a000
LBA[40]: 0x0091d680 0x04710800 0x01436600 0x03976080 0x004cb880 0x037e8880 0x00fc4000 0x08b7d000 0x016d4880 0x030a1880
LBA[50]: 0x075e5800 0x042c0080 0x029aa000 0x0703b000 0x04bbe800 0x0935c080 0x05269b80 0x034f3500 0x0270ad00 0x02cef500
LBA[60]: 0x061e6a00 0x07911500 0x07594e00 0x089adb00 0x05a59780 0x00292c80 0x013da400 0x01a16a80 0x03ecac80 0x07c7e100
LBA[70]: 0x043e3600 0x09a2b800 0x08777600 0x0894c200 0x01946000 0x0855ac00 0x06851080 0x00f1e080 0x05ee8e00 0x05e7b900
LBA[80]: 0x01925b80 0x0285e580 0x07d8b800 0x06b68500 0x018a6100 0x06426800 0x09709800 0x00616800 0x05a0c000 0x0873b380
LBA[90]: 0x00ba0680 0x08251300 0x048de300 0x078f9a00 0x050cd900 0x077a9200 0x08681b00 0x05d01c00 0x06d74080 0x066ecc80

PRE LBA LIST:
LBA[ 0]: 0x07cd6000 0x01b74a00 0x01b55900 0x075df080 0x06ff4080 0x095f8800 0x00647800 0x06462c80 0x09ca6800 0x0219e400
```

有个数据结构强转了一下，转换出来的值相同或者相差1个字节，导致分配到了相同的buffer来存放数据，两次不同的row IO都使用该buffer放数据的话，会被覆盖掉，要非常巧合才能出现，比较偶现。

17:33  
啥原因呀？有没有什么必现的场景  
我看目前20台虚机就两台出这个问题

```
LBA: 0x2c1c003 | HF: 0x83000012 F1: 0x00010742 F2: 0x00e177dd LLOOP: 4261412864 WRITE: 830000143 STRIPE: 1 LBA: 0x160e003 SECTOR: 30976 × NUM: 0 × ZERO: 16777216
01
IO Time: 385877868-2063597569-50331648 50331648:285212672:201326592

Buffer addr: 0x22ba600 | Physical addr: 0x42457b600

LBA LIST:
LBA[ 0]: 0x0002c1c0 0x000656486 0x00069039 0x000830e2 0x000281c6 0x00010f8 0x0004154c 0x0006b7b0 0x00046e67 0x0000e183
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]: 0x00007cd600 0x00001b74a 0x00001b559 0x000075d10 0x00006ff40 0x000095f88 0x000006478 0x00006162c 0x00009ca68 0x000219e1

=====
Current Time: 2023-04-04 09:48:01
```

# 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

```

BUG 004: [Thread: 51 DATA LOST | read[288]
PREV LBA: 0x359ec00 | LOST LBA: 0x20c5c00 | NEXT LBA: 0x1d3c800
PREV LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2ebd00 [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV IO Time: 2023-05-13 11:20:53

LOST LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
LOST IO Time: 2023-05-13 11:22:55

NEXT LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2ebd00 [LOOP: 5 WRITE: 288 SECTOR: 0 / NUM: 1024 / ZERO: 208]
NEXT IO Time: 2023-05-13 11:20:54

LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2ebd00 [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
IO Time: 2023-05-13 11:20:53

Buffer addr: 0x7f6faadd5000 | Physical addr: 0x422516000

LBA LIST:
LBA[ 0]: 0x359ec00 0x020c5c00 0x01d3c800 0x032a8c00 0x00fa88000 0x0043a800 0x042c5c00 0x02d9e400 0x03de2c00 0x03ab1000
LBA[ 1]: 0x0139b400 0x0215ec00 0x039b1000 0x01557c00 0x02055800 0x03520800 0x0398a000 0x0029e000 0x0442a000 0x020240800
LBA[ 20]: 0x00225000 0x03af2400 0x04f8a000 0x041fb000 0x02c1c000 0x03bd1c00 0x001a1000 0x00045c00 0x0033e400 0x0336f000
LBA[ 30]: 0x02873400 0x030f1c00 0x02aec800 0x01864400 0x03410000 0x03b9c00 0x044d4000 0x02507c00 0x01bd0000 0x00ae4400
LBA[ 40]: 0x0339e000 0x015d8400 0x03232000 0x02a80000 0x03520800 0x0398a000 0x0029e000 0x0442a000 0x020240800
LBA[ 50]: 0x00500000 0x0139c400 0x02f5ec00 0x039b1000 0x01557c00 0x02055800 0x03520800 0x0282c000 0x009cc00 0x03128c00
LBA[ 60]: 0x03a3c400 0x04f7f6400 0x04040400 0x000bc800 0x03ddc800 0x0385f000 0x01c4a800 0x03dac00 0x00163000
LBA[ 70]: 0x0496d900 0x04c1c800 0x03518000 0x03577800 0x021c1400 0x01573800 0x03d08800 0x00fb8a800 0x04925000 0x0104e800
LBA[ 80]: 0x028c0000 0x01e32c00 0x0455cc00 0x03a0cc00 0x019a9400 0x02f0c000 0x01026000 0x000bf000 0x03ac1000 0x03c5b800
LBA[ 90]: 0x02b02800 0x02aae400 0x00cee800 0x00d56000 0x0011c000 0x03ac4400 0x01137000 0x0332f000 0x04f5f000 0x02f61000

PRE LBA LIST:
LBA[ 0]: 0x023f1c00 0x01fb0400 0x00277800 0x04b48000 0x01281000 0x0332d800 0x00c4d800 0x00534400 0x01964400

LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
IO Time: 2023-05-13 11:12:55

Buffer addr: 0x7f6faae57000 | Physical addr: 0x41f880000

LBA LIST:
LBA[ 0]: 0x020c5c00 0x01ccb400 0x0266cc00 0x040cb00 0x06995400 0x011c3800 0x019fc000 0x004ec000 0x00857c00 0x013sec00
LBA[ 1]: 0x004ca400 0x03b6b400 0x03028000 0x09c0900 0x06a4b00 0x03108400 0x0496d800 0x01605400 0x00255000 0x03af2400
LBA[ 20]: 0x00454400 0x01dc1000 0x0401f1000 0x0011fc000 0x04254000 0x0142c00 0x01474c00 0x04274900 0x02c5cf00 0x03ec3100
LBA[ 30]: 0x04947a00 0x0474b1000 0x04944500 0x01010000 0x0412f000 0x01472c00 0x01477c00 0x0427a000 0x02c51200
LBA[ 40]: 0x00437a00 0x0484f400 0x0498f800 0x022a7800 0x04a5e000 0x0137d000 0x04941000 0x0137f800 0x0137fc00
LBA[ 50]: 0x04843c00 0x04197c00 0x03f22c00 0x03338000 0x04dbd000 0x01814000 0x0494f100 0x0137f400 0x0137fc00
LBA[ 60]: 0x001c5200 0x04f44b00 0x0494b000 0x021b4000 0x04031000 0x03dc3000 0x0101c000 0x0101a000 0x009163000
LBA[ 70]: 0x00305100 0x02d3000 0x0239c00 0x03398000 0x04091000 0x04158000 0x02351400 0x00497c00 0x02xf1000
LBA[ 80]: 0x021d3c00 0x045ec000 0x02d6cc00 0x041fc500 0x01fcf500 0x01fcf500 0x049f7800 0x017d8000 0x013fc400 0x02dd6400

PRE LBA LIST:
LBA[ 0]: 0x02fc0800 0x04282400 0x04fd1000 0x0084c400 0x045bdc00 0x03ccbc00 0x014bd400 0x01360000 0x02b2800 0x008a3400

NEXT LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2ebd00 [LOOP: 5 WRITE: 280 SECTOR: 0 / NUM: 1024 / ZERO: 208]
NEXT IO Time: 2023-05-13 11:20:54

Buffer addr: 0x7f6fa810f000 | Physical addr: 0x422a5090

LBA LIST:
LBA[ 0]: 0x0133c800 0x00fa8800 0x0043a800 0x042c5c00 0x02d9e400 0x03de2c00 0x03ab1000 0x0139c400 0x013dc800
LBA[ 1]: 0x004ca400 0x03b6b400 0x03028000 0x09c0900 0x06a4b00 0x03108400 0x0496d800 0x01605400 0x00255000 0x03af2400
LBA[ 20]: 0x00454400 0x01dc1000 0x0401f1000 0x0011fc000 0x04254000 0x0142c00 0x01474c00 0x04274900 0x02c5cf00 0x03ec3100
LBA[ 30]: 0x04947a00 0x0474b1000 0x04944500 0x01010000 0x0412f000 0x01472c00 0x01477c00 0x0427a000 0x02c51200
LBA[ 40]: 0x00437a00 0x0484f400 0x0498f800 0x022a7800 0x04a5e000 0x0137d000 0x04941000 0x0137f800 0x0137fc00
LBA[ 50]: 0x04843c00 0x04197c00 0x03f22c00 0x03338000 0x04dbd000 0x01814000 0x0494f100 0x0137f400 0x0137fc00
LBA[ 60]: 0x001c5200 0x04f44b00 0x0494b000 0x021b4000 0x04031000 0x03dc3000 0x0101c000 0x0101a000 0x009163000
LBA[ 70]: 0x00305100 0x02d3000 0x0239c00 0x03398000 0x04091000 0x04158000 0x02351400 0x00497c00 0x02xf1000
LBA[ 80]: 0x021d3c00 0x045ec000 0x02d6cc00 0x041fc500 0x01fcf500 0x01fcf500 0x049f7800 0x017d8000 0x013fc400 0x02dd6400

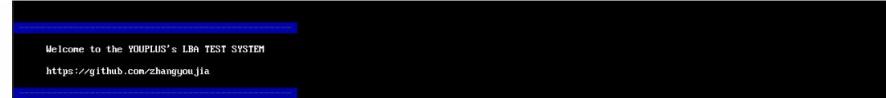
PRE LBA LIST:
LBA[ 0]: 0x02fc0800 0x0359c800 0x0231c100 0x01fb0400 0x00277800 0x04848000 0x01281000 0x0334d800 0x03715c00 0x008c3800

```

数据不一致问题：数据丢失 (10)



还没具体写原因，目前结论是，下电前的部分io对于上层写入成功，对于底层还在元数据在部分节点树上未提交，上电后判断数据树的状态有问题，导致未提交的节点将数据丢失



```

Welcome to the YOOLPLUS's LBA TEST SYSTEM
https://github.com/zhangyoujia

[none] Login: root (automatic login)
echo never > /sys/kernel/mm/transparent_hugepage/enabled
hd_write_verify -c -R -B -S -3 1024 -U all -T 10 -L 102400 -P split -I /dev/sda
Device Test...
Disk: /dev/sda | Thread: 10 | Total Sectors: 83886080 | Total Clusters: 81920 | Sectors of Cluster: 1024 | DIRECT IO & NO Flush | Verify: 5 | Notify: 0
Current Time: 2023-05-13 11:57:40
Thread 8 (tid: 6831): Starting check disk ...
Thread 9 (tid: 6841): Starting check disk ...
Thread 0 (tid: 6751): Starting check disk ...
Thread 3 (tid: 6701): Starting check disk ...
Thread 2 (tid: 6771): Starting check disk ...
Thread 6 (tid: 6881): Starting check disk ...
Thread 7 (tid: 6791): Starting check disk ...
Thread 5 (tid: 6601): Starting check disk ...
Thread 4 (tid: 6711): Starting check disk ...
Thread 8 (tid: 6831): Starting check disk ...
Thread 9 (tid: 6841): Starting check disk ...
Current Time: 2023-05-13 11:57:55
[none] Login: root (automatic login)
PREV LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2ebd00 [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV LBA IO Time: 2023-05-13 11:20:53
NEXT LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
NEXT LBA IO Time: 2023-05-13 11:12:55

```

```

Current Time: 2023-05-13 11:57:55
[none] Login: root (automatic login)
PREV LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV LBA IO Time: 2023-05-13 11:20:53
NEXT LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
NEXT LBA IO Time: 2023-05-13 11:12:55

```

```

Current Time: 2023-05-13 11:57:55
[none] Login: root (automatic login)
PREV LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV LBA IO Time: 2023-05-13 11:20:53
NEXT LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
NEXT LBA IO Time: 2023-05-13 11:12:55

```

```

[none] Login: root (automatic login)
PREV LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
PREV LBA IO Time: 2023-05-13 11:12:55
NEXT LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
NEXT LBA IO Time: 2023-05-13 11:20:54

```

```

[none] Login: root (automatic login)
PREV LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV LBA IO Time: 2023-05-13 11:20:54
NEXT LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 5 WRITE: 280 SECTOR: 0 / NUM: 1024 / ZERO: 208]
NEXT LBA IO Time: 2023-05-13 11:20:54

```

```

[none] Login: root (automatic login)
PREV LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 5 WRITE: 280 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV LBA IO Time: 2023-05-13 11:20:54
Current Time: 2023-05-13 11:57:55
pause_and_exit_work: Press KEY: <Q> to exit!

```

# 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/4eae947-2adb-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/4eae947-2adb-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: 7 tgt: 7 rebuild: 0 hyper: 0

```
node81 INFO 2023/05/25 20:07:19 [gid:137] daos_engine:1 Cluster map 4eae947 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\_IN (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
```



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。

Welcome to the YOUPlus's LBA TESTING SYSTEM

<https://github.com/zhangyoujia/>

```
YOUPlus Login root (automatic login)
root:~$ login: root: 20 16:50:41 GMT+0 2023 on ttys0
0:0 NO MEDIUM RESTRICTED
root@YOLB: /var/iso/tools # lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 1:0 1 8M 0 rom /var/iso
vda 2:0 0 20G 0 disk
root@YOLB: /var/iso/tools # hd_verify_dump -c -D -T 2 /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 (io unknown)
Alignment offset: 0
Disk Size: 21474836480 / 20480M / 20.00G
Sector_Per_Cluster: 2048
Current Time: 2023-05-30 17:36:01
=====
WARNING: [Thread: 2] WRITE SECTORS INCONSISTENT | read[146] | filename: /dev/vda
CORRECT LBA[0]: 0x1b2000 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 2 WRITE: 146 SECTOR: 0 / NUM: 2048 / ZERO: 72]
CORRECT IO Time: 2023-05-30 15:19:31
ERROR LBA[0]: 0x1b2008 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 1 WRITE: 566 SECTOR: 8 / NUM: 2048 / ZERO: 664]
INCONSISTENT IO Time: 2023-05-30 15:19:35
CORRECT LBA[0]: 0x1b2000 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 2 WRITE: 146 SECTOR: 0 / NUM: 2048 / ZERO: 72]
CORRECT IO Time: 2023-05-30 15:19:31
=====
BUG 007[1] LAST IO: [Thread: 2] WRITE SECTOR[0] DIFFER: 0 | read[146] | filename: /dev/vda
hd_verify_dump -c -D -L 0x00178000 disk/file
hd_verify_dump -c -D -L 0x00180000 disk/file
CORRECT LBA[0]: 0x1b2000 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 2 WRITE: 146 SECTOR: 0 / NUM: 2048 / ZERO: 72]
CORRECT IO Time: 2023-05-30 15:19:31
ERROR LBA[0]: 0x1b2008 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 1 WRITE: 566 SECTOR: 8 / NUM: 2048 / ZERO: 664]
INCONSISTENT IO Time: 2023-05-30 15:19:35
CORRECT LBA[0]: 0x1b2010 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 2 WRITE: 146 SECTOR: 16 / NUM: 2048 / ZERO: 72]
CORRECT IO Time: 2023-05-30 15:19:31
=====
LBA: 0x1b2007 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 2 WRITE: 146 SECTOR: 7 / NUM: 2048 / ZERO: 72]
IO Time: 2023-05-30 15:19:31
Buffer addr: 0x7fb04136e00 | Physical addr: 0x422c0e00
LBA LIST:
LBA[ 0]: 0x001b2000 0x0181b000 0x01a1c000 0x021b8000 0x022c8000 0x03c7500 0x01a5e000 0x01bcb000 0x00b34800
LBA[ 1]: 0x0035e000 0x0053e000 0x01a1b000 0x01a1b000 0x01a1b000 0x01a1b000 0x00532000 0x00c3d000 0x00a42000
LBA[ 2]: 0x0174f000 0x00440000 0x002d3000 0x00b2800 0x00e2000 0x01b2000 0x00635000 0x01c95000 0x01d42000 0x00646000
LBA[ 3]: 0x021ac000 0x00e00000 0x00d4d000 0x00223000 0x00d0c000 0x01b24000 0x027e2800 0x001d5000 0x01540000 0x01982000
LBA[ 4]: 0x001b1800 0x01d65000 0x02010400 0x01556000 0x017c800 0x02165000 0x01565000 0x01d37000 0x01628000
LBA[ 5]: 0x001b1800 0x01d65000 0x02010400 0x01556000 0x017c800 0x02165000 0x01565000 0x01d37000 0x01628000
LBA[ 6]: 0x00551800 0x00840000 0x00f31a000 0x0105a000 0x00432000 0x00202300 0x005552000 0x00580000 0x03590000
LBA[ 7]: 0x01cf7000 0x00520000 0x0165e000 0x00210000 0x00408000 0x00123000 0x007f4000 0x00288000 0x01303800
LBA[ 8]: 0x00426000 0x0142e000 0x00903000 0x027d0800 0x01203000 0x025f8000 0x027e6000 0x00763000 0x016c8000 0x01d42000
LBA[ 9]: 0x0198c000 0x0223a000 0x01b0000 0x01010000 0x024a0000 0x018e4000 0x022cc000 0x02416000
PRE LBA LIST:
LBA[ 0]: 0x00317800 0x01652800 0x025d6000 0x0026c000 0x0031c000 0x005a8000 0x0148c800 0x0134c800 0x00c5800
=====
LBA: 0x1b2003 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 1 WRITE: 566 SECTOR: 8 / NUM: 2048 / ZERO: 664]
IO Time: 2023-05-30 15:09:35
Buffer addr: 0x7fb04137000 | Physical addr: 0x425a1300
LBA LIST:
LBA[ 0]: 0x001b2000 0x0181b000 0x01a1c000 0x021b8000 0x022c8000 0x03c7500 0x01a5e000 0x01bcb000 0x00b34800
LBA[ 1]: 0x0035e000 0x0053e000 0x01a1b000 0x01a1b000 0x01a1b000 0x01a1b000 0x00532000 0x00c3d000 0x00a42000
LBA[ 2]: 0x01901800 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 3]: 0x01901800 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 4]: 0x0145e000 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 5]: 0x0145e000 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 6]: 0x0145e000 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 7]: 0x0145e000 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 8]: 0x0145e000 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
LBA[ 9]: 0x0145e000 0x00454000 0x0080b000 0x0080b000 0x00440000 0x027c0000 0x00051800 0x00218000 0x01840000
PRE LBA LIST:
LBA[ 0]: 0x00317800 0x01652800 0x025d6000 0x0026c000 0x0031c000 0x005a8000 0x0148c800 0x0134c800
=====
LBA: 0x1b2010 | HF: 0x08001272 F1: 0x639bdb3c F2: 0xaeecd94c [LOOP: 2 WRITE: 146 SECTOR: 16 / NUM: 2048 / ZERO: 72]
IO Time: 2023-05-30 15:19:31
Buffer addr: 0x7fb04138000 | Physical addr: 0x42369100
LBA LIST:
LBA[ 0]: 0x001b2000 0x0181b000 0x01a1c000 0x021b8000 0x022c8000 0x03c7500 0x01a5e000 0x01bcb000 0x00b34800
LBA[ 1]: 0x0035e000 0x0053e000 0x01a1b000 0x01a1b000 0x01a1b000 0x01a1b000 0x00532000 0x00c3d000 0x00a42000
LBA[ 2]: 0x0174f000 0x00440000 0x002d3000 0x00b2800 0x00e2000 0x01b2000 0x00635000 0x01c95000 0x01d42000 0x00646000
LBA[ 3]: 0x021ac000 0x00e00000 0x00d4d000 0x00223000 0x00d0c000 0x01b24000 0x027e2800 0x001d5000 0x01540000 0x01982000
LBA[ 4]: 0x001b1800 0x01d65000 0x02010400 0x01556000 0x017c800 0x02165000 0x01565000 0x01d37000 0x01628000
LBA[ 5]: 0x001b1800 0x01d65000 0x02010400 0x01556000 0x017c800 0x02165000 0x01565000 0x01d37000 0x01628000
LBA[ 6]: 0x00551800 0x00840000 0x00f31a000 0x0105a000 0x00432000 0x00202300 0x005552000 0x00580000 0x03590000
LBA[ 7]: 0x01cf7000 0x00520000 0x0165e000 0x00210000 0x00408000 0x00123000 0x007f4000 0x00288000 0x01303800
LBA[ 8]: 0x00426000 0x0142e000 0x00903000 0x027d0800 0x01203000 0x025f8000 0x027e6000 0x00763000 0x016c8000 0x01d42000
LBA[ 9]: 0x0198c000 0x0223a000 0x01b0000 0x01010000 0x024a0000 0x018e4000 0x022cc000 0x02416000
PRE LBA LIST:
LBA[ 0]: 0x00317800 0x01652800 0x025d6000 0x0026c000 0x0031c000 0x005a8000 0x0148c800 0x0134c800
=====
current Time: 2023-05-30 17:36:02
root@YOLB: /var/iso/tools #
```

# 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)。

```

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

```

```

Welcome to the YOUPLUS's LBA TESTING SYSTEM
https://github.com/zhangyouJia/
YOUPLUS login: root (automatic login)
last login: Wed Jun 7 16:19:15 CST+8 2023 on ttys0
echo never > /sys/kernel/mm/transparent_hugepage/enabled
hdurits_verify -c -D -K -R 33 -S 2048 -V all -T 10 -L 102400 -P split -I /dev/vda
Device Technology:
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: 42349672360 / 40960M / 40.00G
Disk: /dev/vda | Thread: 10 | Total Sectors: 83886080 | Total Clusters: 40960 | Sectors of Cluster: 2048 | DIRECT IO & NO Flush | Verify: 5 | Notify: 0
Current Time: 2023-06-07 16:19:15
Thread 1 [tid: 1275]: Starting check disk ...
Thread 6 [tid: 1276]: Starting check disk ...
Thread 5 [tid: 1278]: Starting check disk ...
Thread 0 [tid: 1273]: Starting check disk ...
Thread 3 [tid: 1277]: Starting check disk ...
Thread 9 [tid: 1282]: Starting check disk ...
Thread 7 [tid: 1280]: Starting check disk ...
Thread 4 [tid: 1277]: Starting check disk ...
Thread 2 [tid: 1279]: Starting check disk ...
Thread 1 [tid: 1274]: Starting check disk ...

Starting write disk: Thread ID | Read MB - Write MB |
KEY: <P> = pause, KEY: <S> = start, KEY: <Q> = quit
Disk: /dev/vda
Current Time: 2023-06-07 16:19:15
LBA: 0x4108800 | thread_num: 10, bffmap: 0 0 0 0 0 1 0 0 0
LAST CORRECT WRITE:
LBA: 0x240000 | HF: 0x08001275 F1: 0xb703fdb8 F2: 0x28413cac [LOOP: 1 WRITE: 212 SECTOR: 0 / NUM: 2048 / ZERO: 90]
LAST WRITE TO TIME: 2023-06-07 16:14:01
buffer addr: 0x7f41d9de4000 | Physical addr: 0x41bb19000
LBA LIST:
LBA[ 0]: 0x00240000 0x00560000 0x01c0b000 0x03f4a000 0x03630000 0x02806000 0x00524000 0x000e64000 0x0146f800 0x01930800
LBA[10]: 0x01857000 0x035d4000 0x0146a000 0x0133b000 0x00343000 0x00343000 0x002d6000 0x00143000 0x01930800 0x00000000
LBA[20]: 0x01930800 0x0146a000 0x0133b000 0x00343000 0x00343000 0x002d6000 0x00143000 0x01930800 0x01072600
LBA[30]: 0x04353000 0x049c5000 0x02a65000 0x02c23000 0x0085000 0x0193000 0x002a8000 0x04933000 0x04128000
LBA[40]: 0x02234000 0x0281b000 0x0165000 0x0143000 0x0252d000 0x0187d000 0x03d8a000 0x018fc000 0x04181000 0x04129000
LBA[50]: 0x00046000 0x019c8000 0x0c7c000 0x0458800 0x02027800 0x04365000 0x0371d000 0x04c30000 0x01854800 0x04ce8000
LBA[60]: 0x00046000 0x019c8000 0x0c7c000 0x0458800 0x02027800 0x04365000 0x0371d000 0x04c30000 0x01854800 0x04ce8000
LBA[70]: 0x04876000 0x04dd5000 0x02546000 0x01414000 0x03d65000 0x01342000 0x00960800 0x00180000 0x00c0d6000
LBA[80]: 0x03715000 0x03359000 0x041e1000 0x02440000 0x01b15000 0x02432000 0x04257000 0x00800000 0x04ad7800 0x000c0000 0x0a596000
PRE LBA LIST:
LBA[ 0]: 0x011ff000 0x04959000 0x02260000 0x02a34000 0x01d8000 0x034f800 0x02ee0000 0x04d6c000 0x0224c000 0x02440000

RANDOM VERIFY:
LBA: 0x4108800 | HF: 0x08001275 F1: 0xb703fdb8 F2: 0x28413cac [LOOP: 1 WRITE: 99 SECTOR: 0 / NUM: 2048 / ZERO: 90]
RANDOM VERIFY TO TIME: 2023-06-07 16:13:37
buffer addr: 0x7f41d88a0000 | Physical addr: 0x420659000
LBA LIST:
LBA[ 0]: 0x004108000 0x01f0d000 0x0179a000 0x043d600 0x01168000 0x00b59000 0x00251000 0x047a7000 0x00209000 0x03197000
LBA[10]: 0x030a70000 0x00d40000 0x04b7000 0x04443000 0x01c5e000 0x00444b000 0x0411e000 0x04173000 0x0114000 0x03e21000
LBA[20]: 0x02193000 0x00d56000 0x0121f000 0x02d14000 0x04947e000 0x00b29000 0x002c5000 0x03287000 0x045f1000
LBA[30]: 0x02e6e4000 0x03322000 0x00ef6000 0x01d1f000 0x02156000 0x04989000 0x02116000 0x01f6800 0x03b0000 0x018f0000
LBA[40]: 0x01129000 0x00935000 0x02223000 0x012b4000 0x04954000 0x02107000 0x04989000 0x02116000 0x01f6800 0x03b0000 0x018f0000
LBA[50]: 0x00477000 0x00771000 0x02588000 0x02253000 0x047c2000 0x03b3e800 0x02557000 0x00863800 0x04254000 0x03295000
LBA[60]: 0x0426c5000 0x00e559000 0x01c53000 0x014b5000 0x04b58000 0x00188000 0x03923000 0x04b9800 0x01628000 0x04f00000 0x01565800
LBA[70]: 0x0462c0000 0x002020000 0x014b6000 0x012b8000 0x04b93000 0x01700000 0x037d8000 0x04200000 0x03618000 0x04e5f000
LBA[80]: 0x036380000 0x002d24000 0x01490000 0x01e10000 0x045f8000 0x02780000 0x045f7000 0x03919000 0x01565800
PRE LBA LIST:
LBA[ 0]: 0x01168000 0x04608000 0x010de000 0x02280000 0x04e5d000 0x01a33000 0x00c5f1000 0x04aa4000 0x014dc800 0x03800000

DETECT READ ERROR:
[HF: 0x00000000] HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
DETECT ERROR TO TIME: 1900-01-00 00:00:00
buffer addr: 0x7f41d8961000 | Physical addr: 0x42353d000
LBA LIST:
LBA[ 0]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
PRE LBA LIST:
LBA[ 0]: 0x000000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Current Time: 2023-06-07 16:14:02
pause_and_exit_main: Press KEY: <Q> to exit!
```

# 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

The screenshot shows a hex editor interface with two panes. The left pane displays a sequence of bytes, primarily in red, indicating errors. The right pane is a terminal window showing log output from a storage driver or monitor. Key log entries include:

- Current Time: 2023-07-26 15:50:21
- LBA: 0x2bb240 | thread\_num: 10, bitmap: 0 0 1 0 0 0 0 0 0
- LAST CORRECT WRITE:
- LBA: 0x18300 | HF: 0x00081272 F1: 0x63aac4e6 F2: 0x2d5e04f8 [LOOP: 1 WRITE: 6731 SECTOR: 0 / NUM: 0 / ZERO: 0]
- LAST WRITE ID Time: 2023-07-26 15:50:21
- Buffer addr: 0x7f7fb4145000 | Physical addr: 0x347e000
- LBA LIST:

  - LBA[0]: 0x00018300 0x03343f60 0x02c25868 0x0079e8b0 0x00723bc0 0x035a4b20 0x04ec17c0 0x029c42b0 0x0360fe20 0x02ac1c08
  - LBA[10]: 0x021a1eeb 0x01695e50 0x0124da50 0x0059aefc0 0x034cf4c0 0x03c417d0 0x036fe7f0 0x04cbdf50 0x03eb8ef0
  - LBA[20]: 0x02afa8a0 0x01540a40 0x01698040 0x03d1b9e0 0x0209b8c0 0x010dab80 0x032098a0 0x005ddd00 0x037de7c0 0x003f84c0
  - LBA[30]: 0x02adac20 0x00d80a40 0x04c7a9c0 0x03e50f1d0 0x04713908 0x04986f1d0 0x04fc63e8 0x027fd08 0x00261a0 0x01c713e0
  - LBA[40]: 0x01010a0c 0x03f21d40 0x025630c0 0x03f02f130 0x02d3e180 0x04d23b58 0x01bc82d8 0x01eb40e8 0x0492af10 0x04320f48
  - LBA[50]: 0x011caaa30 0x010b0fc0 0x03c87f88 0x049eecc0 0x019c40b0 0x01b10040 0x00e42cb0 0x03141cd0 0x033d0c30 0x03eb18d0
  - LBA[60]: 0x041580d8 0x00012ce8 0x02f54900 0x00718700 0x00703c80 0x032f9c70 0x03462930 0x01380360 0x004088d40
  - LBA[70]: 0x04d5ec80 0x033bb900 0x01e2a340 0x02837a20 0x01a65960 0x03c2a0c0 0x03374ab0 0x03382260 0x04be0668 0x004923b0
  - LBA[80]: 0x0405fa30 0x03241420 0x0395af70 0x02e1d80 0x000f8140 0x03ff0f80 0x02032280 0x012bf700 0x03ta1f80 0x02bab4c0
  - LBA[90]: 0x0273a000 0x025c6780 0x01c73c40 0x02a74500 0x01316820 0x039760e0 0x049a8800 0x0422c240 0x03eacb00 0x00916c98

- PRE LBA LIST:

  - LBA[0]: 0x00a7fff30 0x02aab60 0x03a27d20 0x031d2880 0x04a24d10 0x04d40d50 0x02c59b0 0x029ca020 0x00343d60 0x02070a50

- DETECT READ ERROR:

  - LBA: 0x2bb240 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]

The terminal also shows other log messages related to disk operations and errors.

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公司试用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

@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：



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。



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

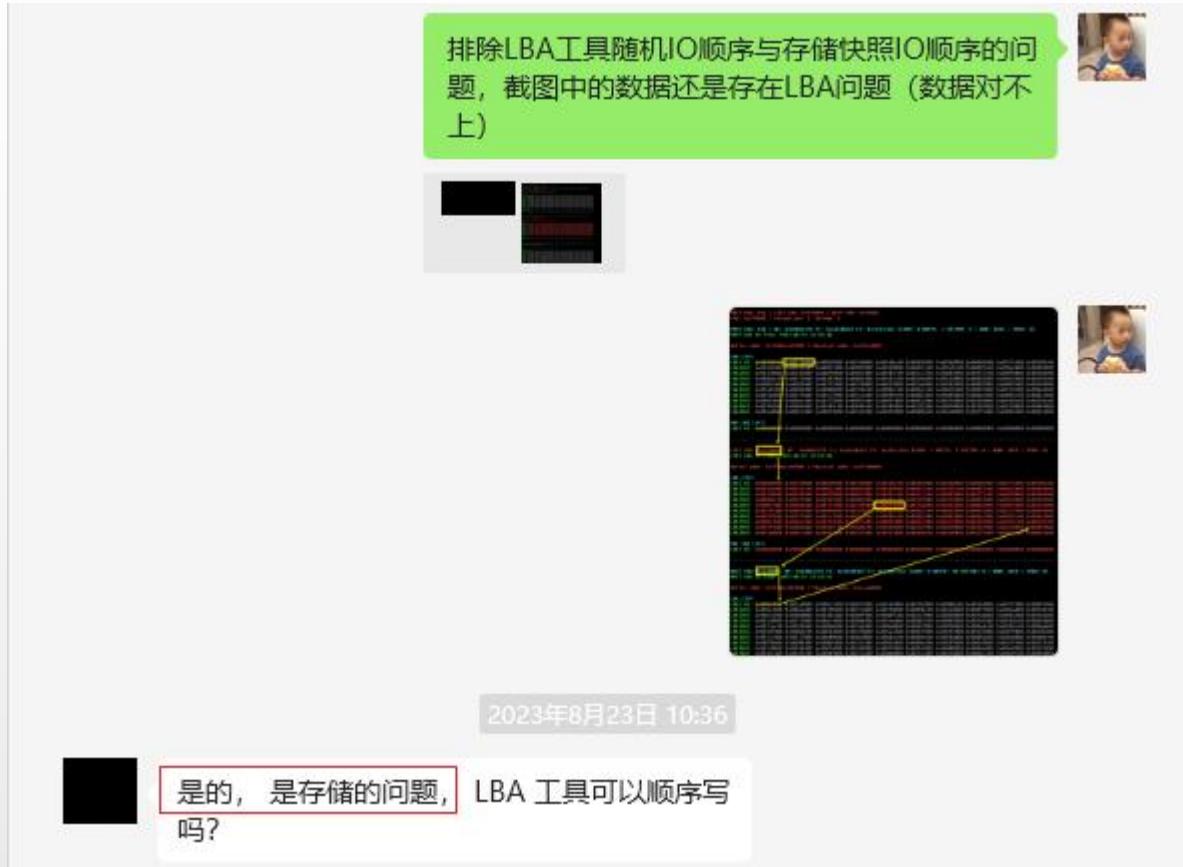


所以LBA工具比fio/vdbench的verify功能强大多了，轻易就测试出来了LBA问题，但fio工具却测试不出来。



# LBA工具介绍：北京XXX公司试用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公司在试用LBA工具中，等待反馈；
- 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 -- 两个最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;

```
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。

```
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 0x00007f5eddcc8bb50 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

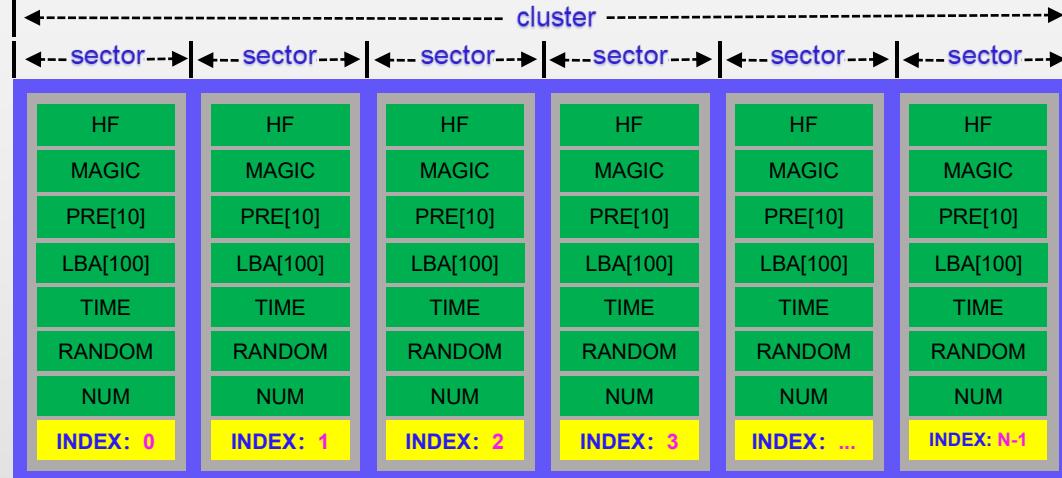


簇

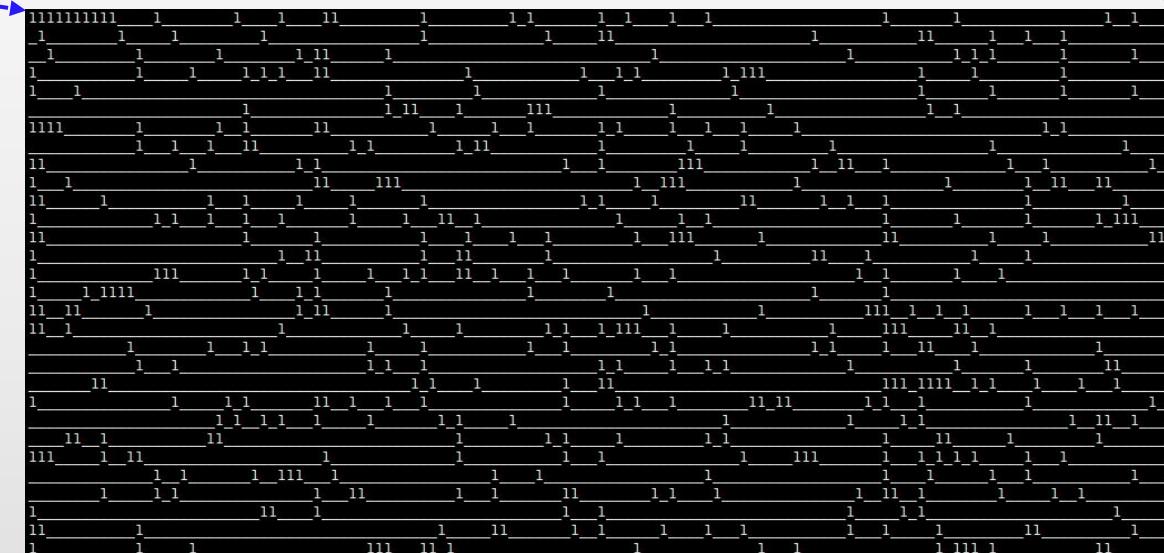
测试/校验固定起始簇



扇区



全局bitmap(内存) bitmapt.txt



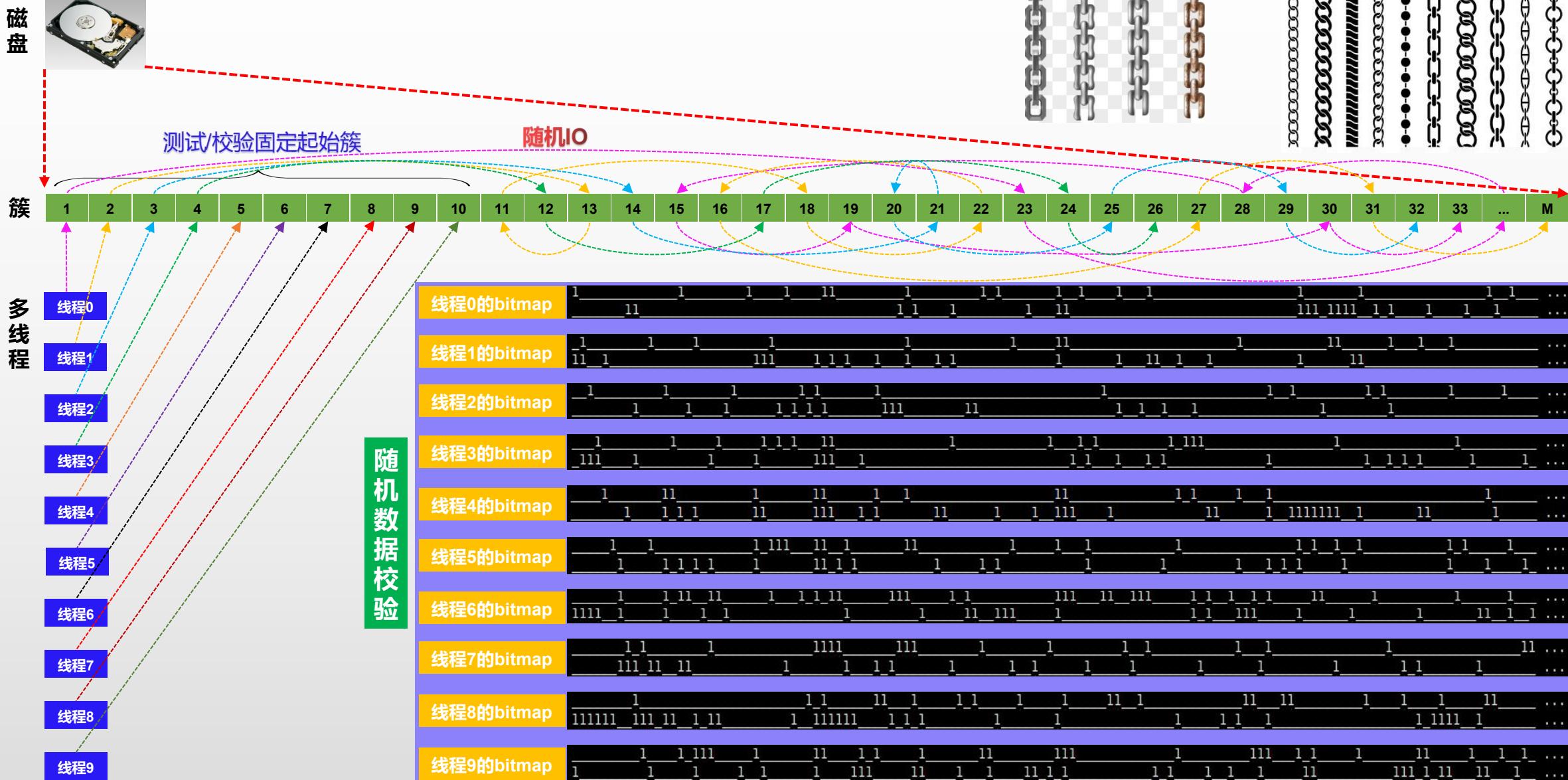
1. 每个磁盘/文件被划分为M个IO数据簇;
2. 每个IO数据簇包含N个扇区;
3. IO数据簇N个扇区中的数据: 每个扇区除index字段表示簇中的扇区序号外, 其它数据均相同

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)



# 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]

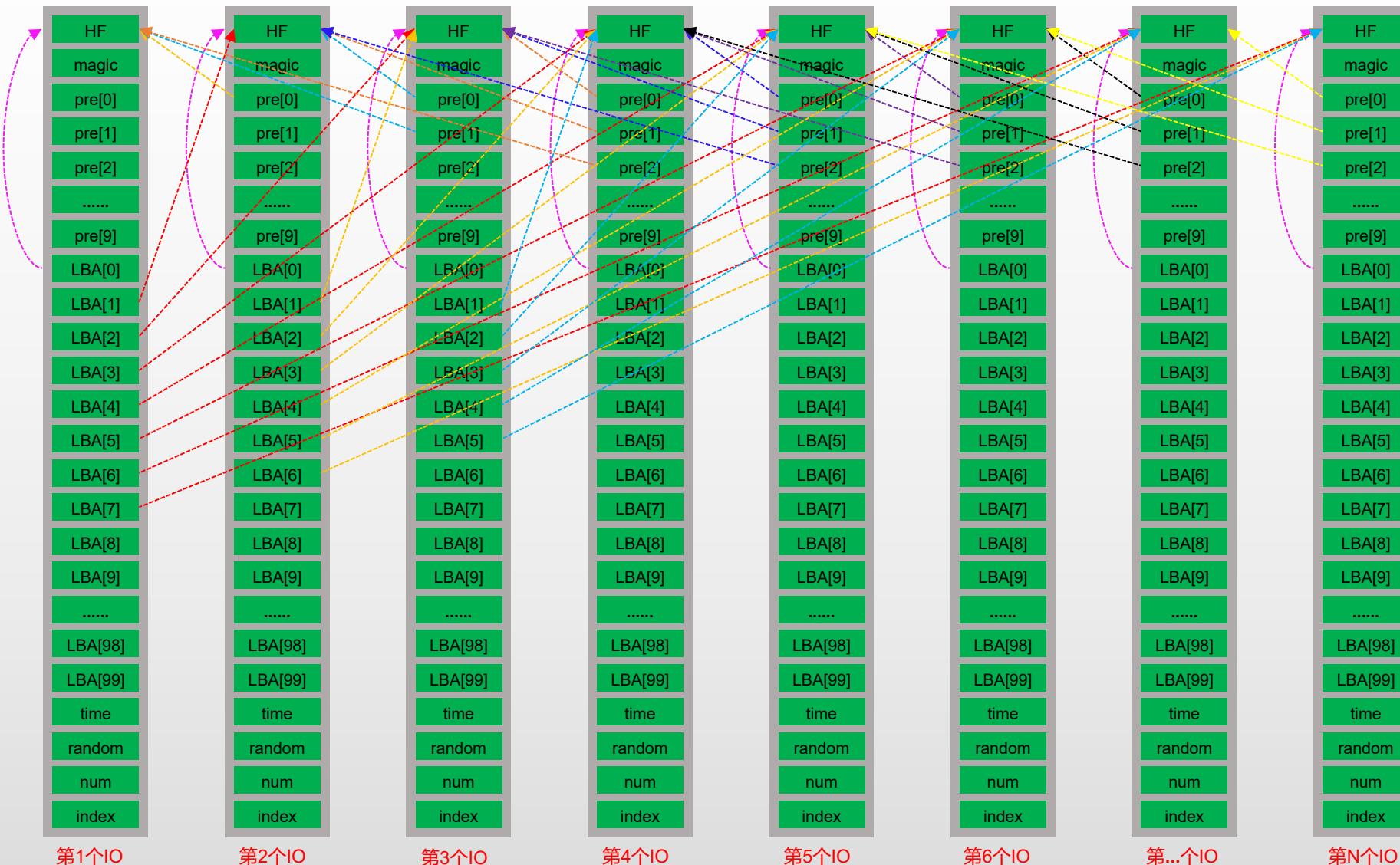
1. 数组1：一个IO数据簇包括N(1024)个扇区，如右图和上图
2. 数组2：包括M+1(702)个扇区的非零数据
3. 数组3：包括N-M-1(322)个扇区的全零数据
4. 数组4：每个扇区中的PRE\_LBA[10]，如上图
5. 数组5：每个扇区中的LBA\_LIST[100]，如上图
6. 队列1：每个扇区中的PRE\_LBA[10]，如下一页
7. 队列2：每个扇区中的LBA\_LIST[100]，如下一页
8. 链表1：PRE\_LBA[10]前向链表，如下一页
9. 链表2：LBA\_LIST[100]后向链表，如下一页



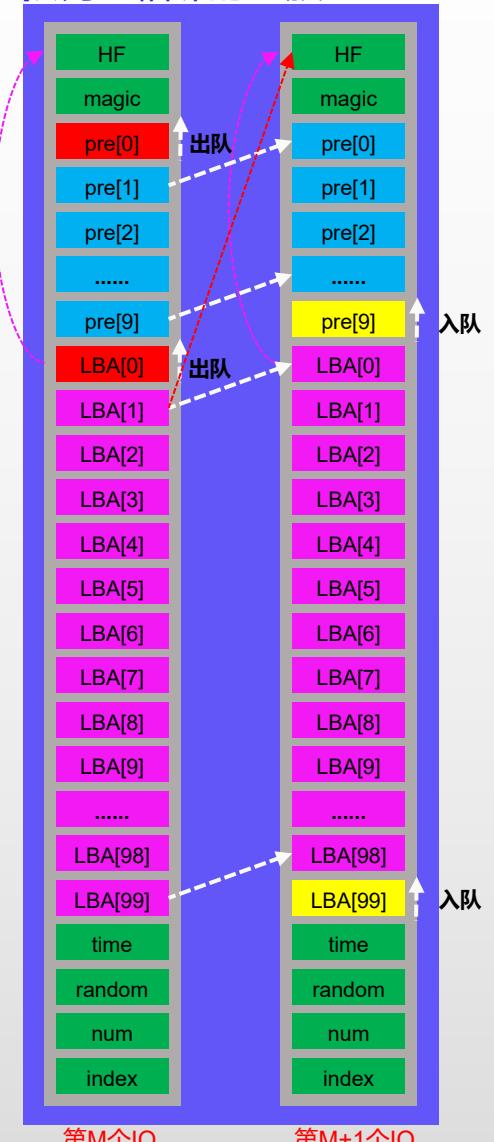
# LBA工具实现原理

## 强双向链表、队列

强双向链表：每个节点，第一个lba指针指向自己，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

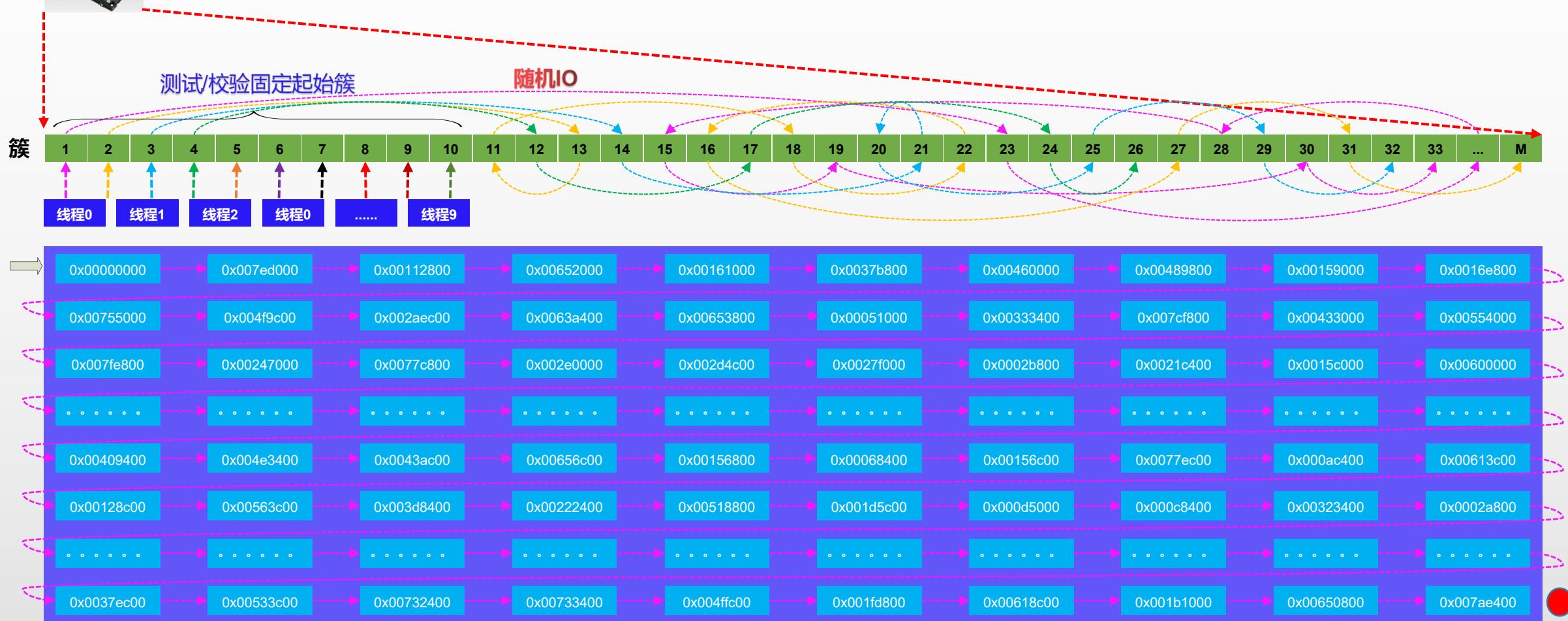




# LBA工具实现原理：数据校验

- 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)

磁盘

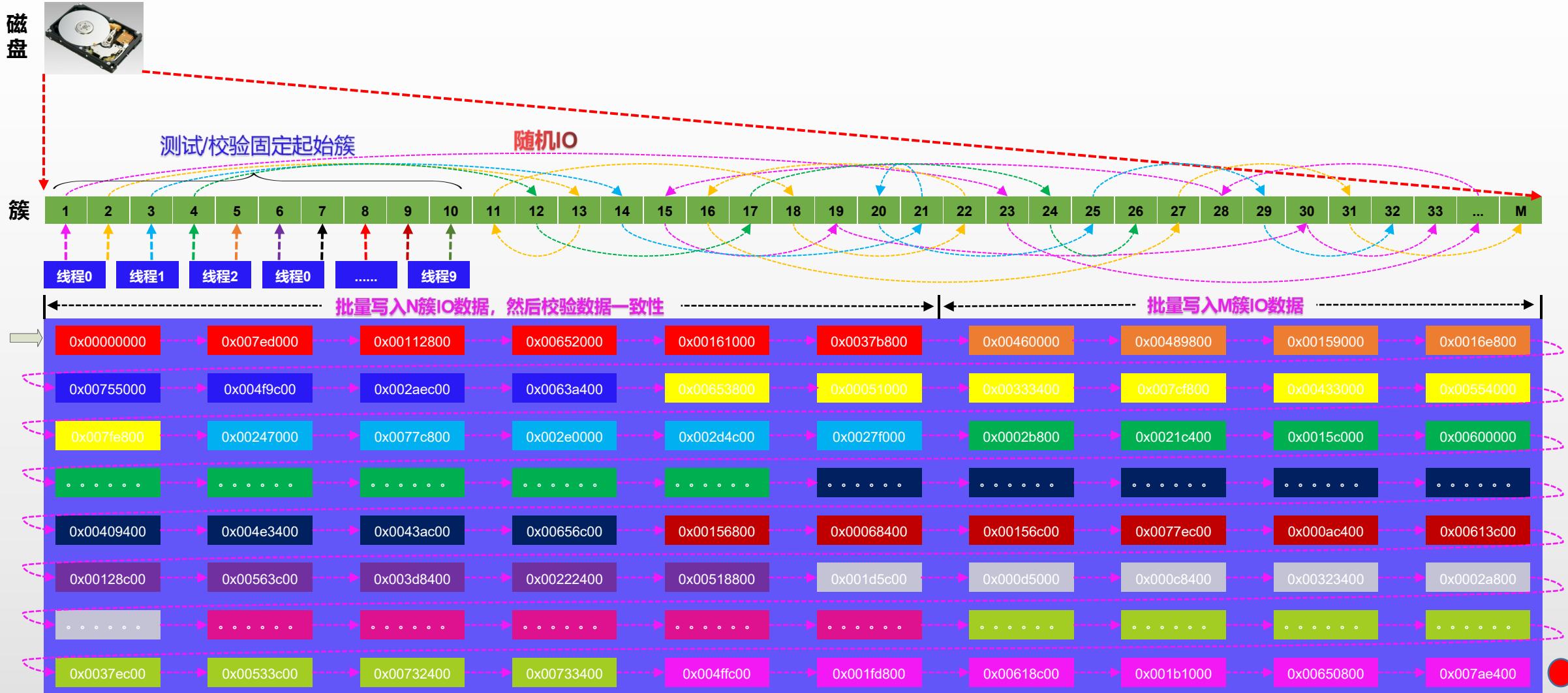


1. runtime的全盘数据校验，即：10个线程向磁盘写入IO数据簇，当磁盘中的数据写满后，10个线程随即对全盘的所有写入数据进行一次完整校验(簇间数据校验、簇内数据校验)；
2. checktime的全盘数据校验，三个重要命令：**hd\_write\_verify -c -D -T 1 disk/file**   **hd\_write\_verify -c -D -T 10 disk/file**   **hd\_write\_verify\_dump -c -D -T 0-9 disk/file**
3. 每次对磁盘写入数据测试完一轮后，可以进行三种数据校验：全盘数据校验、批量数据校验、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)

磁盘

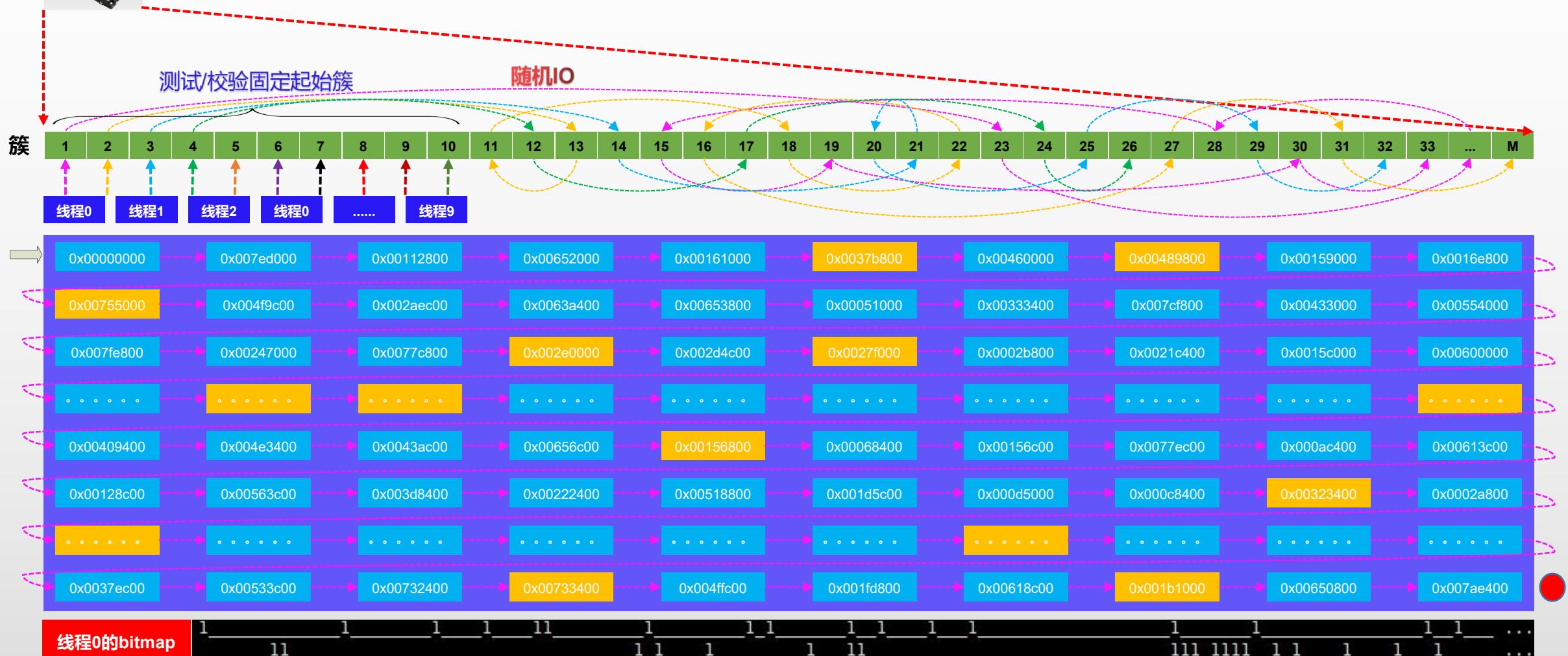


1. 批量数据校验属于**runtime**的数据校验，即：一边写入测试数据，一边校验前面写入数据的一致性；
2. 10个线程，每个线程分批向磁盘写入数据，每批写入磁盘的IO数据簇个数是随机的：**i, j, k, m, n, l, x, y, z, .....**；
3. 在写下一批IO数据簇之前，先校验上一批数据的一致性：簇间数据校验和簇内数据校验，当一个线程的所有数据写完成后，即这个线程的线程数据校验也完成了

# 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)

磁盘

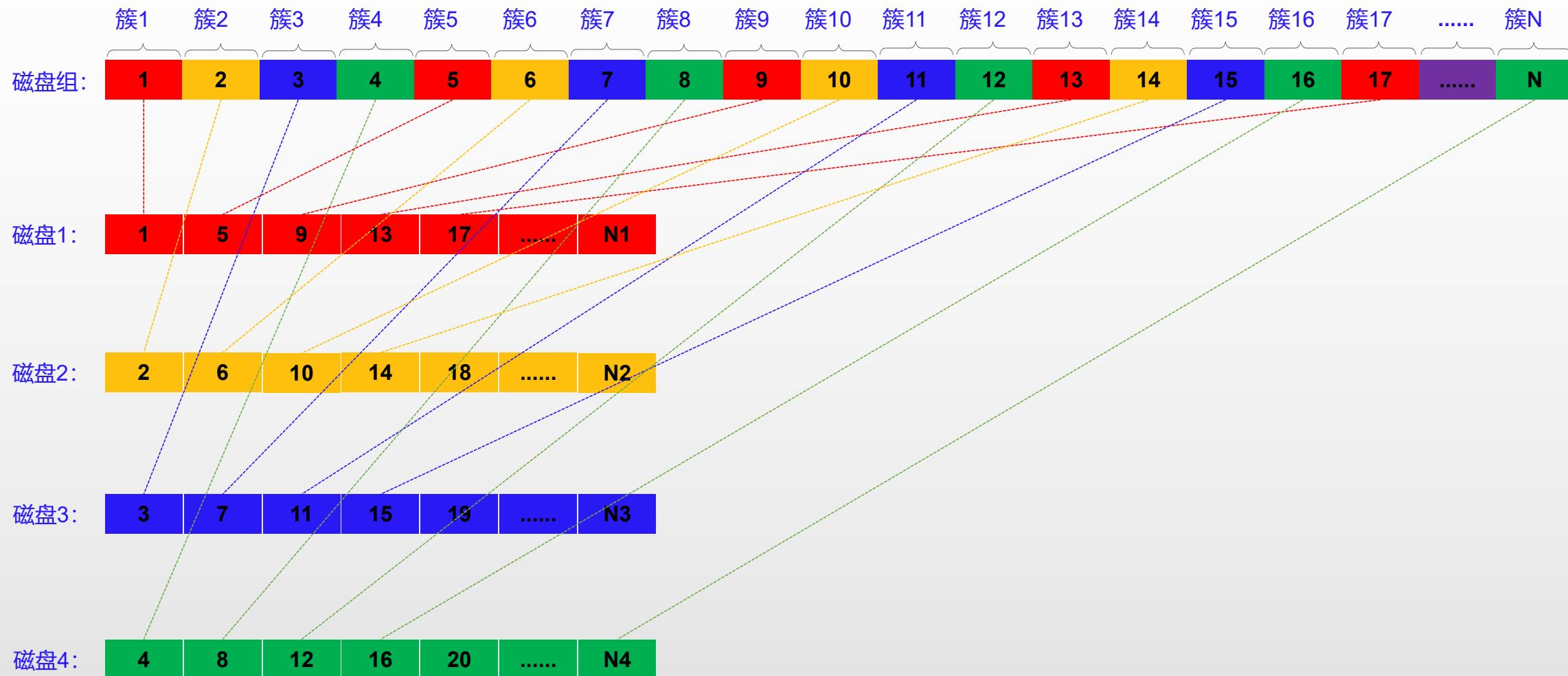


1. 每个线程写入磁盘中的IO数据簇，当每个IO数据簇被分配时，对应的线程bitmap被置位为1；

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 Information:
  File: lba1.raw
  Size: 1073741824
  Blocks: 0
  IO Block: 65536
  Device: 2051
  Inode: 1677992490
  Links: 1
File Size: 1073741824 / 1024M / 1.00G

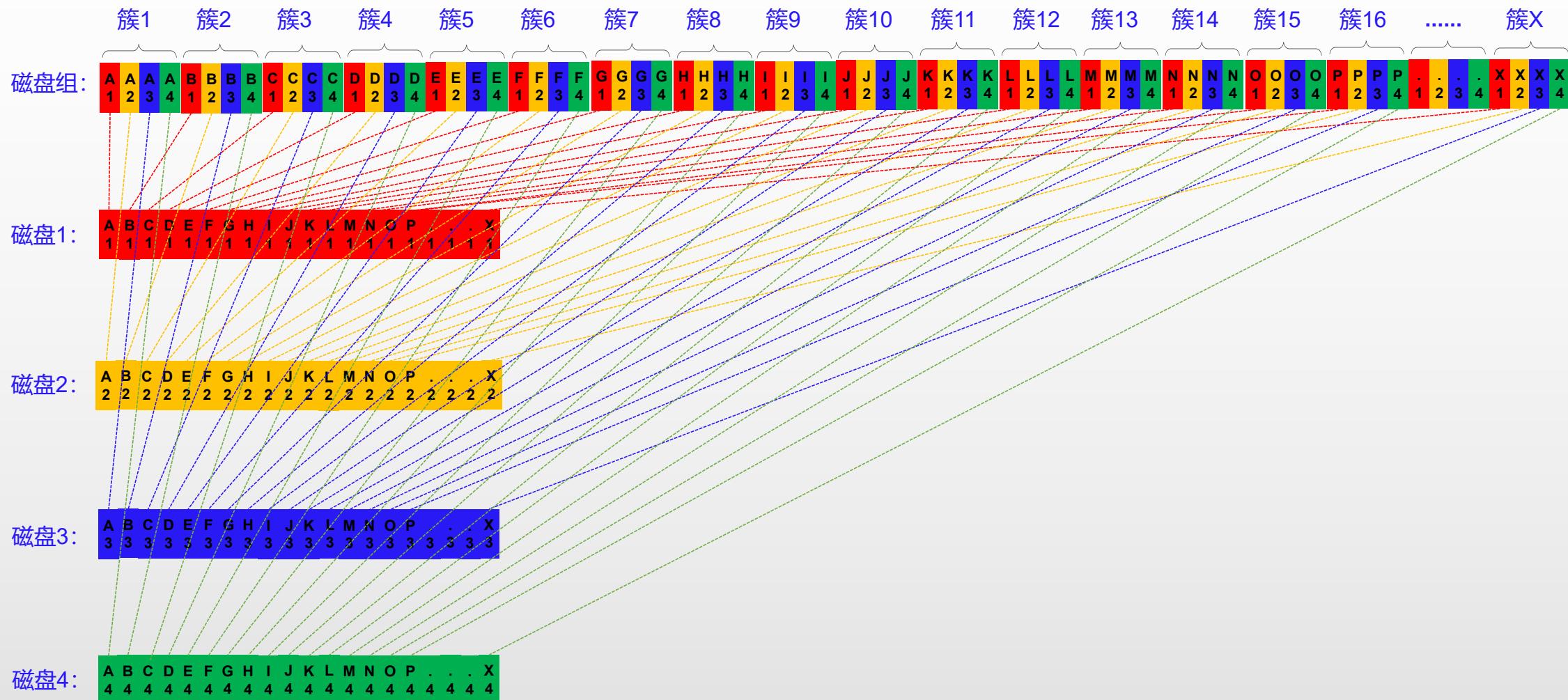
File Information:
  File: lba2.raw
  Size: 1073741824
  Blocks: 0
  IO Block: 65536
  Device: 2051
  Inode: 1677992491
  Links: 1
File Size: 1073741824 / 1024M / 1.00G

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
Sector_Per_Cluster: 1024
Current Time: 2023-06-13 16:27:23
=====
WARNING: [Thread: 2] DIFFER FLAGS | read[213] | filename: lba3.raw
LBA_0: 0x200000 - LBA_1: 0x418800 flags is different
=====
LBA: 0x200000 | HF: 0x04001272 F1: 0x140c2946 F2: 0x969a99c3 | L0OP: 3 WRITE: 212 STRIPE: 0 LBA: 0x80000 SECTOR: 0 / NUM: 1024 / ZERO: 220
IO Time: 2023-06-13 16:24:05
Buffer addr: 0xffff9c950000 | Physical addr: 0x503253d0000
LBA LIST:
LBA[ 0]: 0x00200000 0x00418800 0x0047b800 0x00060c00 0x001b4000 0x00128400 0x007b9000 0x006b2800 0x00712000 0x00440400
LBA[10]: 0x00e6400 0x005ccccc 0x00104400 0x003c9400 0x003a8800 0x00552c00 0x001b6c00 0x007f3c00 0x0041dc00 0x003f9400
LBA[20]: 0x007d9800 0x00420800 0x00477000 0x0073e800 0x007ee000 0x007b0c00 0x005d8c00 0x006ba400 0x0010c000
LBA[30]: 0x0076400 0x002e8400 0x003d4400 0x00470400 0x00698900 0x005c7800 0x00276800 0x00305800 0x00756800 0x007ce000
LBA[40]: 0x001d6c00 0x00774000 0x00683800 0x00052800 0x0047c000 0x00091400 0x005fe800 0x0051ac00 0x002b7c00
LBA[50]: 0x00102400 0x000cf800 0x004b9000 0x006d6000 0x0070c000 0x002f3400 0x002fa800 0x00256000 0x0060e000 0x00269400
LBA[60]: 0x00182c00 0x005d1400 0x001b7400 0x002b9400 0x00153c00 0x0007e000 0x00306000 0x0079800 0x00343800 0x00264000
LBA[70]: 0x00270400 0x006d9400 0x004b1c00 0x007ea000 0x007f7800 0x0063f800 0x0074f000 0x00554800 0x0059400 0x003b8000
LBA[80]: 0x00644000 0x003b6800 0x002a4000 0x0004e400 0x001ec000 0x00317400 0x0019c000 0x0036f400 0x00179800 0x002f9800
LBA[90]: 0x00485000 0x00413000 0x0054ca00 0x002c5000 0x00094400 0x006a3400 0x00led000 0x003f0000 0x006aa400 0x00620c00
PRE LBA LIST:
LBA[ 0]: 0x002ba400 0x004ecc00 0x0079a000 0x007d3800 0x0028e400 0x006alc00 0x006c0400 0x007a0c00 0x004e9c00 0x0001lc00
=====
LBA: 0x418800 | HF: 0x04001278 F1: 0x140c2946 F2: 0xbe924fa4 | L0OP: 2 WRITE: 123 STRIPE: 2 LBA: 0x106000 SECTOR: 0 / NUM: 1024 / ZERO: 330
IO Time: 2023-06-13 16:21:40
Buffer addr: 0xffff9c9f0000 | Physical addr: 0x503ed1c0000
LBA LIST:
LBA[ 0]: 0x00418800 0x005f800 0x005fb00 0x00036800 0x001edc00 0x0007a000 0x00273800 0x000d3c00 0x00479400 0x0076c000
LBA[10]: 0x007c9800 0x00250800 0x001e8000 0x005fc700 0x00537000 0x001d0000 0x00492400 0x002fe400 0x00112000 0x0051400
LBA[20]: 0x00757800 0x0058e000 0x0066d000 0x0009e000 0x0056e000 0x00335400 0x005e4900 0x0071d800 0x001fe000
LBA[30]: 0x005ff400 0x0044f800 0x00098400 0x001e400 0x000fc00 0x002aa00 0x00581c00 0x001ac800 0x00252400 0x000dc000
LBA[40]: 0x004dd400 0x0046e00 0x00221c00 0x00081000 0x00672c00 0x00775400 0x00666000 0x005d3000 0x000c800 0x00118c00
LBA[50]: 0x00024000 0x00524000 0x0061fc00 0x00610800 0x00407800 0x0012c00 0x006fc00 0x00016c00 0x004dc00 0x00562000 0x002ec000
LBA[60]: 0x0076f000 0x0071400 0x0012b00 0x0032400 0x00093400 0x0047b00 0x0075c800 0x00045a00 0x0016d000
LBA[70]: 0x00491400 0x004b600 0x000d900 0x003e4400 0x00559400 0x00670000 0x00183400 0x002f3000 0x0023f000 0x001a400
LBA[80]: 0x001dbc00 0x0058f000 0x002ea800 0x006d8800 0x006cc400 0x00651000 0x001a7800 0x001af000 0x008449400 0x005d6800
LBA[90]: 0x00050500 0x00287800 0x0065f800 0x0052a800 0x00146000 0x006a5cc00 0x0032c400 0x006d5800 0x00503400 0x004fa400
PRE LBA LIST:
LBA[ 0]: 0x00262800 0x0060a400 0x0067a400 0x000ff000 0x006b6400 0x005bd400 0x00365400 0x000eb000 0x0060b400
=====
```

# 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 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

Sector_Per_Cluster: 2048
Current Time: 2023-06-13 17:13:04
=====
WARNING: [Thread: 4] DIFFER FLAGS | read[89] | filename: lba1.raw
LBA_0: 0xb6b6800 - LBA_1: 0x14e000 flags is different

=====
LBA: 0xb6b6800 | HF: 0x08001274 F1: 0xb1406557 F2: 0x044d278d |LOOP: 5 WRITE: 88 STRIPE: 0 LBA: 0x1ada00 SECTOR: 0 / NUM: 2048 / ZERO: 313]
IO Time: 2023-06-13 17:04:26
Buffer addr: 0xfffffa8320000 | Physical addr: 0x100c1750000
PRE LBA LIST:
LBA[ 0]: 0x006ad000 0x002c9000 0x00434000 0x00202800 0x00689000 0x000de000 0x005c9000 0x00220000 0x0015a800 0x00349800

=====
LBA: 0x14e000 | HF: 0x08001270 F1: 0xb1406557 F2: 0x857d67d0 |LOOP: 3 WRITE: 104 STRIPE: 0 LBA: 0x53800 SECTOR: 0 / NUM: 2048 / ZERO: 194]
IO Time: 2023-06-13 17:00:04
Buffer addr: 0xfffffa840000 | Physical addr: 0x1021b81000
PRE LBA LIST:
LBA[ 0]: 0x0014e000 0x00700000 0x0048c800 0x006d2800 0x003c1800 0x00162800 0x000b1000 0x000f3000 0x0031f000 0x0010e000
LBA[10]: 0x0076d000 0x00428800 0x00230800 0x00427000 0x00246800 0x00364800 0x00641800 0x00145000 0x007c8800 0x007a6000
LBA[20]: 0x006e5000 0x00570000 0x00230900 0x00230900 0x006e8000 0x003e0000 0x0032a000 0x0036e000 0x00737800 0x00015800
LBA[30]: 0x005ae000 0x00680000 0x00654800 0x00529800 0x0045e4000 0x0045e4000 0x002ca000 0x004b5800 0x00a8800 0x00738800
LBA[40]: 0x0047c000 0x00181b00 0x0006d000 0x0051000 0x004d8800 0x00725800 0x0050e000 0x001c7800
LBA[50]: 0x004c7000 0x00100000 0x003e9000 0x006d7000 0x005c5800 0x0057c800 0x0075f800 0x00402800 0x00593000 0x002be800
LBA[60]: 0x0066e000 0x0077c800 0x00056000 0x0041800 0x006d4800 0x00369800 0x00631800 0x00636000 0x00369800 0x00538000
LBA[70]: 0x0068e000 0x007fb000 0x00631800 0x00630000 0x00530000 0x0069d000 0x0032a000 0x007b6000 0x007f1800 0x0007b800
LBA[80]: 0x0044e800 0x006c5800 0x00434800 0x0052000 0x00235800 0x00101000 0x0063800 0x0050c000 0x001ba000
LBA[90]: 0x00380000 0x0010c800 0x00031800 0x002fd800 0x0061000 0x00647000 0x00201000 0x002fb800 0x001dc000 0x001e1000
PRE LBA LIST:
LBA[ 0]: 0x0028a000 0x00461000 0x006f3000 0x0006c800 0x00244000 0x0024d000 0x0014d000 0x000504800 0x004fe000

=====
Current Time: 2023-06-13 17:13:09
[Root@localhost lba]#
```

# 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: 0x1ab5ec90 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/vda
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

```
hd_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
```

```
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:         if (num_reqs == 0) {
443:             sector_num = req->sector_num;
444:             nb_sectors = niov = 0;
445:             start = i;
446:         }
447:
448:         nb_sectors += req->qiov.size / BDRV_SECTOR_SIZE;
449:         niov += req->qiov.niov;
450:         num_reqs++;
451:     }
452:
453:     submit_requests(blk, mrb, start, num_reqs, niov);
454:     mrb->num_reqs = 0;
455: }
```

# LBA工具实现原理

- 支持4K磁盘(簇IO随机拆分功能仍然生效)

```
Welcome to the YOUPPLUS's LBA TESTING SYSTEM
https://github.com/zhangyoujia/
YOUPPLUS login: root (automatic login)
Last login: Wed Aug 30 00:26:50 GMT-8 2023 on ttys0
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: 4096
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: 4096
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: 2048
Current Time: 2023-08-30 23:59:01
Thread 5 [tid: 1036]: Starting check disk ...
Thread 7 [tid: 1038]: Starting check disk ...
Thread 8 [tid: 1039]: Starting check disk ...
Thread 6 [tid: 1037]: Starting check disk ...
Thread 3 [tid: 1034]: Starting check disk ...
Thread 2 [tid: 1033]: Starting check disk ...
Thread 9 [tid: 1040]: Starting check disk ...
Thread 0 [tid: 1031]: Starting check disk ...
Thread 4 [tid: 1035]: Starting check disk ...
Thread 1 [tid: 1032]: 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: 2023-08-30 23:59:01
261, 274 | 265, 276 | 257, 279 | 246, 289 | 253, 283 | 271, 281 | 280, 286

Welcome to the YOUPPLUS's LBA TESTING SYSTEM
https://github.com/zhangyoujia/
YOUPPLUS login: root (automatic login)
Last login: Wed Aug 30 15:12:29 GMT-8 2023 on ttys0
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: 4096
Physical block size: 4096
Minimum I/O size: 4096
Optimal I/O size: 0 (0: unknown)
Alignment offset: 0
Disk Size: 10737418240 / 10240M / 10.00G

Device Topology:
Disk: /dev/vdb
Logical block size: 4096
Physical block size: 4096
Minimum I/O size: 4096
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: 2023-08-30 18:27:25
Thread 2 [tid: 1026]: Starting check disk ...
Thread 4 [tid: 1028]: Starting check disk ...
Thread 8 [tid: 1032]: Starting check disk ...
Thread 5 [tid: 1029]: Starting check disk ...
Thread 0 [tid: 1024]: Starting check disk ...
Thread 7 [tid: 1031]: Starting check disk ...
Thread 1 [tid: 1025]: Starting check disk ...
Thread 9 [tid: 1033]: Starting check disk ...
Thread 3 [tid: 1027]: Starting check disk ...
Thread 6 [tid: 1030]: 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: 2023-08-30 18:27:26
448, 401 | 488, 426 | 455, 405 | 440, 409 | 447, 402 | 465, 411 | 431, 418 | 472, 415 | 433, 417 | 473, 417 | _
```

# 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

```
1200: static int save_zero_page_to_file(RAMState *rs, QEMUFfile *file,
1201:                                     RAMBlock *block, ram_addr_t offset)
1202: {
1203:     uint8_t *p = block->host + offset;
1204:     int len = 0;
1205:
1206:     if (buffer_is_zero(p, TARGET_PAGE_SIZE)) {
1207:         len += save_page_header(rs, file, block, offset | RAM_SAVE_FLAG_ZERO);
1208:         qemu_put_byte(file, 0);
1209:         len += 1;
1210:         ram_release_page(block->idstr, offset);
1211:     }
1212:     return len;
1213: }
131: static void blk_send(QEMUFfile *f, BlkMigBlock *blk)
132: {
133:     int len;
134:     uint64_t flags = BLK_MIG_FLAG_DEVICE_BLOCK;
135:
136:     if (block_mig_state.zero_blocks &&
137:         buffer_is_zero(blk->buf, BLK_MIG_BLOCK_SIZE)) {
138:         flags |= BLK_MIG_FLAG_ZERO_BLOCK;
139:     }
140: }
```

```
420: static bool
421: extent_copy (int src_fd, int dest_fd, char
422:                 size_t hole_size, off_t src_
423:                 enum Sparse_type sparse_mode
424:                 char const *src_name, char c
425:                 bool *require_normal_copy)
426: {
427:     struct extent_scan scan;
428:     off_t last_ext_start = 0;
429:     off_t last_ext_len = 0;
430:
431:     /* Keep track of the output position.
432:      We may need this at the end, for a final
433:      off_t dest_pos = 0;
434:
435:     extent_scan_init (src_fd, &scan);
436:
437:     *require_normal_copy = false;
438:     bool wrote_hole_at_eof = true;
439:
440:     do
441:     {
442:         if (! extent_scan_read (&scan));
443:         if (! ok)
444:             break;
445:     } while (last_ext_start + last_ext_len < hole_size);
446: }
```

```
l2 table[8186], data offset: 0x6c650000 [0x800000006c650000: COPIED] | vdisk offset: 0xdfffa000
l2 table[8187], data offset: 0x6c660000 [0x800000006c660000: COPIED] | vdisk offset: 0xdfffb000
l2 table[8188], data offset: 0x6c700000 [0x800000006c670000: ZERO] | vdisk offset: 0xdffcc000
l2 table[8189], data offset: 0x6c680000 [0x800000006c680000: ZERO] | vdisk offset: 0xdffdd000
l2 table[8190], data offset: 0x6c690000 [0x800000006c690000: ZERO] | vdisk offset: 0xdffee000
l2 table[8191], data offset: 0x6c6a0000 [0x800000006c6a0000: ZERO] | vdisk offset: 0xdffff000
l2 Entry: unaligned: 0, invalid: 0, used: 4132, zero alloc: 21, zero_plain: 76 unused: 3963
-----
l1 table[ 7], l2 offset: 0xdf0000 [0x8000000000df0000: COPIED]
l2 table[ 0], data offset: 0xf4b0000 [0x8000000000f4b000: COPIED] | vdisk offset: 0xe0000000
l2 table[ 1], data offset: 0xf4c0000 [0x8000000000f4c000: COPIED] | vdisk offset: 0xe0010000
l2 table[ 2], data offset: 0xf4d0000 [0x8000000000f4d000: COPIED] | vdisk offset: 0xe0020000
l2 table[ 3], data offset: 0xf4e0000 [0x8000000000f4e000: COPIED] | vdisk offset: 0xe0030000
l2 table[ 4], data offset: 0xf4f0000 [0x8000000000f4f000: COPIED] | vdisk offset: 0xe0040000
l2 table[ 5], data offset: 0xf500000 [0x8000000000f50000: COPIED] | vdisk offset: 0xe0050000
l2 table[ 6], data offset: 0xf510000 [0x8000000000f51000: COPIED] | vdisk offset: 0xe0060000
l2 table[ 7], data offset: 0xf520000 [0x8000000000f52000: COPIED] | vdisk offset: 0xe0070000
l2 table[ 8], data offset: 0xf530000 [0x8000000000f53000: COPIED] | vdisk offset: 0xe0080000
l2 table[ 9], data offset: 0xf540000 [0x8000000000f54000: COPIED] | vdisk offset: 0xe0090000
l2 table[10], data offset: 0xf550000 [0x8000000000f55000: COPIED] | vdisk offset: 0xe00a0000
l2 table[11], data offset: 0xf560000 [0x8000000000f56000: COPIED] | vdisk offset: 0xe00b0000
l2 table[12], data offset: 0xf570000 [0x8000000000f57000: COPIED] | vdisk offset: 0xe00c0000
l2 table[13], data offset: 0xf580000 [0x8000000000f58000: 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：

```
68: #define WRITE_SAME          0x41
69: #define UNMAP                0x42
70: #define NVME_CMD_FLUSH        0x00
71: #define NVME_CMD_WRITE         0x01
72: #define NVME_CMD_READ          0x02
73: #define NVME_CMD_WRITE_UNCOR   0x04
74: #define NVME_CMD_COMPARE       0x05
75: #define NVME_CMD_WRITE_ZEROES  0x08
76: #define NVME_CMD_DSM           0x09
77: #define NVME_CMD_VERIFY        0x0c
78: #define NVME_CMD_RESV_REGISTER 0x0d
79: #define NVME_CMD_RESV_REPORT   0x0e
80: #define NVME_CMD_RESV_ACQUIRE  0x11
81: #define NVME_CMD_RESV_RELEASE  0x15
82: #define NVME_CMD_ZONE_MGMT_SEND 0x79
83: #define NVME_CMD_ZONE_MGMT_RECV 0x7a
84: #define NVME_CMD_ZONE_APPEND   0x7d
85:
86: enum nvme_opcode {
87:     nvme_cmd_flush        = 0x00,
88:     nvme_cmd_write         = 0x01,
89:     nvme_cmd_read          = 0x02,
90:     nvme_cmd_write_uncor   = 0x04,
91:     nvme_cmd_compare       = 0x05,
92:     nvme_cmd_write_zeroes  = 0x08,
93:     nvme_cmd_dsm           = 0x09,
94:     nvme_cmd_verify        = 0x0c,
95:     nvme_cmd_resv_register = 0x0d,
96:     nvme_cmd_resv_report   = 0x0e,
97:     nvme_cmd_resv_acquire  = 0x11,
98:     nvme_cmd_resv_release  = 0x15,
99:     nvme_cmd_zone_mgmt_send= 0x79,
100:    nvme_cmd_zone_mgmt_recv = 0x7a,
101:    nvme_cmd_zone_append   = 0x7d,
102: };
103:
104: static int blkdev_common_ioctl(struct block_device *bdev, fmode_t mode,
105:                                unsigned cmd, unsigned long arg, void __user *argp)
106: {
107:     unsigned int max_sectors;
108:
109:     switch (cmd) {
110:     case BLKFLSBUF:
111:         return blkdev_flushbuf(bdev, mode, cmd, arg);
112:     case BLKROSET:
113:         return blkdev_roset(bdev, mode, cmd, arg);
114:     case BLKDISCARD:
115:         return blk_ioctl_discard(bdev, mode, arg, 0);
116:     case BLKSECDISCARD:
117:         return blk_ioctl_discard(bdev, mode, arg,
118:                               BLKDEV_DISCARD_SECURE);
119:     case BLKZEROOUT:
120:         return blk_ioctl_zeroout(bdev, mode, arg);
121:     default:
122:         return -ENOTTY;
123:     }
124: }
```

(1) mkfs.ext4 /dev/sdb # cat /etc/fstab  
(2) mount -o discard /dev/sdb /mnt/test/ UUID=3453g54-6628-2346-8123435f /home xfs defaults,discard 0 0  
(3) fstrim -a

```
440: u16 nvmet_bdev_parse_io_cmd(struct nvmet_req *req)
441: {
442:     struct nvme_command *cmd = req->cmd;
443:
444:     switch (cmd->common.opcode) {
445:     case nvme_cmd_read:
446:     case nvme_cmd_write:
447:         req->execute = nvmet_bdev_execute_rw;
448:         if (req->sq->ctrl->pi_support && nvmet_ns_has_pi(req->ns))
449:             req->metadata_len = nvmet_rw_metadata_len(req);
450:         return 0;
451:     case nvme_cmd_flush:
452:         req->execute = nvmet_bdev_execute_flush;
453:         return 0;
454:     case nvme_cmd_dsm:
455:         req->execute = nvmet_bdev_execute_dsm;
456:         return 0;
457:     case nvme_cmd_write_zeroes:
458:         req->execute = nvmet_bdev_execute_write_zeroes;
459:         return 0;
460:     default:
461:         return nvmet_report_invalid_opcode(req);
462:     }
463: }
```

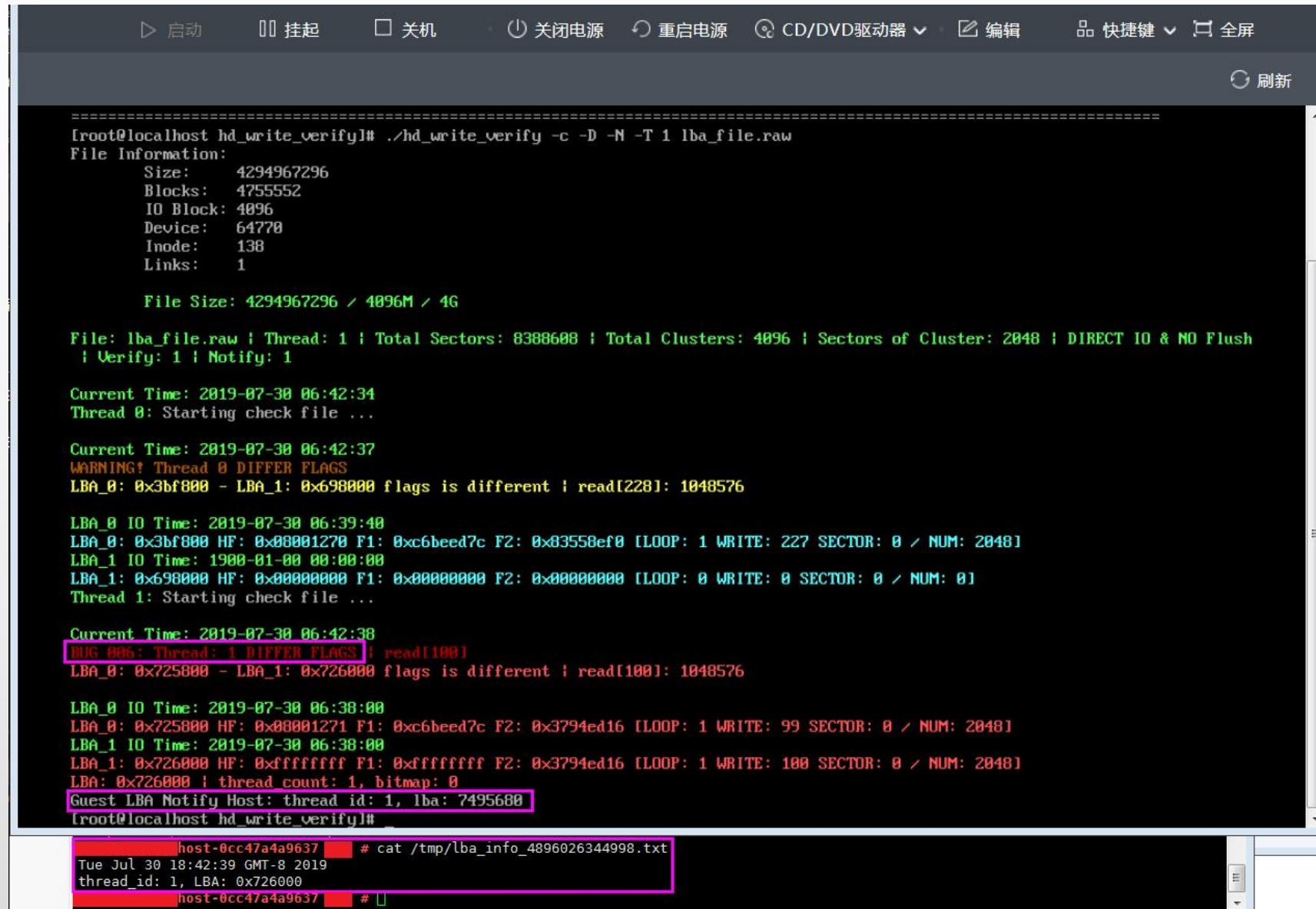
## LBA工具实现原理

➤ 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？

# 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)



The screenshot shows a terminal window with a dark theme. At the top, there is a menu bar with options: 启动 (Launch), 挂起 (Suspend), 关机 (Shutdown), 关闭电源 (Power Off), 重启电源 (Reboot), CD/DVD驱动器 (CD/DVD Drive), 编辑 (Edit), 快捷键 (Shortcuts), 全屏 (Full Screen), and 刷新 (Refresh). Below the menu, the terminal output is displayed:

```
froot@localhost hd_write_verify]# ./hd_write_verify -c -D -N -T 1 lba_file.raw
=====
File Information:
  Size: 4294967296
  Blocks: 4755552
  IO Block: 4096
  Device: 64770
  Inode: 138
  Links: 1

  File Size: 4294967296 / 4896M / 4G

File: lba_file.raw : Thread: 1 : Total Sectors: 8388608 : Total Clusters: 4096 : Sectors of Cluster: 2048 : DIRECT IO & NO Flush
: Verify: 1 : Notify: 1

Current Time: 2019-07-30 06:42:34
Thread 0: Starting check file ...

Current Time: 2019-07-30 06:42:37
WARNING! Thread 0 DIFFER FLAGS
LBA_0: 0x3bf800 - LBA_1: 0x698000 flags is different : read[228]: 1048576

LBA_0 IO Time: 2019-07-30 06:39:40
LBA_0: 0x3bf800 HF: 0x08001270 F1: 0xc6beed7c F2: 0x83558ef0 [LOOP: 1 WRITE: 227 SECTOR: 0 / NUM: 2048]
LBA_1 IO Time: 1900-01-00 00:00:00
LBA_1: 0x698000 HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0]
Thread 1: Starting check file ...

Current Time: 2019-07-30 06:42:38
BUG #06: Thread: 1 DIFFER FLAGS! read[100]
LBA_0: 0x725800 - LBA_1: 0x726000 flags is different : read[100]: 1048576

LBA_0 IO Time: 2019-07-30 06:38:00
LBA_0: 0x725800 HF: 0x08001271 F1: 0xc6beed7c F2: 0x3794ed16 [LOOP: 1 WRITE: 99 SECTOR: 0 / NUM: 2048]
LBA_1 IO Time: 2019-07-30 06:38:00
LBA_1: 0x726000 HF: 0xffffffff F1: 0xffffffff F2: 0x3794ed16 [LOOP: 1 WRITE: 100 SECTOR: 0 / NUM: 2048]
LBA: 0x726000 | thread count: 1, bitmap: 0
Guest LBA Notify Host: thread id: 1, lba: 7495680
froot@localhost hd_write_verify]#
```

At the bottom of the terminal, there is a red box highlighting the command and its output:

```
host-0cc47a4a9637 # cat /tmp/lba_info_4896026344998.txt
Tue Jul 30 18:42:39 GMT-8 2019
thread_id: 1, LBA: 0x726000
host-0cc47a4a9637 #
```

# 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]#
```

```
-----
LBA: 0x5f1000 [HF: 0xffffffff F1: 0aaaaaaaaa F2: 0xf043e3a4 ]LOOP: 1 WRITE: 5 SE
CTOR: 0 / NUM: 2048]
IO Time: 2020-03-03 01:25:21

Buffer addr: 0xfffff7ef2000 | Physical addr: 0x3cd70000

Breakpoint 1, hd_verify_single_cluster (verify=0xfffff7ef2000,
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	3CD6FEB80	C0 4E D7 00 00 00 00 E0 4E D7 00 00 00	cal addr: 0x582dae40000
200.201.24.154	3CD6FEA0	40 00 00 00 00 00 00 E0 4E D7 00 00 00	cal addr: 0x582bac30000
200.201.33.40	3CD6FEC0	72 73 70 00 00 00 00 00 00 00 00 00	cal addr: 0x58286850000
	3CD6FEE0	64 61 74 61 5F 70 74 72 00 00 00 00	cal addr: 0x582b8ea0000
	3CD6FF00	40 4F D7 00 00 00 00 08 00 00 00 00	cal addr: 0x582be010000
	3CD6FF20	40 00 00 00 00 00 00 00 00 4F D7 00 00	cal addr: 0x582a4030000
	3CD6FF40	72 38 00 00 00 00 00 00 00 00 00 00	read: 0xfffff9744f120
	3CD6FF60	69 6E 74 36 34 00 00 00 00 00 00 00	cal addr: 0x582ab20000
	3CD6FF80	C0 4F D7 00 00 00 00 11 00 00 00 00	cal addr: 0x582b4730000
	3CD6FFA0	20 00 00 00 00 00 00 E0 4F D7 00 00	cal addr: 0x582b7e00000
	3CD6FFC0	65 66 6C 61 67 73 00 00 88 9E E9 D8	cal addr: 0x582bd830000
	3CD6FFE0	5F 65 66 6C 61 67 73 00 00 7F 7F 00	cal addr: 0x582bd830000
	3CD70000	FF FF FF FF AA AA AA AA 00 00 00 00	cal addr: 0x582bd830000
	---	3733563982736.qemu_back_mem_objects_mem-node0.sdcMy0	cal addr: 0x582b4740000

```
[root@localhost hd_write_verify]# cat mem_map.1579591_00_1579592
verify_bitmap: ID: 0, Tid: 1579592, Thread: 0xfffff9744f120
Buffer addr: 0xfffff90010000 | Physical addr: 0x58349b30000
```

```
no_memalign: ID: 0, Tid: 1579592, Thread: 0xfffff9744f120
Buffer addr: 0xfffff949b0000 | Physical addr: 0x58321850000
Buffer addr: 0xfffff949c0000 | Physical addr: 0x582b0dc0000
Buffer addr: 0xfffff949d0000 | Physical addr: 0x583e88a0000
Buffer addr: 0xfffff949e0000 | Physical addr: 0x58329ad0000
Buffer addr: 0xfffff949f0000 | Physical addr: 0x583caf40000
Buffer addr: 0xfffff949a0000 | Physical addr: 0x583f5500000
Buffer addr: 0xfffff94a10000 | Physical addr: 0x583cca80000
Buffer addr: 0xfffff94a20000 | Physical addr: 0x58318a60000
Buffer addr: 0xfffff94a30000 | Physical addr: 0x58310000000
Buffer addr: 0xfffff94a40000 | Physical addr: 0x5830fe70000
Buffer addr: 0xfffff94a50000 | Physical addr: 0x58394730000
Buffer addr: 0xfffff94a60000 | Physical addr: 0x582c5020000
Buffer addr: 0xfffff94a70000 | Physical addr: 0x582bf910000
Buffer addr: 0xfffff94a80000 | Physical addr: 0x583cd3c0000
Buffer addr: 0xfffff94a90000 | Physical addr: 0x582a9840000
Buffer addr: 0xfffff94aa0000 | Physical addr: 0x582b82a0000
Buffer addr: 0xfffff94ab0000 | Physical addr: 0x58285240000
```

```
test_head: ID: 0, Tid: 1579592, Thread: 0xfffff9744f120
Buffer addr: 0xfffff7d540000 | Physical addr: 0x582bca60000
Buffer addr: 0xfffff7d550000 | Physical addr: 0x582a1fb0000
Buffer addr: 0xfffff7d560000 | Physical addr: 0x582b8ea0000
Buffer addr: 0xfffff7d570000 | Physical addr: 0x582be010000
Buffer addr: 0xfffff7d580000 | Physical addr: 0x582a1fe0000
Buffer addr: 0xfffff7d590000 | Physical addr: 0x582a2200000
Buffer addr: 0xfffff7d5a0000 | Physical addr: 0x582bf3a0000
Buffer addr: 0xfffff7d5b0000 | Physical addr: 0x582a35b0000
Buffer addr: 0xfffff7d5c0000 | Physical addr: 0x582a3590000
Buffer addr: 0xfffff7d5d0000 | Physical addr: 0x582adbe0000
Buffer addr: 0xfffff7d5e0000 | Physical addr: 0x582ae070000
Buffer addr: 0xfffff7d5f0000 | Physical addr: 0x582adb00000
```

# 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@localhost lba]# 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      0-9 [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]

[EXAMPLES]:
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

[root@localhost lba]#
[root@localhost lba]# hd_write_verify_dump -v
hd_write_verify_dump version 23.07, Copyright (c) 2016-2023 YOUPPLUS
[root@localhost lba]#
```

## 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

# 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( <b>突出重点信息</b> )
-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 ( <b>高级功能</b> )
-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 0-10 [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: <b>循环运行中校验</b> : 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) <b>校验完即退出</b> : check(只读全盘数据校验) = loop, once(只读校验一次) = all
-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 ( <b>高级功能</b> )
其它参数	<b>高级功能, 略</b>

# 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 ( <b>高级功能</b> )
-Z   --zero-sectors 0-1024[default: random]	<b>高级功能</b> , 略

# 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 ...
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 ... 工作线程: 序号及ID

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: 0xe58e1ecd [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:
  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 块大小: 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: 0xaad400 - LBA_1: 0x754800 flags is different
```

```
LBA地址: 扇区号 硬编码flag、全局魔数、每轮测试毫秒 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: 0xaad400 | HF: 0x08001275 F1: 0xbef02254 F2: 0xd6fdfa70 [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: 0xfffffb16d0000 | 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]: 0xaad400	0x00754800	0x0128b800	0x0126b000	0x01272000	0x0087c000	0x0011aa000	0x00e89000	0x006e6000	0x012ad800	
LBA[10]:	0x00232800	0x00577800	0x001cc800	0x000fe800	0x00a7b800	0x0073c000	0x00e4d000	0x00c6e000	0x00ze2000	0x01386b00
LBA[20]:	0x0134c800	0x007c5800	0x0029e800	0x0016c000	0x012e5800	0x00ccf000	0x0002d000	0x003eb800	0x0125c000	0x0139b000
LBA[30]:	0x004e0800	0x00b01800	0x00ce7000	0x0069e000	0x013aa800	0x00fec800	0x00dc2000	0x01241800	0x012af800	0x008e5800
LBA[40]:	0x00b13800	0x00d30000	0x00266800	0x013e6800	0x001c4000	0x007ec800	0x01053d800	0x010e1800	0x00db0800	0x00681800
LBA[50]:	0x00b73800	0x00dc7000	0x01360800	0x01978800	0x0122f000	0x01345800	0x004a1000	0x00746000	0x00018000	0x011a8000
LBA[60]:	0x008eb000	0x00cd0000	0x00a33800	0x00628000	0x00b4d800	0x00a4e000	0x002f8800	0x0085a800	0x008ec000	0x00c05800
LBA[70]:	0x00aa7000	0x007ff6800	0x00772800	0x00458000	0x00554800	0x00893800	0x00109000	0x00563000	0x012b8800	0x013fa000
LBA[80]:	0x000797800	0x00870800	0x005df000	0x00c07800	0x009b800	0x00115c800	0x00ad8000	0x0026d9800	0x0074f800	0x00dc8800
LBA[90]:	0x009e3800	0x00751800	0x0038c800	0x00cb800	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]: 0x000f0e800 0x01392000 0x0136e800 0x00efa800 0x001ae800 0x01179800 0x00c21800 0x0132d800 0x0077e800 0x00fe4000
```

```
LBA地址: 扇区号 硬编码flag、全局魔数、每轮测试毫秒 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: 0xe224c11f [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: 0xfffffb17f0000 | 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]: 0xaad400	0x00754800	0x0099c0000	0x00576800	0x00b3d000	0x00961000	0x00f29800	0x0101f000	0x00cla800	0x00bf4000	
LBA[10]:	0x00998000	0x00bda000	0x00fad800	0x00781800	0x00f9c800	0x00336800	0x00add000	0x00c08000	0x008b0800	0x0098b000
LBA[20]:	0x01388000	0x00981000	0x01321000	0x011d7000	0x01084800	0x00a7a7000	0x011bb800	0x0067f800	0x00f0ff800	
LBA[30]:	0x00731000	0x00ca1000	0x00d0c800	0x00e76000	0x0058f000	0x007f0800	0x00410000	0x0094f000	0x009d1000	0x0032e800
LBA[40]:	0x00fe7000	0x00630000	0x01022800	0x0080c800	0x00778000	0x00181000	0x00650800	0x012a9800	0x013f7000	0x00129000
LBA[50]:	0x0041d000	0x00670000	0x00c82800	0x0011cb800	0x00640000	0x00bd7800	0x002c9000	0x0ed0800	0x00844000	0x00f36800
LBA[60]:	0x005cb800	0x00fe1800	0x00bd9800	0x00d82000	0x00efc3800	0x011d5000	0x00716000	0x00c18000	0x00897000	0x00438000
LBA[70]:	0x00ee0000	0x01125000	0x0099e5800	0x0097d000	0x009a7000	0x00c90800	0x00b29800	0x007cf000	0x0074f800	0x01150800
LBA[80]:	0x00048000	0x00d79000	0x00cb4800	0x011ca000	0x00ae5000	0x00a19000	0x010c5000	0x0078a800	0x00074800	0x00720000
LBA[90]:	0x01128000	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
```

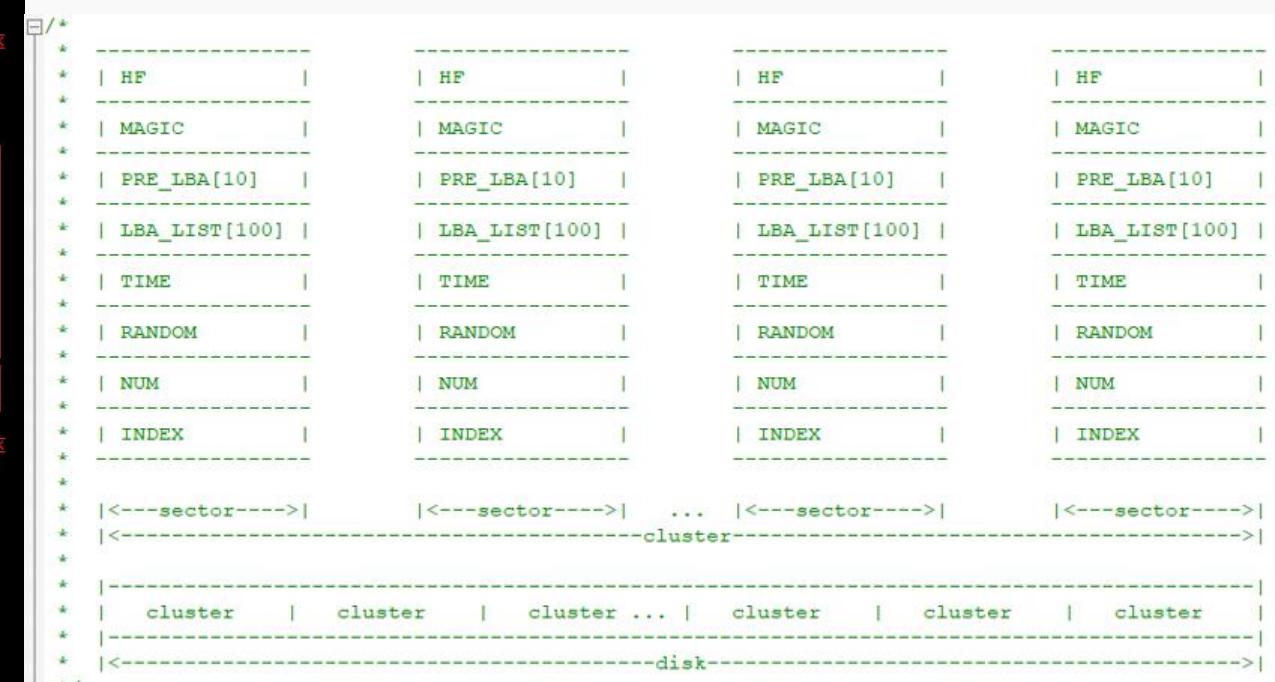
```
LBA[ 0]: 0x00067000 0x0059f800 0x00222800 0x009f6000 0x0117d000 0x00768000 0x00ca8000 0x00dbe000 0x007e6000 0x00fb0000
```

```
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]
=====
```



# 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: 1077952576  
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]: 0x11eb5c0 0x011e8a00 0x00bb6680 0x00e9d700 0x0075bb00 0x010b8e00 0x01199000 0x008eca00 0x00a4c5c0 0x00f2a340

LBA[10]: 0x00095a00 0x0a75bc0 0x008b9880 0x00d80b00 0x012095c0 0x01358340 0x01236480 0x0119a300 0x01185900 0x0b040f80

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 0x003bd80 0x00230c80 0x0042e8c0 0x0021e7c0

LBA[50]: 0x01013300 0x005fe880 0x00044400 0x006aa000 0x00824000 0x009fec00 0x001aacf0 0x00669300 0x00e1b680 0x00c7bbc0

LBA[60]: 0x0047b900 0x00094280 0x005ff040 0x00d40c80 0x00d2c600 0x0102ae40 0x0067b6c0 0x000d7780 0x00dc7e80 0x00365c00

LBA[70]: 0x01337080 0x002f2400 0x0108b600 0x00bdf000 0x00e03700 0x00850f40 0x00a83000 0x0098f2c0 0x001dff00 0x00847c40

LBA[80]: 0x0013a100 0x01175e00 0x013251c0 0x011f2880 0x00e94b00 0x00df3a00 0x0102b200 0x00eebf8c0 0x00a64100 0x0051fd40

LBA[90]: 0x01259900 0x00174640 0x000bab00 0x00629680 0x004fdf40 0x000d537c0 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

LBA[10]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[20]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[30]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[40]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[50]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[60]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[70]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[80]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

LBA[90]: 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040 0x40404040

PRE LBA LIST:

LBA[ 0]: 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 0x003f800 0x00530000 0x00738000 0x00239800 0x007a5800 0x00080400  
LBA[20]: 0x000cd400 0x0060b800 0x00242000 0x0018a800 0x00080000 0x0041fc00 0x000207000 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 0x00150400  
LBA[60]: 0x0002e4000 0x007e4000 0x00036400 0x005ff000 0x001f3800 0x00410c00 0x00329800 0x00450400  
LBA[70]: 0x0004c1000 0x003ed000 0x0047f000 0x00279c00 0x000410c00 0x00329800 0x00450400 0x003e8400 0x0076d400  
LBA[80]: 0x00182000 0x00036800 0x000b0000 0x0064f000 0x005f7000 0x00367000 0x004fc000 0x0022ac00 0x0014b400 0x001cb400  
LBA[90]: 0x0028e800 0x0010a000 0x006bd800 0x0060a000 0x0057a400 0x00530c00 0x0067a000 0x0051f800 0x0035800 0x00768400

PRE LBA LIST:

LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

---checktime校验

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: 0x4001270 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: 0xfffffb80f0000 | 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 0x003elc00 0x00434c00 0x00289c00 0x00020000 0x0042dc00 0x00423400  
LBA[70]: 0x006dbc00 0x004e7000 0x002c1400 0x00420400 0x0015c000 0x00652400 0x004e0400 0x004d7000 0x00242c00 0x00693000  
LBA[80]: 0x00539400 0x0040f400 0x0009c9000 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: 0xfffff9aa00000 | 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 0x000207000 0x00618000 0x005e6000 0x00101000  
LBA[30]: 0x0005a5400 0x00091400 0x002e9400 0x0044fc00 0x00050000 0x004f9c00 0x0049a800 0x00320000 0x007ef400 0x00329000  
LBA[40]: 0x0004bc000 0x006c9000 0x005fd000 0x00497000 0x00142000 0x005f1800 0x00320800 0x001b5800 0x0043e800 0x0076f800  
LBA[50]: 0x000675800 0x00739c00 0x00245000 0x00378400 0x00198000 0x0009b800 0x002f4c00 0x005a3400 0x00150400  
LBA[60]: 0x0002e4000 0x007e4000 0x00036400 0x005ff000 0x001f3800 0x000410c00 0x00329800 0x00450400 0x003e8400 0x0076d400  
LBA[70]: 0x0004c1000 0x003ed000 0x0047f000 0x00279c00 0x000410c00 0x00329800 0x00450400 0x004a2000 0x007c4800 0x001ld000  
LBA[80]: 0x00182000 0x00036800 0x000b0000 0x0064f000 0x005f7000 0x00367000 0x004fc000 0x0022ac00 0x0014b400 0x001cb400  
LBA[90]: 0x0028e800 0x0010a000 0x006bd800 0x0060a000 0x0057a400 0x00530c00 0x0067a000 0x0051f800 0x0035800 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: 0x9c38fb9 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: 0x9c38fb9 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: 0x9c38fb9 F2: 0x16672d2a [LOOP:

CORRECT LBA IO Time: 2023-07-04 01:13:15

ERROR LBA: 0x28c00 | HF: 0x00801273 F1: 0x9c38fb9 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: 0x9c38fb9 F2: 0x16672d2a [LOOP: 1 WRITE: 193 SECTOR: 0 / NUM: 128 / ZERO: 2]

LAST WRITE IO Time: 2023-07-04 01:13:19

Buffer addr: 0xfffff8c030000 | Physical addr: 0x383dfe30000

LBA LIST:

LBA[ 0]: 0x00156280 0x0007a800 0x000le900 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 0x000bdcc0 0x001c3d00  
LBA[40]: 0x00152500 0x0013a100 0x000f9c00 0x0007fd80 0x00102000 0x0006f300 0x00170900 0x001b9580 0x00084f00 0x0014c400  
LBA[50]: 0x001f0780 0x0001f400 0x000196c00 0x001b5e00 0x00075a80 0x00193a00 0x0011c600 0x0018b880 0x00011c00 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 0x001cdde00 0x0004e000

PRE LBA LIST:

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

DETECT READ ERROR:

LBA: 0x28c00 | HF: 0x00801273 F1: 0x9c38fb9 F2: 0x16672d2a [LOOP: 1 WRITE: 139 SECTOR: 0 / NUM: 128 / ZERO: 2]

DETECT ERROR IO Time: 2023-07-04 01:13:16

Buffer addr: 0xfffff9c30000 | 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 0x0011l8d00 0x0003e800 0x00087000 0x00129d00 0x000fb180  
LBA[30]: 0x00087280 0x00020400 0x00107600 0x001db000 0x001bef80 0x00085480 0x0015d980 0x000e7680 0x00187b00 0x001f3800  
LBA[40]: 0x00130000 0x000e8680 0x001a8e80 0x001f2600 0x000ae700 0x001cl580 0x001a7200 0x0014c200 0x00084700 0x00020900  
LBA[50]: 0x000e7400 0x000197080 0x000f2680 0x0006ae00 0x0ce462e3e 0x00007a80 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 0x000bdcc0 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 / 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**

---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: 0xffffd04030000 | Physical addr: 0x23503c0000

LBA LIST:

LBA[ 0]: 0x0005fa80 0x0004c000 0x00123c00 0x00080000 0x001bb000 0x00008200 0x001lecc00 0x000dc800 0x00148700 0x00126500  
LBA[10]: 0x00084000 0x0000fe00 0x00146200 0x00089000 0x001ad500 0x00069e00 0x00126000 0x0003f400 0x00178200 0x001de900  
LBA[20]: 0x001fe00 0x00189080 0x0002a880 0x001b7400 0x000ceaa0 0x000b4180 0x00016880 0x00011b880 0x000e2000 0x001c8680  
LBA[30]: 0x00061e00 0x00021ff00 0x00179980 0x00034e80 0x0012ff00 0x0008c680 0x001fcf00 0x00056080 0x0006c300 0x001d6280  
LBA[40]: 0x001d9380 0x00167000 0x0008c400 0x00191400 0x001b0800 0x001d1l00 0x001le6d00 0x00192c80 0x00181480 0x001fb800  
LBA[50]: 0x00119c200 0x0000b4200 0x000a0280 0x00014800 0x000c9600 0x0016a000 0x001f7800 0x0015a100 0x000650280 0x00164000  
LBA[60]: 0x0012e780 0x0000e900 0x0016e600 0x0001d400 0x0010ee00 0x00140100 0x00021e00 0x0006100 0x00152680 0x00110000  
LBA[70]: 0x000e0400 0x00140180 0x00026500 0x001e5400 0x000cc700 0x0010d800 0x0015180 0x0014da00 0x00021000 0x001d2e80  
LBA[80]: 0x000b5b80 0x00086380 0x00181400 0x00066280 0x00058a80 0x00005700 0x00090a80 0x0019db80 0x001d3a00 0x0007e080  
LBA[90]: 0x00092100 0x00045680 0x00144300 0x0007f1680 0x000da300 0x001fbcc0 0x00186d00 0x0018680 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: 0xffffeee30000 | Physical addr: 0x24b3ba0000

LBA LIST:

LBA[ 0]: 0x00000380 0x000e0000 0x000d7900 0x00064880 0x001c2600 0x0002fa00 0x0007f880 0x00042200 0x0010ed80 0x0015ec00  
LBA[10]: 0x001b1800 0x0010a980 0x001c3e00 0x0011l800 0x0012ad80 0x0002c900 0x001le880 0x00071800 0x001c6000 0x001db400  
LBA[20]: 0x00031480 0x000fb380 0x000e2200 0x0009d880 0x000f3400 0x00035000 0x0007a00 0x0017c000 0x00007e00 0x0005fa80  
LBA[30]: 0x0004c000 0x00123c00 0x00080000 0x001bb000 0x00008200 0x000lecc00 0x000dc800 0x00148700 0x00126500 0x00084000  
LBA[40]: 0x000fe00 0x00146200 0x00089000 0x001ad500 0x00069e00 0x00126000 0x0003f400 0x00178200 0x001de900 0x00119c200  
LBA[50]: 0x00189080 0x0002a880 0x001b7400 0x000ceaa0 0x000b4180 0x00016880 0x00011b880 0x000e2000 0x001c8680 0x0012e780  
LBA[60]: 0x00021ff00 0x00179980 0x00034e80 0x0012ff00 0x0008c680 0x001fcf00 0x00056080 0x0006c300 0x001d6280 0x00092100  
LBA[70]: 0x00167000 0x0000b4200 0x000a0280 0x00014800 0x000c9600 0x0016a000 0x001f7800 0x00151a00 0x00050280 0x00164000 0x0012e780  
LBA[80]: 0x000b5b80 0x00086380 0x00181400 0x00066280 0x00058a80 0x00005700 0x00090a80 0x0019db80 0x001d3a00 0x0007e080  
LBA[90]: 0x00090900 0x0016e600 0x0001d400 0x001fbcc0 0x00186d00 0x0018680 0x001bad80 0x000be280

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 / ZI  
PREV IO Time: 2023-06-25 20:11:47

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

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

```
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: 0xfffff98090000 | 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 0x0025c00 0x00057000 0x00531800 0x002fc000 0x00631800 0x005b5000 0x0050a000 0x004a8000
LBA[30]: 0x002af000 0x004fc00 0x00581c00 0x00060000 0x0037b400 0x0008fc00 0x003cf800 0x007a5400 0x003db400 0x00799400
LBA[40]: 0x002bf000 0x003e7c00 0x000e7800 0x005ea800 0x0013ec00 0x003c6400 0x00070000 0x00130000 0x002cac00 0x001d400
LBA[50]: 0x00446800 0x00787000 0x0072ac00 0x002a6800 0x002c2800 0x00231800 0x002c700 0x007a0c00 0x007b7b00 0x007a8400
LBA[60]: 0x00452400 0x0054d000 0x00389000 0x002c5800 0x0002c5800 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 0x002c200 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: 0xfffff6fd90000 | Physical addr: 0x50298440000

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

---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: 0xfffff756e0000 | 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 0x006alc00 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 0x007alc00 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: 0xfffff74ba0000 | 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 0x00b38ffa0 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 0x007alc00 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 0x00160800 0x000d5c00

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

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  
  
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 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 0x00bcd400 0x009f2800 0x006eb400 0x00bf1400 0x00e5f800 0x00ab0d000 0x01399400 0x00ee400  
LBA[10]: 0x00ae5c00 0x008ec000 0x0074400 0x00040800 0x00534400 0x01245c00 0x0055b400 0x0014b000 0x00c4a800 0x00276000  
LBA[20]: 0x0066e000 0x008ac2c00 0x0090c400 0x00a36c00 0x003fc800 0x007b1000 0x00acb000 0x00ccfc00 0x00a85800 0x007a1400  
LBA[30]: 0x00416c00 0x0083b400 0x0178000 0x003a6400 0x0068d800 0x01160400 0x010ddc00 0x009e7400 0x0081lc00  
LBA[40]: 0x00685000 0x0112ac00 0x00820c00 0x012db000 0x0129c000 0x012cd400 0x006c7c00 0x01116000 0x00caf400 0x003d5800  
LBA[50]: 0x00bd9000 0x0095dc00 0x0013a7400 0x00f70000 0x00958400 0x007f7900 0x01222400 0x00c7e800 0x00ff7800 0x009315c00  
LBA[60]: 0x00192000 0x0011b000 0x00274800 0x00114000 0x0004c000 0x00b3a000 0x0090c800 0x001d8400 0x01353400 0x00795000  
LBA[70]: 0x0000e000 0x01le8000 0x008c8000 0x00719000 0x012e7000 0x00d46800 0x0099f800 0x012e3400 0x01233800 0x00ed6800  
LBA[80]: 0x00226800 0x0101d400 0x01301400 0x010cac00 0x00fd2d00 0x0120a400 0x008d2400 0x009888000 0x00231800 0x00885000  
LBA[90]: 0x00140800 0x0053e400 0x00c2c800 0x00bb0000 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 0x00bd7000 0x002c9400 0x00a7d800 0x002bd400 0x00d61400 0x00058c00 0x00485800  
LBA[10]: 0x00760400 0x0016f800 0x00b89800 0x008ad400 0x01316000 0x0012e800 0x0023ec00 0x00404800 0x0008c00 0x0040f400  
LBA[20]: 0x001260800 0x00a6fc00 0x001605800 0x00447800 0x0025c00 0x00f31c00 0x0038b400 0x003d4000 0x009bcc00 0x00401c00  
LBA[30]: 0x00b8c000 0x00ec6800 0x008c3000 0x00731400 0x008d3000 0x011ae000 0x00974400 0x011ab800 0x00b04800 0x009a800  
LBA[40]: 0x00e49c00 0x0121f000 0x0096cc00 0x00040000 0x00e19000 0x00a15000 0x00af0c00 0x00345000 0x00651400 0x0093f800  
LBA[50]: 0x002ba00 0x0116c400 0x00c1fc00 0x00317000 0x00a97400 0x0060c200 0x0088200 0x01140800 0x00786c00  
LBA[60]: 0x0002c2400 0x00676400 0x00d4bc00 0x013e8000 0x012d2000 0x009494c00 0x00fb0400 0x00c7c400 0x005dc400 0x0109dc00  
LBA[70]: 0x0010d1000 0x0005cc00 0x0061a400 0x007f9400 0x01072800 0x00399c00 0x0024c000 0x002fd00 0x01244800  
LBA[80]: 0x00001e400 0x0003a4000 0x00c65800 0x00626800 0x0099b800 0x0093a8000 0x00838000 0x00422400 0x00818400 0x0001c400  
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  
  
Buffer addr: 0xffff8920e000 | Physical addr: 0x2026da5e000  
  
LBA LIST:  
LBA[ 0]: 0xd9d9d9d9  
LBA[10]: 0xd9d9d9d9  
LBA[20]: 0xd9d9d9d9  
LBA[30]: 0xd9d9d9d9  
LBA[40]: 0xd9d9d9d9  
LBA[50]: 0xd9d9d9d9  
LBA[60]: 0xd9d9d9d9  
LBA[70]: 0xd9d9d9d9  
LBA[80]: 0xd9d9d9d9  
LBA[90]: 0xd9d9d9d9  
  
PRE LBA LIST:  
LBA[ 0]: 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

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 0x011cf00 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 / 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
```

---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
LAST CORRECT WRITE:
LBA: 0x0ff4c00 | 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: 0xffffea4ee0000 | Physical addr: 0x42bd030000

LBA LIST:
LBA[ 0]: 0x00ff4c00 0x00cd5c00 0x00ad7400 0x0019d000 0x000d0c300 0x009080000 0x008df800 0x01239000 0x002b7400
LBA[10]: 0x00ba4400 0x0082cc00 0x00b3c300 0x00ee800 0x0078e800 0x00e4dc00 0x00e4dc400 0x0055f800 0x0135ec00 0x00989400
LBA[20]: 0x00ed4d00 0x006d7400 0x00f37400 0x00a5c500 0x003e5400 0x00cz400 0x006bd000 0x0fe6c00 0x01eb800 0x006a6c00
LBA[30]: 0x0018e800 0x00faa400 0x00ce1c00 0x0045c000 0x0127e600 0x002b8000 0x00368e00 0x010d6400 0x00f55800 0x0006f1000
LBA[40]: 0x005b7c00 0x0124dc00 0x0070c000 0x00318c00 0x0085dc00 0x0099a400 0x004b9c00 0x00337c00 0x00456400 0x010a9400
LBA[50]: 0x005af000 0x0048f400 0x00089400 0x001ec400 0x00cc2b400 0x00fe3400 0x00396c00 0x0064c400 0x00b03800 0x0033bc00
LBA[60]: 0x002b8000 0x002dc400 0x00db3800 0x00db3800 0x00445800 0x005dc00 0x00dc0c00 0x01031400 0x0102d400 0x00818c00
LBA[70]: 0x00d7b400 0x00c09400 0x00d6e600 0x00253400 0x00e61400 0x00f8f000 0x00ac000 0x00848c00 0x00116400 0x007adc00
LBA[80]: 0x00e2e800 0x0056dc00 0x0088c800 0x012a800 0x0129a800 0x00163800 0x01329400 0x01124400 0x00f95400 0x002ffc00
LBA[90]: 0x0003d400 0x0110f000 0x00a5c800 0x00fdcf00 0x00e2b800 0x002cc000 0x00c2b00 0x009ef400 0x00981c00 0x005b3000

PRE LBA LIST:
LBA[ 0]: 0x0110c800 0x0015d000 0x003ff400 0x003b1c00 0x0112b400 0x01b28c00 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: 0xffffea43a0000 | Physical addr: 0x435adb0000

LBA LIST:
LBA[ 0]: 0x0063f400 0x00cec400 0x00fb3000 0x01led800 0x00dab000 0x00483c00 0x0028ac00 0x001b5800 0x00a12800 0x00183800
LBA[10]: 0x00668000 0x00bb4c00 0x0131fc00 0x001ef000 0x006c7c00 0x00234c00 0x006b9800 0x008f9800 0x00057000 0x00a56800
LBA[20]: 0x00957c00 0x00857400 0x00af5000 0x0052fc00 0x00db5000 0x0082a000 0x00da5000 0x00b5c000 0x00953400 0x00d2c000
LBA[30]: 0x002ca400 0x005cc00 0x00415800 0x010cd400 0x00368400 0x0011bc00 0x00a73800 0x002e1800 0x006e9000 0x00c69c00
LBA[40]: 0x003c3000 0x00ec3400 0x005f3c00 0x001lad400 0x0012b800 0x01067800 0x00cd000 0x00ab6800 0x008ea800 0x001b6400
LBA[50]: 0x00837c00 0x00316400 0x00666000 0x013f9000 0x0057cc00 0x011ee400 0x01055000 0x00f16c00 0x01333000 0x00cb7400
LBA[60]: 0x001ld400 0x00a2l200 0x00a66000 0x003c3dc00 0x0108e400 0x003eac00 0x00fabc00 0x008c1000 0x00c6c800 0x00147400
LBA[70]: 0x0028bc00 0x00d21400 0x00701400 0x01ld0400 0x013f5000 0x0099b800 0x00febcb0 0x01298800 0x009a5000 0x008d4400
LBA[80]: 0x01135c00 0x00e1e800 0x00924000 0x00b51c00 0x003a1400 0x009b2800 0x0096c800 0x005c1000 0x0036b400 0x0138ec00
LBA[90]: 0x012bf800 0x00bb1400 0x00186800 0x008cd00 0x00bla800 0x00d6f000 0x01018400 0x012a6c00 0x00a7cc00 0x00f1d400

PRE LBA LIST:
LBA[ 0]: 0x011cf00 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: 0xffffea43f0e00 | Physical addr: 0x443ac80e00

LBA LIST:
LBA[ 0]: 0x0063f400 0x00cec400 0x00fb3000 0x01led800 0x00dab000 0x00483c00 0x0028ac00 0x001b5800 0x00a12800 0x00183800
LBA[10]: 0x00668000 0x00bb4c00 0x0131fc00 0x001ef000 0x006c7c00 0x00234c00 0x006b9800 0x008f9800 0x00057000 0x00a56800
LBA[20]: 0x00957c00 0x00857400 0x00af5000 0x0052fc00 0x042b99760 0x0082a000 0x00da5000 0x00b5c000 0x00953400 0x00d2c000
LBA[30]: 0x002ca400 0x005cc00 0x00415800 0x010cd400 0x00368400 0x0011bc00 0x00a73800 0x002e1800 0x006e9000 0x00c69c00
LBA[40]: 0x003c3000 0x00ec3400 0x005f3c00 0x001lad400 0x0012b800 0x01067800 0x00cd000 0x00ab6800 0x008ea800 0x001b6400
LBA[50]: 0x00837c00 0x00316400 0x00666000 0x013f9000 0x0057cc00 0x011ee400 0x01055000 0x00f16c00 0x01333000 0x00cb7400
LBA[60]: 0x001ld400 0x00a2l200 0x00a66000 0x003c3dc00 0x0108e400 0x003eac00 0x00fabc00 0x008c1000 0x00c6c800 0x00147400
LBA[70]: 0x0028bc00 0x00d21400 0x00701400 0x01ld0400 0x013f5000 0x0099b800 0x00febcb0 0x01298800 0x009a5000 0x008d4400
LBA[80]: 0x01135c00 0x00e1e800 0x00924000 0x00b51c00 0x003a1400 0x009b2800 0x0096c800 0x005c1000 0x0036b400 0x0138ec00
LBA[90]: 0x012bf800 0x00bb1400 0x00186800 0x008cd00 0x00bla800 0x00d6f000 0x01018400 0x012a6c00 0x00a7cc00 0x00f1d400

PRE LBA LIST:
LBA[ 0]: 0x011cf00 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 / 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
```

---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: 0xffffe9d70000 | Physical addr: 0x203705230000

LBA LIST:  
LBA[ 0]: 0x009eb800 0x018d5400 0x005eb800 0x0221b400 0x00490c00 0x016elc00 0x00dbf400 0x01703c00 0x02573000 0x0129f000  
LBA[10]: 0x0fb0400 0x01a1b800 0x04fb800 0x01d6fc00 0x00902a800 0x00deec00 0x01df1800 0x00e29000 0x012fc400 0x0011bc00  
LBA[20]: 0x15ac800 0x00683c00 0x018cb400 0x01cb2c00 0x0027c800 0x021a5400 0x00457400 0x026cd00 0x026abc00 0x02546c00  
LBA[30]: 0x01dc800 0x00ec7c00 0x0202bc00 0x01f7f400 0x021f59c00 0x02ef800 0x01f91800 0x01ef4c00 0x025fb000 0x0006b400  
LBA[40]: 0x002bcc00 0x016b5c00 0x017f5000 0x013fb000 0x009dd6400 0x02087900 0x0221d900 0x01919c00 0x00aa34a00 0x00661c00  
LBA[50]: 0x005b7b00 0x02734c00 0x00a3b400 0x02656000 0x00476400 0x0059c400 0x0058b000 0x0258b000 0x00e9d400 0x0176a00  
LBA[60]: 0x162c400 0x007e3c00 0x01add800 0x00008000 0x0063a900 0x02629000 0x01838000 0x021dd400 0x00688c00  
LBA[70]: 0x01b36800 0x00dd7900 0x00f02800 0x01757800 0x0027e800 0x02385000 0x008fb000 0x0084b000 0x00107c00 0x02048c00  
LBA[80]: 0x0017b400 0x0161b000 0x019d9800 0x012c8000 0x00fd12000 0x01338c00 0x018bd400 0x02429800 0x01f1c00 0x00bb5b00  
LBA[90]: 0x000ac800 0x00678c00 0x00797400 0x025f3c00 0x01b1b83800 0x00ad6800 0x01376c00 0x00136000 0x01be7c00 0x0101d400

PRE LBA LIST:  
LBA[ 0]: 0x006e1800 0x00b3f000 0x02249c00 0x01e6b800 0x007c4800 0x0098b00 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: 0xffffe910000 | Physical addr: 0x2025880a0000

LBA LIST:  
LBA[ 0]: 0x02577c00 0x01c3a800 0x0059b400 0x006f5000 0x026fc00 0x02345000 0x01603400 0x027ee000 0x02387c00 0x01340400  
LBA[10]: 0x01ff4000 0x01b6a000 0x006a8800 0x01b49400 0x021b8000 0x013b6800 0x00648400 0x00e91800 0x014da400 0x0146e400  
LBA[20]: 0x01eda800 0x0265f000 0x021bc800 0x027fb800 0x00aa44000 0x023bd800 0x00945400 0x01393000 0x00dc8400 0x00fd0400  
LBA[30]: 0x01b05800 0x003fc00 0x007b9000 0x004ee000 0x0002a000 0x01ad2400 0x017a2000 0x010c9800 0x01fff400 0x0064c400  
LBA[40]: 0x00716400 0x00072900 0x01b49000 0x023da000 0x00eecc000 0x00248800 0x01b47800 0x001a0e00 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 0x022dc00  
LBA[70]: 0x00309000 0x02489000 0x01091800 0x002628000 0x00e95400 0x0225f800 0x021c14800 0x01818800  
LBA[80]: 0x02537800 0x00338800 0x00599c00 0x0146ec00 0x0197a400 0x02014800 0x01ef5c00 0x00792c00 0x01e90000 0x020ae000  
LBA[90]: 0x023bec00 0x01ff79000 0x00ebf000 0x00cc1800 0x02587800 0x007c0400 0x01a9f800 0x00be1400 0x024db000 0x016b1000

PRE LBA LIST:  
LBA[ 0]: 0x00853400 0x009e0800 0x00e46000 0x02578000 0x026a3400 0x01a06400 0x01c42000 0x0036e400 0x01bb5800 0x00b99c00

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

Buffer addr: 0xffffe91dd00 | Physical addr: 0x20258265da00

LBA LIST:  
LBA[ 0]: 0x02577c00 0x01c3a800 0x0059b400 0x006f5000 0x026fc00 0x02345000 0x01603400 0x027ee000 0x02387c00 0x01340400  
LBA[10]: 0x01ff4000 0x01b6a000 0x006a8800 0x01b49400 0x021b8000 0x013b6800 0x00648400 0x00e91800 0x014da400 0x0146e400  
LBA[20]: 0x01eda800 0x0265f000 0x021bc800 0x027fb800 0x00aa44000 0x023bd800 0x00945400 0x01393000 0x00dc8400 0x00fd0400  
LBA[30]: 0x01b05800 0x003fc00 0x007b9000 0x004ee000 0x0002a000 0x01ad2400 0x017a2000 0x010c9800 0x01fff400 0x0064c400  
LBA[40]: 0x00716400 0x00072900 0x01b49000 0x023da000 0x00eecc000 0x00248800 0x01b47800 0x001a0e00 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 0x022dc00  
LBA[70]: 0x00309000 0x02489000 0x01091800 0x002628000 0x00e95400 0x020fc800 0x00bf9a000 0x0226d400 0x021c14800 0x01818800  
LBA[80]: 0x02537800 0x00338800 0x00599c00 0x0146ec00 0x0197a400 0x02014800 0x01ef5c00 0x00792c00 0x01e90000 0x020ae000  
LBA[90]: 0x023bec00 0x01ff79000 0x00ebf000 0x00cc1800 0x02587800 0x007c0400 0x01a9f800 0x00be1400 0x024db000 0x016b1000

PRE LBA LIST:  
LBA[ 0]: 0x00853400 0x009e0800 0x00e46000 0x02578000 0x026a3400 0x01c42000 0x0036e400 0x06c3ae975 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 0x115dc00 disk/file
```

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  
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: 64 / ZERO: 2]  
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: 0xfffff6c050000 | Physical addr: 0x28308370000  
  
LBA LIST:  
LBA[ 0]: 0x013691c0 0x00fa2f80 0x013c3880 0x00711d80 0x001fb000 0x0087f800 0x003ad400 0x00455000 0x0013c180 0x0121b480  
LBA[10]: 0x00fb7400 0x0044b640 0x006d0640 0x00aa8900 0x004e7840 0x00b01140 0x003d040 0x0093da00 0x00f5e480 0x00a8a640  
LBA[20]: 0x00a85700 0x00ecce480 0x011c2980 0x000f6b00 0x00530600 0x010f7c80 0x0014da00 0x010e6300 0x00073c00 0x00788d00  
LBA[30]: 0x001e7300 0x002c0e40 0x00309b80 0x012d7000 0x0047a000 0x01065a00 0x0038cf40 0x00d79380 0x00ab5c00  
LBA[40]: 0x00683000 0x01289b40 0x0030d780 0x013d6300 0x005200c0 0x00a75180 0x00a34e40 0x00521e00 0x00da4f80 0x000184c0  
LBA[50]: 0x005c5600 0x00940000 0x00f35000 0x00edc1f00 0x00dc3140 0x005bf000 0x00640040 0x006f9180  
LBA[60]: 0x00193300 0x003a2000 0x00aa8180 0x011f9840 0x0034a140 0x00f02000 0x0096fc00 0x00d197c0 0x00074380 0x0054e980  
LBA[70]: 0x006c5280 0x01115700 0x00af5e80 0x00ed1b00 0x00992a80 0x00cc1a80 0x006d9140 0x000cd500 0x012b8080 0x00df0900  
LBA[80]: 0x00068900 0x004e83c0 0x00ee2e00 0x00f67e00 0x01161d00 0x004a2b80 0x008de900 0x00678300 0x001d9fc0  
LBA[90]: 0x008ec740 0x0072eac0 0x0132b4c0 0x013d8040 0x00166040 0x00bb7f80 0x012cfaa40 0x011e3940 0x01325540 0x0035a440  
  
PRE LBA LIST:  
LBA[ 0]: 0x00fb6d00 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: 0xfffff6c070000 | Physical addr: 0x2005f1b0000  
  
LBA LIST:  
LBA[ 0]: 0x0057ef00 0x0115dc00 0x00176880 0x0136fc00 0x00ac8480 0x0119f1c0 0x00e129c0 0x00289a40 0x00fbba000 0x00d3fd00  
LBA[10]: 0x00cf7700 0x00727240 0x00ec27c0 0x00afe200 0x00aa8380 0x003d0c80 0x0084100 0x00e4d40 0x00bc8c00 0x00606000  
LBA[20]: 0x0103f80 0x004a0740 0x00686000 0x00999900 0x00aba040 0x00f03540 0x00081440 0x00d33b80 0x00682800  
LBA[30]: 0x00785440 0x005c1d80 0x006dfdc0 0x00337540 0x00cfe100 0x00cf7740 0x01298940 0x0094a800 0x00791640 0x00f081c0  
LBA[40]: 0x0054b6c0 0x0023f940 0x0022d400 0x0072c200 0x00dfd000 0x00a43c00 0x005676c0 0x01069580 0x01388500  
LBA[50]: 0x00383d80 0x0124e200 0x001le7100 0x0032f000 0x00658a40 0x00ec9800 0x00fbfa00 0x00b8780 0x00660540 0x0124f00  
LBA[60]: 0x0025b800 0x00fabcc0 0x012ff740 0x0034d800 0x00cd9800 0x00023900 0x004b8e40 0x007ec800 0x00768040 0x003a8000  
LBA[70]: 0x0054f000 0x01077940 0x009b4d80 0x00089b2f00 0x0137de00 0x0049ec0 0x00e46400 0x01258200 0x012c800 0x011bd5c0  
LBA[80]: 0x0100d500 0x002bd540 0x01007000 0x00ecf00 0x00584800 0x00fd4980 0x004e1c00 0x003e4800 0x000f0000 0x0084b80  
LBA[90]: 0x00500c00 0x0035dd40 0x01398000 0x00698000 0x00bc4280 0x006fb800 0x00cb4f00 0x009fb800 0x006ece80 0x0110a500  
  
PRE LBA LIST:  
LBA[ 0]: 0x001f2200 0x01099d80 0x00a47680 0x00829800 0x00f3f300 0x00529880 0x0091e2c0 0x00f7be00 0x00a00880 0x013bc800
```

```
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: 0xfffff6c077e00 | Physical addr: 0x2005f1b7e00  
  
LBA LIST:  
LBA[ 0]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[10]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[20]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[30]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[40]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[50]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[60]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[70]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[80]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c  
LBA[90]: 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c 0x7c7c7c7c
```

```
Current Time: 2023-07-03 18:11:13  
pause_and_exit_work: Press KEY: <0> 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
```

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 0x012edlc0 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
```

```
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: 0xffffeb8050000 | Physical addr: 0x373df70000

```
LBA LIST:  
LBA[ 0]: 0x01185880 0x00489e00 0x0119c000 0x013fb00 0x00aae000 0x00cf5480 0x005044c0 0x01359180 0x004e5400 0x00015000
LBA[10]: 0x00992740 0x00eef100 0x00e3f580 0x00344600 0x00c8b680 0x0111b80 0x0059e340 0x00dc50c0 0x01039a00 0x001ae980
LBA[20]: 0x00507000 0x01019380 0x00d81240 0x00f08380 0x0021ac80 0x00045800 0x0088d600 0x00e9b800 0x00c3d900 0x009d1840
LBA[30]: 0x0043900 0x01030300 0x00d84480 0x004ca200 0x00336c00 0x00d35080 0x0096c480 0x006e6ec0 0x00e27080 0x00b4da80
LBA[40]: 0x00bf4c80 0x00424c00 0x0094a400 0x0044c080 0x00d7e400 0x0058d7c0 0x00aa0a40 0x00f27640 0x00435ac0 0x01305c00
LBA[50]: 0x00957a40 0x00a2900 0x00354880 0x008b0880 0x01269680 0x0125f840 0x006b6600 0x001d1880 0x00846a80 0x00c1e4c0
LBA[60]: 0x003e8980 0x010f0800 0x0098e640 0x00898380 0x01022cc0 0x00f0dc0 0x008da4c0 0x008cb600 0x00591440
LBA[70]: 0x0187c700 0x002a2e640 0x0089cc40 0x00631440 0x00d6f600 0x009e0400 0x00474c60 0x003ed380 0x00c36200 0x0005de00
LBA[80]: 0x01395e40 0x01916180 0x009fcce0 0x0105fffc0 0x0087ca00 0x00fd1e80 0x003t7700 0x00a9880 0x007f7300 0x0006f280
LBA[90]: 0x01fb4c0 0x002d8380 0x0093c440 0x00abb40 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: 0xffffeb8070000 | Physical addr: 0x2544d50000

```
LBA LIST:  
LBA[ 0]: 0x0115e140 0x00bef680 0x00b1d980 0x00211680 0x01301900 0x013ad480 0x0032a580 0x01217dc0 0x003c0d00 0x008cb180
LBA[10]: 0x010bd180 0x00a6ef00 0x00458740 0x0082eac0 0x00ecd80 0x0129b40 0x00040640 0x00e5cd80 0x0235200 0x002e69c0
LBA[20]: 0x00ed5d80 0x01088000 0x00920780 0x000924c0 0x000924a900 0x00094900 0x01137a00 0x0134fec0 0x0001e2c0 0x008658c0 0x013491c0
LBA[30]: 0x00d6d8080 0x01b08a80 0x01216c00 0x0097f4300 0x0020b880 0x002e6f00 0x000bc880 0x0076c200 0x0122d200 0x010de100
LBA[40]: 0x000cf580 0x000ae4800 0x00072a900 0x00bf6180 0x00b2c200 0x0009b80 0x00f27440 0x003d2d80 0x00066680
LBA[50]: 0x007908c0 0x00391a40 0x00210200 0x01158c00 0x00619a80 0x008cd300 0x008efb80 0x01090a00 0x011ef240 0x00c6a800
LBA[60]: 0x00ee2000 0x005f61c0 0x0024d780 0x00d0980 0x00b3c80 0x003fa800 0x002c9d80 0x00039800 0x009ce040 0x01269b00
LBA[70]: 0x00f48040 0x01b9300 0x01314220 0x0048d000 0x006f2280 0x00121840 0x00ae0c80 0x00e6d180 0x00551200 0x010d4240
LBA[80]: 0x00410040 0x0007fd40 0x011e0c0 0x00aaed00 0x0008580 0x0008d5a00 0x0126eccc0 0x000db780 0x011df280 0x00554780
LBA[90]: 0x009b8780 0x019c2c0 0x012f15c0 0x01154ac0 0x011da100 0x0063d80 0x00841300 0x00ab7c0 0x00fd6680 0x0051f000
```

```
PRE LBA LIST:  
LBA[ 0]: 0x012edlc0 0x00fe0880 0x00844280 0x00bb740 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: 0xffffeb8075e00 | Physical addr: 0x2544d55e00

```
LBA LIST:  
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[80]: 0x55ab6910 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
```

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

Current Time: 2023-07-03 20:04:19

pause\_and\_exit\_work: Press KEY: <0> 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 0x08cf800 disk/file
hd_write_verify_dump -c -D -L 0x0de7800 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 CONNECT 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: 0xfffffee4430000 | Physical addr: 0x2824e20000

LBA LIST:  
 LBA[ 0]: 0x0039d800 0x00724000 0x00011800 0x00a21800 0x0023d800 0x012a800 0x005eb800 0x0000e000 0x00f8a800 0x00af5000  
 LBA[10]: 0x009ad000 0x002cc800 0x00fd0a00 0x00f4000 0x00767800 0x005c6800 0x00402000 0x007d8800 0x01242800 0x0010b000  
 LBA[20]: 0x00322800 0x010e3000 0x00c9a800 0x00fc0c00 0x009dc800 0x00217000 0x0013f7000 0x00de6800 0x0094d800  
 LBA[30]: 0x0033d800 0x0028e000 0x00cf800 0x009e2000 0x00670000 0x00e0f800 0x01253800 0x012fa000 0x00f59800 0x00481000  
 LBA[40]: 0x00e1d800 0x005d6000 0x0119f000 0x0135f000 0x0094e000 0x0113f800 0x00424000 0x00964000 0x00fa5800 0x00a3a3000  
 LBA[50]: 0x009ad000 0x008c5800 0x0001d800 0x002dc000 0x0112d800 0x00af1800 0x000f8000 0x00456800 0x00103000 0x00126000  
 LBA[60]: 0x00e37000 0x006e8000 0x00289000 0x00995000 0x00a89800 0x0029d000 0x00e77800 0x00185000 0x0058a800 0x001c4800  
 LBA[70]: 0x00a8c000 0x0044c800 0x00e5b800 0x00dd6000 0x0092800 0x01036000 0x00648800 0x01236000 0x00b79000  
 LBA[80]: 0x00b68000 0x0041d800 0x00995800 0x00dc6000 0x00a9a900 0x0117d800 0x0069d000 0x001df800 0x010cf800 0x005ff000  
 LBA[90]: 0x0080b800 0x00434000 0x012df000 0x01395800 0x003b5000 0x00f2f000 0x00c87800 0x002b9000 0x0055f000 0x005d1800

PRE LBA LIST:  
 LBA[ 0]: 0x0074b000 0x0000a0000 0x009cc0000 0x00fc0000 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: 0xfffffea0180000 | Physical addr: 0x21e5cd0000

LBA LIST:  
 LBA[ 0]: 0x00b2a000 0x00de7800 0x000f2000 0x010f3800 0x01041800 0x00e43000 0x0028a000 0x011c3000 0x0014e000 0x009d0800  
 LBA[10]: 0x0110a800 0x00fc0000 0x009cc000 0x0000a000 0x0074b000 0x0039d800 0x00724000 0x00011800 0x00a21800 0x0023d800  
 LBA[20]: 0x012a8000 0x005eb800 0x0000e000 0x00f8a800 0x00af5000 0x009ad000 0x002cc800 0x00fd0a00 0x000f4000 0x00767800  
 LBA[30]: 0x005c6800 0x000402000 0x007d8800 0x01242800 0x010e3000 0x009dc800 0x0013f7000 0x00de6800 0x00a3a3000  
 LBA[40]: 0x00217000 0x0013f000 0x003b7000 0x00de6800 0x0094d800 0x00424000 0x00964000 0x00fa5800 0x00481000  
 LBA[50]: 0x0000f800 0x01253800 0x012fa000 0x00f759800 0x00481000 0x00e1d800 0x005d6000 0x0119f000 0x0135f000 0x0094e000  
 LBA[60]: 0x0113f800 0x00424000 0x00964000 0x00fa5800 0x00ba03000 0x00a9d000 0x008c5800 0x0001d800 0x002dc000 0x0112d800  
 LBA[70]: 0x00a8c000 0x00456800 0x0001d800 0x01036000 0x00e37000 0x006e6800 0x00289000 0x00995000 0x00a89800  
 LBA[80]: 0x0029d000 0x00e77800 0x00185000 0x0058a800 0x001c4800 0x0092800 0x00648800 0x00dd6000 0x00f2800  
 LBA[90]: 0x01036000 0x00648800 0x0084b800 0x01236000 0x00b79000 0x006b6800 0x0041d800 0x00995800 0x00dc6000 0x00a9a900

PRE LBA LIST:  
 LBA[ 0]: 0x008cf800 0x000f1800 0x01032800 0x001aa800 0x00113800 0x00710800 0x00fe4000 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: 0xfffffea01a6400 | Physical addr: 0x237d7a6400

LBA LIST:  
 LBA[ 0]: 0x00000000  
 LBA[10]: 0x00000000  
 LBA[20]: 0x00000000  
 LBA[30]: 0x00000000  
 LBA[40]: 0x00000000  
 LBA[50]: 0x00000000  
 LBA[60]: 0x00000000  
 LBA[70]: 0x00000000  
 LBA[80]: 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 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: 0xffffeb14d0000 | Physical addr: 0x53a2320000

LBA LIST:
LBA[ 0]: 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000 0x006df400 0x007df800 0x00117800 0x00310c00
LBA[10]: 0x00249400 0x005f0800 0x001cac00 0x0079400 0x0003d400 0x00503c00 0x004c800 0x001a1c00 0x00560800
LBA[20]: 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca00 0x00073800 0x00107000 0x006f77400 0x002dc00 0x00037800
LBA[30]: 0x003c3400 0x005d9800 0x0040c000 0x0008a000 0x0008a000 0x00059800 0x001a2400 0x0065bc00 0x00449c00 0x00018c00 0x00724000
LBA[40]: 0x002fc400 0x0068a000 0x001ef800 0x005f7400 0x00216000 0x006f9400 0x00182400 0x0049ab00 0x002d5400 0x0004e400
LBA[50]: 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f400 0x00366000 0x007d400 0x001c4000 0x0075400 0x00590000 0x002ad800 0x002cc400
LBA[60]: 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x00059400 0x00590000 0x002ad800 0x002cc400
LBA[70]: 0x004c4000 0x00509800 0x00191000 0x006f8400 0x0060e400 0x001a3c00 0x002c4400 0x00386000 0x00179c00 0x00421400
LBA[80]: 0x00514c00 0x00484000 0x0035f000 0x0008a400 0x00585c00 0x000b9c00 0x00173c00 0x0040b800 0x00133800 0x003fac00
LBA[90]: 0x00501000 0x00dd3800 0x007c8c00 0x00611400 0x000c7c00 0x006b9800 0x00174000 0x000fe400 0x00386c00 0x006bfcc00

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: 0xffffeb08f0000 | 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]: 0x006d7400 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001caca00 0x0079400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x0040c800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca00 0x00073800
LBA[40]: 0x00107000 0x006fd7400 0x002ddcc00 0x003c4000 0x00037800 0x003ca00 0x000409800 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x0049b800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00590400 0x00590000 0x002ad800 0x002c400 0x004ca400 0x00590000 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: 0xffffeb0922000 | Physical addr: 0x53aeff2000

LBA LIST:
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006f8c00 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006d7400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000
LBA[20]: 0x006971fb4 | 0x007df800 0x00117800 0x00310c00 0x00249400 0x005f0800 0x001caca00 0x0079400 0x0003d400 0x000e9000
LBA[30]: 0x00503c00 0x0040c800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca00 0x00073800
LBA[40]: 0x00107000 0x006fd7400 0x002ddcc00 0x003c4000 0x00037800 0x003ca00 0x000409800 0x0008a000 0x000d9800 0x001a2400
LBA[50]: 0x0065bc00 0x00449c00 0x00018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400
LBA[60]: 0x00182400 0x0049b800 0x002d5400 0x0004e400 0x0072fc00 0x00376c00 0x001be400 0x00706c00 0x00057400 0x0014f800
LBA[70]: 0x00366000 0x0079d400 0x006b3c00 0x00489c00 0x0070e800 0x006d7c00 0x0013f800 0x00401400 0x001c4000 0x007e5400
LBA[80]: 0x00590400 0x00590000 0x002ad800 0x002c400 0x004ca400 0x00590000 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: <0> 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 / NUM: 46260 / ZERO: 3031741620]  
INCONSISTENT IO Time: -1263223776--1263225675--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: 0xfffffb000000 | Physical addr: 0x2026e3060000

```
LBA LIST:  
LBA[0]: 0x00717400 0x00697800 0x00560000 0x00343000 0x00621400 0x003e8400 0x00010c00 0x002a8800 0x0066e000 0x0037ec00  
LBA[10]: 0x00317400 0x00769900 0x00738400 0x00003400 0x005d4000 0x00444800 0x00510c00 0x005c4400 0x007b1800 0x004de400  
LBA[20]: 0x004a4b00 0x007c5400 0x0007a5800 0x003b3c00 0x00044000 0x0075fc00 0x002f6f800 0x005a800 0x001a9400 0x0028ac00  
LBA[30]: 0x00254000 0x00422800 0x0002a3000 0x0016bc00 0x006f9400 0x00048c00 0x0047000 0x0000b400 0x005ed000 0x00772800  
LBA[40]: 0x00793c00 0x00415400 0x00688000 0x00095c00 0x004fe000 0x006dd400 0x0029c00 0x0012e800 0x000b4c00 0x0033b000  
LBA[50]: 0x00210400 0x0028cc00 0x000af400 0x0066e800 0x0051d800 0x0017c00 0x00713800 0x0049c00 0x00735800 0x002b0000  
LBA[60]: 0x001f8000 0x0016b400 0x000743800 0x000533400 0x00094000 0x00094000 0x0027400 0x001dc400 0x0005c0400  
LBA[70]: 0x00485800 0x00088400 0x007b7ec00 0x0053e300 0x00066c00 0x0049800 0x003f5400 0x00565400 0x00565400  
LBA[80]: 0x0028b200 0x000539400 0x0000d4800 0x003cf800 0x002f7800 0x003dc00 0x00556c00 0x0040d800 0x004f4800 0x004a0c00  
LBA[90]: 0x0077ac00 0x00347c00 0x00045c00 0x0060b400 0x0030d000 0x002a4800 0x007d8c00 0x0070c800 0x00219c00 0x00555400
```

```
PRE LBA LIST:  
LBA[0]: 0x000b2000 0x0054fc00 0x0000bd800 0x00041800 0x00639c00 0x00247c00 0x00155000 0x005c8800 0x002dbc00 0x002lc800
```

```
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: 0xfffffb380000 | Physical addr: 0x43e3fe0000

```
LBA LIST:  
LBA[0]: 0x007e9800 0x0060b000 0x003e400 0x007af400 0x0019d400 0x00648400 0x004ffcc0 0x0021c800 0x002dbc00 0x005c8800  
LBA[10]: 0x00155000 0x00247c00 0x00639c00 0x00041800 0x0000bd800 0x0054fc00 0x0000b2000 0x00717400 0x00697800 0x00560000  
LBA[20]: 0x00343000 0x00621400 0x003e8400 0x0001c00 0x002a8800 0x00044800 0x0037ec00 0x00317400 0x00769900 0x00738400  
LBA[30]: 0x000d3400 0x005d4000 0x00444800 0x0051c00 0x0045c00 0x007b1800 0x004de400 0x004ab400 0x007c5400 0x007a5800  
LBA[40]: 0x0003b3c00 0x0000e4000 0x00675fc00 0x0026f800 0x005a800 0x001a9400 0x0028ac00 0x00254000 0x00422800 0x002a3000  
LBA[50]: 0x0016b200 0x0006f9400 0x00048c00 0x00447000 0x00051d400 0x000ed400 0x00793c00 0x00415400 0x0005b800  
LBA[60]: 0x00095c00 0x0004fe000 0x0006d400 0x002220c00 0x0012e800 0x000bac00 0x003b000 0x00210400 0x0028cc00 0x000afa400  
LBA[70]: 0x00066e800 0x00051d800 0x001700c00 0x0073800 0x0049d400 0x0002b0000 0x001fb800 0x001bd400 0x000d3800  
LBA[80]: 0x000505c00 0x00133400 0x0055a000 0x0009a000 0x0022f400 0x001dc400 0x006c400 0x00485800 0x00088400 0x007bec00  
LBA[90]: 0x00053e000 0x00066c00 0x0049800 0x003f6000 0x00295800 0x00655400 0x0028b200 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
```

Buffer addr: 0xfffffb3f9400 | Physical addr: 0x45f8669400

```
LBA LIST:  
LBA[0]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[10]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[20]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[30]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[40]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[50]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[60]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[70]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[80]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[90]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4
```

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

---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 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 0x0023a000 0x007d9c00 0x0012a000 0x006a2000 0x006fbcc0 0x0065a400
LBA[20]: 0x00634000 0x005c2c00 0x0029c000 0x0075f400 0x0078f800 0x0059d400 0x006ed800 0x00336400 0x00762000 0x0025a000
LBA[30]: 0x00058c00 0x001dd800 0x00062c00 0x000f1000 0x003a8000 0x00104000 0x0007b400 0x00380800 0x00698400 0x005c9400
LBA[40]: 0x00299400 0x0014d400 0x002d3c00 0x00310000 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 0x0057e800 0x001b2400 0x006b8400 0x00599400 0x002ff400
LBA[70]: 0x000b4800 0x006cc800 0x00749800 0x00557800 0x00226400 0x0039c000 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]: 0x003elc00 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: 0x300deb2000

LBA LIST:
LBA[ 0]: 0x00615c00 0x007b3000 0x0072fc00 0x0050e000 0x00381c00 0x003elc00 0x005fa800 0x0062d000 0x00337000 0x00643800
LBA[10]: 0x00312800 0x0075c000 0x00080000 0x00323800 0x00486800 0x007d9400 0x002cc400 0x0013c400 0x007fac00 0x0023a000
LBA[20]: 0x007d9c00 0x0012a000 0x006a2000 0x000f7000 0x006fbcc0 0x0065a400 0x00634000 0x005c2c00 0x0029c000 0x0075f400
LBA[30]: 0x0078f800 0x0059d400 0x006ed800 0x00336400 0x00762000 0x0025a000 0x00058c00 0x001dd800 0x00062c00 0x000f1000
LBA[40]: 0x0003a800 0x00110400 0x0007b400 0x00380800 0x00698400 0x0059c400 0x00299400 0x0014d400 0x002d3c00 0x00331000
LBA[50]: 0x001d2800 0x00403000 0x00560400 0x0057d800 0x000b3400 0x003ec400 0x003a5000 0x00631c00 0x0077f800 0x00218000
LBA[60]: 0x000d2400 0x002eaac00 0x0051f000 0x00200800 0x0043800 0x00669c00 0x00460c00 0x000a4000 0x002aa800 0x000d4400
LBA[70]: 0x005f0800 0x0057a800 0x001b2400 0x006b8400 0x00599400 0x002ff400 0x004hb800 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 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 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: 0xfffff806f0000 | 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: 0xfffff80790000 | 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 0x005d4000  
LBA[60]: 0x009ed000 0x00e08000 0x0060dc00 0x00a49000 0x00806c00 0x010c1c00 0x002eb000 0x00be8400 0x00eb4000 0x0056e000  
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

# 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

```
BUG 004: [Thread: 5] DATA LOST | read[288]
PREV LBA: 0x359ec00 | LOST LBA: 0x20c5c00 | NEXT LBA: 0x1d3c800

PREV LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2eb00 [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
PREV IO Time: 2023-05-13 11:20:53

LOST LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
LOST IO Time: 2023-05-13 11:12:55

NEXT LBA: 0x1d3c800 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2eb00 [LOOP: 5 WRITE: 288 SECTOR: 0 / NUM: 1024 / ZERO: 208]
NEXT IO Time: 2023-05-13 11:20:54
```

LBA: 0x359ec00 | HF: 0x04001275 F1: 0xa62d2928 F2: 0xab2eb00 [LOOP: 5 WRITE: 286 SECTOR: 0 / NUM: 1024 / ZERO: 208]
IO Time: 2023-05-13 11:20:53

Buffer addr: 0x7f6faadd5000 | Physical addr: 0x422516000

LBA LIST:  
LBA[0]: 0x0359ec00 0x020c5c00 0x01d3c800 0x032a8c00 0x00fa8800 0x0043a800 0x042c5c00 0x02d9e400 0x03de2c00 0x03ab1000  
LBA[10]: 0x0139b400 0x013d3800 0x040ca400 0x03b00400 0x03020800 0x00e09c00 0x00a48c00 0x03108400 0x0498d800 0x01685400  
LBA[20]: 0x00225000 0x03af2400 0x04fb000 0x02e1c00 0x03bd1000 0x001a1000 0x00045c00 0x0033c400 0x0336f000  
LBA[30]: 0x02873400 0x030f1c00 0x02aec800 0x01864400 0x03410000 0x03b9ecc0 0x044d1000 0x02507c00 0x01bd0000 0x00ae1400  
LBA[40]: 0x03390000 0x017a2000 0x045d8400 0x03232000 0x02a08000 0x03520800 0x0398a000 0x0029e000 0x0442a000 0x002d0800  
LBA[50]: 0x00500080 0x0139c400 0x02f5ec00 0x039b1000 0x01557c00 0x02055000 0x00775000 0x0202c000 0x00e0cc00 0x03128c00  
LBA[60]: 0x03a3c400 0x047f6400 0x04040400 0x000bc000 0x03ddc000 0x03a6c000 0x0305f000 0x01e4a000 0x03dac000 0x00163000  
LBA[70]: 0x0496d000 0x04c51800 0x041fc000 0x0117a400 0x03d08800 0x0308a800 0x04925000 0x01064800  
LBA[80]: 0x020cc000 0x01e32c00 0x0455c00 0x03a0c000 0x019a9400 0x02f0c000 0x01026000 0x000bf000 0x03a1000 0x03c5b800  
LBA[90]: 0x02b82800 0x02aee400 0x00cce800 0x00d56000 0x0011c000 0x0332f000 0x04f5f000 0x02f61000

PRE LBA LIST:  
LBA[0]: 0x023f1c00 0x01fb0400 0x0027800 0x04848000 0x01281000 0x0332d800 0x03715c00 0x00c4d800 0x00534400 0x01964400

LBA: 0x20c5c00 | HF: 0x04001272 F1: 0xa62d2928 F2: 0x1d66938a [LOOP: 4 WRITE: 8030 SECTOR: 0 / NUM: 1024 / ZERO: 142]
IO Time: 2023-05-13 11:12:55

Buffer addr: 0x7f6faae57000 | Physical addr: 0x41f880000

LBA LIST:  
LBA[0]: 0x020c5c00 0x01ccb400 0x0266cc00 0x040cb400 0x00995400 0x011e3800 0x019fb00 0x00e4e800 0x008b7c00 0x0136ec00  
LBA[10]: 0x01958000 0x02b7dc00 0x011fd000 0x00946900 0x0340b100 0x048bf00 0x02347400 0x01b3d000  
LBA[20]: 0x0424b000 0x01cbc400 0x001cf000 0x03f2f000 0x0430b000 0x0147f200 0x0147f100 0x04357400 0x02ce4000 0x048dc000  
LBA[30]: 0x02b24e00 0x019ae400 0x00990c00 0x011fc00 0x0185d400 0x00eb6c00 0x0120100 0x04c5dc00 0x0120e000 0x00215c00  
LBA[40]: 0x0437ac00 0x013ddc00 0x048df400 0x0496f800 0x04a45000 0x032d8400 0x022a7000 0x04ca6000 0x01377000 0x01378c00  
LBA[50]: 0x0438c100 0x0199c00 0x03f32c00 0x03330000 0x04dbd00 0x01819400 0x0499f100 0x01f93c00 0x0137f400 0x0369f100  
LBA[60]: 0x01cde800 0x0323e000 0x049a6400 0x02f28800 0x0364cc00 0x02ff3400 0x03dc6000 0x03293000 0x013dcfc00 0x01391800  
LBA[70]: 0x018b2000 0x0043bac00 0x01393c00 0x01e4f400 0x0181f600 0x03b37000 0x01849000 0x0292bc00 0x0181fc00 0x0139hc00  
LBA[80]: 0x0365fc00 0x0139dc00 0x0139c000 0x045ec500 0x04b9fc00 0x0145a800 0x02b51400 0x04197c00 0x02cff800

```
hd_junit_verify -c -D -K -R 33 -S 128 -V all -T 10 -L 102400 -P split -I /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: 21474836480 / 20480M / 20,008

disk: /dev/vda | Thread: 10 | Total Sectors: 41943040 | Total Clusters: 327680 | Sectors of Cluster: 128 | DIRECT IO & NO Flush | Verify: 5 | Notify: 0
Current Time: 2023-06-13 10:03:42
Thread 7 [tid: 1250]: Starting check disk ...
Thread 6 [tid: 1249]: Starting check disk ...
Thread 1 [tid: 1244]: Starting check disk ...
Thread 5 [tid: 1248]: Starting check disk ...
Thread 2 [tid: 1245]: Starting check disk ...
Thread 9 [tid: 1252]: Starting check disk ...
Thread 8 [tid: 1251]: Starting check disk ...
Thread 4 [tid: 1247]: Starting check disk ...
Thread 0 [tid: 1243]: Starting check disk ...
Thread 3 [tid: 1246]: Starting check disk ...

Current Time: 2023-06-13 10:03:57
hd_junit_verify -c -D -K -R 33 -S 128 -V all -T 10 -L 102400 -P split -I /dev/vda
PREV LBA: 0x4a5100 - NEXT LBA: 0x35480 flags is different | read[224]
PREV LBA: 0x4a5100 | HF: 0x00801275 F1: 0x01280ba6 F2: 0xa7c44528 [LOOP: 2 WRITE: 2247 SECTOR: 0 / NUM: 128 / ZERO: 39]
PREV LBA ID Time: 2023-06-12 19:14:23
NEXT LBA: 0x35480 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
NEXT LBA ID Time: 1900-01-00 00:00:00

Current Time: 2023-06-13 10:03:57
hd_junit_verify -c -D -K -R 33 -S 128 -V all -T 10 -L 102400 -P split -I /dev/vda
PREV LBA: 0x4a5100 - NEXT LBA: 0x35480 | thread_num: 10, bitmap: 0000000000000000
PREV LBA: 0x4a5100 | HF: 0x00801275 F1: 0x01280ba6 F2: 0xa7c44528 [LOOP: 2 WRITE: 2247 SECTOR: 0 / NUM: 128 / ZERO: 39]
PREV LBA ID Time: 2023-06-12 19:14:23
Buffer addr: 0x7f183c02f000 | Physical addr: 0x423ca7000

LBA LIST:
LBA[0]: 0x000a5100 0x00035480 0x01c93780 0x0178ce00 0x00ca0800 0x01a31280 0x00870000 0x0205b000 0x02505d00 0x01ba3a00
LBA[10]: 0x00c58a00 0x00230000 0x01583600 0x02541600 0x020fb700 0x00eddef0 0x00593200 0x0267af00 0x01c4a080 0x001b3780
LBA[20]: 0x013e8a00 0x01441100 0x02126800 0x00737a00 0x00aeb300 0x00490200 0x010d3300 0x00b11000 0x0060e800
LBA[30]: 0x009cc000 0x00ec0020 0x026d900 0x00207d00 0x004dc400 0x00ce0f00 0x02163b00 0x0099fa00 0x00a5e000
LBA[40]: 0x01a17a00 0x00327000 0x01740100 0x011b3000 0x00ea5f00 0x014f1500 0x01c29580 0x00d7d500 0x01b5b500 0x01919600
LBA[50]: 0x00028f00 0x02462900 0x00923080 0x004c7080 0x0038c200 0x01c72000 0x00007600 0x0057bc00 0x015c2900
LBA[60]: 0x016f4f00 0x01419400 0x01f4f200 0x00626900 0x01076800 0x024b1400 0x008ace00 0x00efdd00 0x0141cd00 0x018a1500
LBA[70]: 0x01619100 0x00208100 0x02561800 0x01f6b600 0x00639000 0x0205b000 0x02495e00 0x01561f00 0x00e97c00
LBA[80]: 0x000e1700 0x011c8200 0x0119c200 0x025ce000 0x00280900 0x01deda00 0x0193b200 0x0163d100 0x0185f100
LBA[90]: 0x00068c00 0x02790480 0x01632400 0x02030500 0x015b500 0x00d4f700 0x018c2000 0x0020100 0x0078e00 0x00537800

PRE LBA LIST:
LBA[0]: 0x00bad100 0x023f580 0x01bf7100 0x00365280 0x02739680 0x0174ee00 0x0236d680 0x0045fa00 0x00d40800 0x01231f80

LOST LBA: 0x35480 | HF: 0x00000000 F1: 0x00000000 F2: 0x00000000 [LOOP: 0 WRITE: 0 SECTOR: 0 / NUM: 0 / ZERO: 0]
LOST LBA ID Time: 1900-01-00 00:00:00
Buffer addr: 0x7f183c01e000 | Physical addr: 0x42cd09000

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

NEXT LBA: 0x1c93780 | HF: 0x00801275 F1: 0x01280ba6 F2: 0xa7c44528 [LOOP: 2 WRITE: 2249 SECTOR: 0 / NUM: 128 / ZERO: 39]
NEXT LBA ID Time: 2023-06-12 19:14:23
Buffer addr: 0x7f183c040000 | Physical addr: 0x425918000

LBA LIST:
LBA[0]: 0x01c93780 0x0178ce00 0x00ca0800 0x01a31280 0x00870000 0x0205b000 0x02505d00 0x01ba3a00 0x00c5a800 0x0023a000
LBA[10]: 0x015a5600 0x02541600 0x020fb700 0x00eddef0 0x00593200 0x01c4a080 0x001b3780 0x01c441100
LBA[20]: 0x02126800 0x00737a00 0x00aeb300 0x00490200 0x010d3300 0x00b11000 0x0060e800 0x009cf00 0x00eecd200
LBA[30]: 0x026d900 0x02727d00 0x0027d00 0x004dc400 0x00ce0f00 0x02163b00 0x0099fa00 0x00a5e000 0x01a17a00
LBA[40]: 0x01740100 0x011b3000 0x00ea5f00 0x014f1500 0x01c29580 0x00d7d500 0x01b5b500 0x01919600 0x0028f00 0x0246f300
LBA[50]: 0x0242c400 0x00923080 0x004c7080 0x007600 0x0057bc00 0x01c72000 0x00007600 0x0057bc00 0x0141b900
LBA[60]: 0x001f4f00 0x00208100 0x00626900 0x001f7680 0x024b1400 0x008ace00 0x00efdd00 0x0141cd00 0x018a1500 0x0145f100 0x00208100
LBA[70]: 0x02561800 0x00208100 0x00626900 0x001f7680 0x024b1400 0x008ace00 0x00efdd00 0x0141cd00 0x018a1500 0x0145f100 0x00208100
LBA[80]: 0x0119c200 0x025ce000 0x0026d800 0x001d6700 0x0056700 0x002495e00 0x01561f00 0x00e97c00 0x00ef2700 0x012b2600
LBA[90]: 0x00183c400 0x0203a500 0x0015b500 0x00d4f700 0x018c2000 0x0008a000 0x001969e00 0x0101f6900

PRE LBA LIST:
LBA[0]: 0x00035480 0x004a5100 0x00bad100 0x023f580 0x001bf7100 0x00363580 0x02739680 0x0174ee00 0x0236d680 0x0045fa00

Current Time: 2023-06-13 10:03:57
pause_and_exit_work: Press KEY: <Q> to exit!
```

# 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 0x0009e9800 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 0x0006e9000 0x00353c00 0x000ae400 0x00420400 0x0028a400 0x0006a000  
LBA[50]: 0x0024f800 0x007ca400 0x00220000 0x005c5800 0x0032cc00 0x00379800 0x00518400 0x0005c4000 0x0013b000 0x0032c000  
LBA[60]: 0x00140000 0x00069400 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 0x007fd000 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

--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: 0xfffff98af0000 | 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 0x0009e9800 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 0x0006e9000 0x00353c00 0x000ae400 0x00420400 0x0028a400 0x0006a000  
LBA[50]: 0x0024f800 0x007ca400 0x00220000 0x005c5800 0x0032cc00 0x00379800 0x00518400 0x0005c4000 0x0013b000 0x0032c000  
LBA[60]: 0x00140000 0x00069400 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 0x007fd000 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

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: 0xfffff98af0200 | 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 0x0009e9800 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 0x0006e9000 0x00353c00 0x000ae400 0x00420400 0x0028a400 0x0006a000  
LBA[50]: 0x0024f800 0x007ca400 0x00220000 0x005c5800 0x0032cc00 0x00379800 0x00518400 0x0005c4000 0x0013b000 0x0032c000  
LBA[60]: 0x00140000 0x00069400 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 0x007fd000 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

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

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 -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

```
Start
0, 178 | 0, 178 | 0, 177 | 0, 174 | 0, 177 | 0, 173 | 0, 181 | 0, 184 | 0, 178 | 0, 169 |

Current Time: 2023-06-19 15:43:58
Thread 0 [tid: 2212396]: Starting check disk ...
Thread 8 [tid: 2212404]: Starting check disk ...
Thread 7 [tid: 2212403]: Starting check disk ...
Thread 4 [tid: 2212400]: Starting check disk ...
Thread 5 [tid: 2212401]: Starting check disk ...
Thread 6 [tid: 2212402]: Starting check disk ...
Thread 9 [tid: 2212405]: Starting check disk ...
Thread 1 [tid: 2212397]: Starting check disk ...
Thread 3 [tid: 2212399]: Starting check disk ...
Thread 2 [tid: 2212398]: Starting check disk ...

Current Time: 2023-06-19 15:43:59
BUG 006: [Thread: 5] VERIFY DIFFER FLAGS | read[248], write[709] | filename: /dev/loop4
PREV LBA: 0x66400 - NEXT LBA: 0xc2c00 flags is different | read[248]
LBA: 0xc2c00 | thread_num: 10, bitmap: 0 0 0 0 0 1 0 0 0 0

PREV LBA: 0x66400 | HF: 0x04001275 F1: 0x7b9930dd F2: 0x23b74b96 [LOOP: 3 WRITE: 247 SECTOR: 0 / NUM: 1024 / ZERO: 139]
PREV LBA IO Time: 2023-06-19 15:43:11

Buffer addr: 0xfffffae70000 | Physical addr: 0x5025e7d0000

LBA LIST:
LBA[ 0]: 0x00066400 0x000c2c00 0x000fa000 0x00192000 0x0034c800 0x0021c000 0x00408000 0x004cf000 0x006d8000 0x00116c00
LBA[10]: 0x00470400 0x00664c00 0x001bd800 0x001bcd00 0x00760800 0x007ff400 0x00159800 0x006b6800 0x006f0800 0x0050c400
LBA[20]: 0x006e9c00 0x00339000 0x00646800 0x003af400 0x0038ac00 0x007b6800 0x00429000 0x00008c00 0x0076b400 0x000a0cc00
LBA[30]: 0x0059b000 0x0007e000 0x001f5000 0x0059d400 0x005e8400 0x004de800 0x007e3000 0x006c9800 0x002a5400 0x0013cc00
LBA[40]: 0x00299800 0x00216000 0x00108400 0x0079d800 0x00257800 0x00505000 0x00595c00 0x006f8800 0x00308800 0x006e3000
LBA[50]: 0x005fa400 0x00326800 0x00618800 0x00005c00 0x006aac00 0x00100c00 0x0032a000 0x0029c400 0x004d1400 0x006f1000
LBA[60]: 0x0066a400 0x00183400 0x005b4400 0x00140000 0x0004e000 0x004c3800 0x00519400 0x00794400 0x003b7c00 0x002d6c00
LBA[70]: 0x0038e800 0x0017d800 0x002e8000 0x00726c00 0x006d7400 0x001c7800 0x00278800 0x003c5c00 0x00577000 0x0073c800
LBA[80]: 0x00764800 0x0009f800 0x00230400 0x005ab800 0x006fd800 0x00521000 0x004f5800 0x003df400 0x00506c00 0x00307800
LBA[90]: 0x004d8800 0x001dd000 0x00679c00 0x00754800 0x003e9800 0x002d9800 0x00088400 0x001d9000 0x001l9400 0x0046a400

PRE LBA LIST:
LBA[ 0]: 0x00117800 0x0072f800 0x005c2400 0x0045a000 0x003ca000 0x0014d000 0x006c5c00 0x000d7000 0x0058e400 0x00474400

NEXT LBA: 0xc2c00 | HF: 0xe8e8e8e8 F1: 0xe8e8e8e8 F2: 0xe8e8e8e8 [LOOP: 3907578088 WRITE: 3907578088 SECTOR: 59624 / NUM: 59624 / ZERO: 3907578088]
NEXT LBA IO Time: -387387308--387389207--387389208 -387389208:-387389208:-387389208

Buffer addr: 0xfffffa24b0000 | Physical addr: 0x58382720000

LBA LIST:
LBA[ 0]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[10]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[20]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[30]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[40]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[50]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[60]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[70]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[80]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8
LBA[90]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8

PRE LBA LIST:
LBA[ 0]: 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8 0xe8e8e8e8

Current Time: 2023-06-19 15:43:59
pause_and_exit_work: Press KEY: <Q> to exit!
```

# 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

BUG 007[1] LAST IO?: [Thread: 9] VERIFY SECTOR[271] DIFFER: 1 | read[241] | filename: /dev/loop4

hd\_write\_verify\_dump -c -D -L 0x000c4000 disk/file  
hd\_write\_verify\_dump -c -D -L 0x006ef000 disk/file

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  
INCONSISTENT IO Time: -690561470--690563369--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

Current Time: 2023-06-19 15:43:59  
BUG 007[1]: [Thread: 9] VERIFY SECTOR[271] DIFFER: 1 | read[241] | filename: /dev/loop4  
LBA: 0x407000 | thread\_num: 10, bitmap: 0 0 0 0 0 0 0 0 0 1  
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  
Buffer addr: 0xfffffa22d0000 | Physical addr: 0x261a50000  
LBA LIST:  
LBA[ 0]: 0x000407000 0x006ef000 0x007cc800 0x00095400 0x00215400 0x004ef000 0x0049c400 0x00322000 0x003c5000 0x0020a000  
LBA[10]: 0x0029b400 0x00461400 0x0056b400 0x003dc00 0x00109800 0x00118000 0x00103800 0x004bf000 0x005e3800 0x0012f000  
LBA[20]: 0x00495000 0x0034c400 0x001a5c00 0x00497c00 0x00263400 0x000d7400 0x00062800 0x0006d800 0x001ca800 0x00315000  
LBA[30]: 0x0000dfc00 0x0004f800 0x001f0800 0x00130000 0x003c0400 0x002bbc00 0x0023a800 0x006e9400 0x0007a400 0x007c0c00  
LBA[40]: 0x006d6800 0x00707000 0x004c1c00 0x00611000 0x001f0c00 0x000a6800 0x007e7400 0x002ab400 0x0023c400 0x007fa800  
LBA[50]: 0x00132000 0x005fc800 0x0077e400 0x000a2400 0x0039c800 0x001c9000 0x003d8400 0x00520000 0x00397c00 0x00235800  
LBA[60]: 0x0029e800 0x004b9000 0x00219800 0x00073000 0x006a8800 0x00248c00 0x00420000 0x005a0c00 0x007bc000 0x0037a800  
LBA[70]: 0x00293c00 0x0037b700 0x00006400 0x0026e000 0x003e4800 0x003a7800 0x00631000 0x00472c00 0x001d1800 0x00027400  
LBA[80]: 0x00294400 0x0035b500 0x004db800 0x003b1800 0x003b9000 0x00202c00 0x007fbcc0 0x00673800 0x001d8c00 0x00544c00  
LBA[90]: 0x00062c00 0x000fd800 0x002ed800 0x0001e800 0x003cb800 0x001a0800 0x00292c00 0x00718c00 0x007fec00 0x00640c00  
PRE LBA LIST:  
LBA[ 0]: 0x0000c4000 0x00702800 0x005c5800 0x00381400 0x00500800 0x0043ac00 0x0001d400 0x001d2c00 0x001c0800 0x00663c00  
-----  
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  
Buffer addr: 0xfffffa22f1e00 | Physical addr: 0x261a71e00  
LBA LIST:  
LBA[ 0]: 0xd6d6d6d6  
LBA[10]: 0xd6d6d6d6  
LBA[20]: 0xd6d6d6d6  
LBA[30]: 0xd6d6d6d6  
LBA[40]: 0xd6d6d6d6  
LBA[50]: 0xd6d6d6d6  
LBA[60]: 0xd6d6d6d6  
LBA[70]: 0xd6d6d6d6  
LBA[80]: 0xd6d6d6d6  
LBA[90]: 0xd6d6d6d6  
PRE LBA LIST:  
LBA[ 0]: 0xd6d6d6d6  
-----  
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  
Buffer addr: 0xfffffa22f2000 | Physical addr: 0x261a72000  
LBA LIST:  
LBA[ 0]: 0x000407000 0x006ef000 0x007cc800 0x00095400 0x00215400 0x004ef000 0x0049c400 0x00322000 0x003c5000 0x0020a000  
LBA[10]: 0x0029b400 0x00461400 0x0056b400 0x003dc00 0x00109800 0x00118000 0x00103800 0x004bf000 0x005e3800 0x0012f000  
LBA[20]: 0x00495000 0x0034c400 0x001a5c00 0x00497c00 0x00263400 0x000d7400 0x00062800 0x0006d800 0x001ca800 0x00315000  
LBA[30]: 0x0000dfc00 0x0004f800 0x001f0800 0x00130000 0x003c0400 0x002bbc00 0x0023a800 0x006e9400 0x0007a400 0x007c0c00  
LBA[40]: 0x006d6800 0x00707000 0x004c1c00 0x00611000 0x001f0c00 0x000a6800 0x007e7400 0x002ab400 0x0023c400 0x007fa800  
LBA[50]: 0x00132000 0x005fc800 0x0077e400 0x000a2400 0x0039c800 0x001c9000 0x003d8400 0x00520000 0x00397c00 0x00235800  
LBA[60]: 0x0029e800 0x004b9000 0x00219800 0x00073000 0x006a8800 0x00248c00 0x00420000 0x005a0c00 0x007bc000 0x0037a800  
LBA[70]: 0x00293c00 0x0037b700 0x00006400 0x0026e000 0x003e4800 0x003a7800 0x00631000 0x00472c00 0x001d1800 0x00027400  
LBA[80]: 0x00294400 0x0035b500 0x004db800 0x003b1800 0x003b9000 0x00202c00 0x007fbcc0 0x00673800 0x001d8c00 0x00544c00  
LBA[90]: 0x00062c00 0x000fd800 0x002ed800 0x0001e800 0x003cb800 0x001a0800 0x00292c00 0x00718c00 0x007fec00 0x00640c00  
PRE LBA LIST:  
LBA[ 0]: 0x0000c4000 0x00702800 0x005c5800 0x00381400 0x00500800 0x0043ac00 0x0001d400 0x001d2c00 0x001c0800 0x00663c00  
-----  
Current Time: 2023-06-19 15:43:59  
pause\_and\_exit\_work: Press KEY: <Q> to exit!

# 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 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: 0x5345cfle00

LBA LIST:  
LBA[ 0]: 0x00762800 0x006dec00 0x0069f800 0x00138800 0x006fc800 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00  
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000  
LBA[20]: 0x006df400 0x007df800 0x00117800 0x0031c000 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000  
LBA[30]: 0x00503c00 0x00449c800 0x00131c00 0x00560800 0x003t6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800  
LBA[40]: 0x00187000 0x006f7400 0x002dc000 0x0003t7800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400  
LBA[50]: 0x0065bc00 0x00449c00 0x0018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400  
LBA[60]: 0x00182400 0x004b4800 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 0x002c400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0066e400 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 0x006fc800 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00  
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000  
LBA[20]: 0x0050371fb4 0x007df800 0x00117800 0x0031c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000  
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800  
LBA[40]: 0x00107000 0x006f7400 0x002dc000 0x0003t7800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400  
LBA[50]: 0x0065bc00 0x00449c00 0x0018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400  
LBA[60]: 0x00182400 0x004b4800 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 0x002c400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0066e400 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 0x006fc800 0x007a9800 0x00449800 0x0031dc00 0x000d4800 0x00190c00  
LBA[10]: 0x0021ac00 0x0030d400 0x00714400 0x005b0c00 0x006f2400 0x002c8800 0x00441c00 0x005d1400 0x006f6400 0x00256000  
LBA[20]: 0x006df400 0x007df800 0x00117800 0x0031c00 0x00249400 0x005f0800 0x001cac00 0x00799400 0x0003d400 0x000e9000  
LBA[30]: 0x00503c00 0x004cc800 0x001a1c00 0x00560800 0x003f6800 0x00125c00 0x00728400 0x00447c00 0x003ca000 0x00073800  
LBA[40]: 0x00107000 0x006f7400 0x002dc000 0x0003t7800 0x003ca400 0x005d9800 0x0040c000 0x0008a000 0x000d9800 0x001a2400  
LBA[50]: 0x0065bc00 0x00449c00 0x0018c00 0x00724000 0x002fc400 0x0068a000 0x001ef800 0x0067f400 0x00216000 0x006f9400  
LBA[60]: 0x00182400 0x004b4800 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 0x002c400 0x004ca400 0x00509800 0x00191000 0x006f8400 0x0066e400 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

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: 0xffffeb14d0000 | Physical addr: 0x53a2320000

---runtime校验

# 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

```
BUG 007[2] LAST IO?: [Thread: 8] VERIFY ZERO SECTOR[853] DIFFER: 1 | read[259] | filename: /dev/loop4
hd_write_verify_dump -c -D -L 0x00349000 disk/file
hd_write_verify_dump -c -D -L 0x00549c00 disk/file

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

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

=====
LBA: 0x793354 | 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: 0xfffffbcbfa800 | Physical addr: 0x1002238a800

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

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

=====
LBA: 0x793355 | HF: 0xb5b5b5b5 F1: 0xb5b5b5b5 F2: 0xb5b5b5b5 [LOOP: 3048584629 WRITE: 3048584629 SECTOR: 46517 / NUM: 46517 / ZERO: 3048584629]
IO Time: -1246380767--1246382666--1246382667 -1246382667:-1246382667

Buffer addr: 0xfffffbcbfaa00 | Physical addr: 0x1002238aa00

LBA LIST:
LBA[ 0]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[10]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[20]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[30]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[40]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[50]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[60]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[70]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[80]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5
LBA[90]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5

PRE LBA LIST:
LBA[ 0]: 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5 0xb5b5b5b5

=====
LBA: 0x793356 | 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: 0xfffffbcbfac00 | Physical addr: 0x1002238ac00

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

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

=====
```

Current Time: 2023-06-19 15:59:16

# 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

BUG 007[2] LAST IO?: [Thread: 4] VERIFY ZERO SECTOR[930] DIFFER: 1 | read[79] | filename: lba.raw

hd\_write\_verify\_dump -c -D -L 0x00d8b000 disk/file  
hd\_write\_verify\_dump -c -D -L 0x00538000 disk/file

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

LBA: 0x52d7a1 | 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: 0xfffffa82f4200 | Physical addr: 0x78165c54200

LBA LIST:

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

PRE LBA LIST:

LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

LBA: 0x52d7a2 | 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: 0xfffffa82f4400 | Physical addr: 0x78165c54400

LBA LIST:

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

PRE LBA LIST:

LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

LBA: 0x52d7a3 | 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: 0xfffffa82f4600 | Physical addr: 0x78165c54600

LBA LIST:

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

PRE LBA LIST:

LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000

Current Time: 2023-06-23 11:13:38

# 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 0x00065000 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 0x00030200
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]: 0x0052e000 0x001e1000 0x006dd000 0x00539800 0x00422c00 0x00696c00 0x006a0400 0x00542400 0x002b4400 0x005be000
LBA[60]: 0x00289c00 0x00065000 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  
  
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: 1900-01-01 00:00:00  
  
Buffer addr: 0xfffff55489200 | Physical addr: 0x223be19200  
  
LBA LIST:  
LBA[ 0]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[10]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[20]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[30]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[40]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[50]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[60]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[70]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[80]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
LBA[90]: 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000  
  
PRE LBA LIST:  
LBA[ 0]: 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  
  
Buffer addr: 0xfffff55489400 | Physical addr: 0x223be19400  
  
LBA LIST:  
LBA[ 0]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[10]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[20]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[30]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[40]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[50]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[60]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[70]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[80]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
LBA[90]: 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4 0xb4b4b4b4  
  
PRE LBA LIST:  
LBA[ 0]: 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: 0xfffff552d0000 | Physical addr: 0x21a2df0000  
  
LBA LIST:  
LBA[ 0]: 0x0006b000 0x003ee400 0x007af400 0x0019d400 0x00648400 0x004ffcc0 0x0021c800 0x002dbcc0 0x005c8800 0x00155000  
LBA[10]: 0x00247c00 0x00639c00 0x00041800 0x000bd800 0x0054fc00 0x000b2000 0x00717400 0x00697800 0x00560000 0x00343000  
LBA[20]: 0x00621400 0x003e8400 0x00010c00 0x00066e00 0x0037ec00 0x00317400 0x00769000 0x00738400 0x000d3400  
LBA[30]: 0x005d4000 0x00444800 0x00051c00 0x005c4400 0x007b1800 0x004de400 0x004ab400 0x007c5400 0x007a5800 0x003b3c00  
LBA[40]: 0x000e4000 0x0075fc00 0x0026f800 0x00056800 0x001a9400 0x0028ac00 0x00252400 0x00422800 0x002a3000 0x0016bc00  
LBA[50]: 0x006f9400 0x00048c00 0x004a7000 0x000ba00 0x005ed000 0x00772800 0x00793c00 0x00415400 0x006b8800 0x00095c00  
LBA[60]: 0x004ef000 0x006d400 0x000220c0 0x0012800 0x000bac00 0x0033b000 0x00210400 0x0028cc00 0x000af400 0x006e8800  
LBA[70]: 0x0051d800 0x00173800 0x0049dc00 0x00735800 0x002ba00 0x00178000 0x001bd400 0x000d3800 0x00505c00  
LBA[80]: 0x00133400 0x00555a00 0x0009a000 0x0022f400 0x0001dc400 0x0006c0400 0x00485800 0x00088400 0x007bec00 0x0053ce00  
LBA[90]: 0x00066c00 0x004b9800 0x003f6000 0x00295800 0x00655400 0x0028bc00 0x00539400 0x000d4800 0x003cf800  
  
PRE LBA LIST:  
LBA[ 0]: 0x007e9bca 0x006e5c00 0x005de400 0x006ad400 0x0019800 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

```
[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: 0xalb5ec90 F2: 0x27c3ef0a [L0OP: 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: 0xalb5ec90 F2: 0x27c3ef0a [L0OP: 1 WRITE: 85 SECTOR: 0 / NUM: 1024 / ZERO: 246]
ERROR LBA IO Time: 2023-06-12 17:12:00
```

```
LBA: 0x11de000 | HF: 0x04001272 F1: 0xalb5ec90 F2: 0x27c3ef0a [L0OP: 1 WRITE: 84 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:11:59

Buffer addr: 0xfffff92050000 | Physical addr: 0x7024d560000

LBA LIST:
LBA[ 0]: 0x011de000 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000 0x00fb5000 0x011c000 0x00d49000 0x00898000
LBA[10]: 0x00bb6c00 0x001cc000 0x00f46400 0x00013c00 0x00192800 0x00e60400 0x0015a800 0x0094c000 0x00db6000 0x00f92c00
LBA[20]: 0x003ec000 0x0081d000 0x000ca000 0x00222000 0x008d7800 0x010f4800 0x002a8000 0x00ca1000 0x011c4000 0x000cb000
LBA[30]: 0x007e8000 0x001f4000 0x00872400 0x010c8000 0x008c6800 0x011e5000 0x00664000 0x000e34c00 0x00ab5800 0x00768400
LBA[40]: 0x00d77000 0x00901c00 0x00e16000 0x00499000 0x00a80000 0x0007e800 0x00c2a000 0x00217800 0x0074f800 0x00a67800
LBA[50]: 0x00847000 0x012a0400 0x006fb800 0x004e1400 0x004e1000 0x0055cc00 0x00976000 0x00118000 0x0105c00 0x00370000
LBA[60]: 0x001b1000 0x00f40000 0x000e8c00 0x00f2a000 0x00638000 0x00780000 0x00fb6400 0x008c1000 0x00d08800 0x002a3800
LBA[70]: 0x011af000 0x00a62000 0x00cf4400 0x005b4000 0x013ed800 0x01195800 0x008c3400 0x00509c00 0x00513800 0x00e16400
LBA[80]: 0x007c9400 0x00462000 0x00088800 0x0095c000 0x00d4b000 0x0008e000 0x00a9d000 0x01038000 0x00b50000 0x00e16400
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 0x00eb8000
```

```
LBA: 0x95d000 | HF: 0x04001272 F1: 0xalb5ec90 F2: 0x27c3ef0a [L0OP: 1 WRITE: 85 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:12:00

Buffer addr: 0xfffff920f0000 | Physical addr: 0x782b6920000

LBA LIST:
LBA[ 0]: 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000 0x00fb5000 0x011c000 0x00d49000 0x00898000 0x00bb6c00
LBA[10]: 0x0001c2800 0x00013c00 0x00060400 0x0015a800 0x0094c000 0x00db6000 0x00f92c00 0x003ec000
LBA[20]: 0x0081d000 0x000ca000 0x0032a000 0x008d7800 0x010f4800 0x002a8000 0x00ca1000 0x011c4000 0x000cb000 0x007e8000
LBA[30]: 0x001f4000 0x008572400 0x010c8000 0x008c6800 0x011e5000 0x00664000 0x000e34c00 0x00ab5800 0x00768400 0x00d77000
LBA[40]: 0x0001c000 0x00e16000 0x00499000 0x00a80000 0x0007e800 0x00c2a000 0x00217800 0x0074f800 0x00a67800 0x00847000
LBA[50]: 0x012a0400 0x00f40000 0x004e1400 0x004e1000 0x0055cc00 0x00976000 0x00118000 0x0105c00 0x00370000 0x001b1000
LBA[60]: 0x000f4000 0x00e5c000 0x00f2a000 0x000838000 0x00780000 0x008c3400 0x00098800 0x00238000 0x011af000
LBA[70]: 0x00962000 0x000f4400 0x005b4000 0x013ed800 0x01195800 0x008c3400 0x000509c00 0x00513800 0x00e16400 0x00770000
LBA[80]: 0x00462000 0x00088800 0x0095c000 0x00d4b000 0x0008e000 0x00a9d000 0x01038000 0x00b50000 0x00c76000 0x00b62400
LBA[90]: 0x00b62400 0x00807800 0x00ff1000 0x011e9000 0x00c61400 0x0135bc00 0x013e2000 0x00316400 0x00c76000 0x00b62400

PRE LBA LIST:
LBA[ 0]: 0x01168000 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: 0xalb5ec90 F2: 0x27c3ef0a [L0OP: 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]: 0x00000000 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 0x0001b400 0x004a0000 0x00fc0c00 0x00a80400 0x00c22000 0x00350000 0x00f60000 0x001295000
LBA[40]: 0x007fb800 0x00291c00 0x003b7800 0x010f8000 0x00d52000 0x00bca7000 0x00534000 0x002ff000 0x006c8000 0x01la1400
LBA[50]: 0x0077b400 0x0001d000 0x012a5800 0x0046f000 0x00ee62000 0x009d3c00 0x00aa0000 0x00d3000 0x008eb000 0x00f1800
LBA[60]: 0x011fa000 0x003dc800 0x0007a000 0x00ef7f000 0x0107c400 0x01100000 0x00ff5f000 0x00cccd00 0x00c7000 0x00f31000
LBA[70]: 0x00891400 0x0040c000 0x00ba1c00 0x00eb000 0x009fa400 0x0052f000 0x01010400 0x01138000 0x0105c400 0x00e5c000
LBA[80]: 0x00417000 0x001cc400 0x000a30000 0x011de000 0x0130fc00 0x0095d000 0x0002e1400 0x00eccc800 0x010176000 0x00fb5000
LBA[90]: 0x011c0c00 0x00d49000 0x00898000 0x0001c000 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: 0xalb5ec90 F2: 0x27c3ef0a [L0OP: 1 WRITE: 30 SECTOR: 0 / NUM: 1024 / ZERO: 246]
IO Time: 2023-06-12 17:11:18
```

```
Buffer addr: 0xfffffa4a20000 | Physical addr: 0x1032c530000
```

```
LBA LIST:
LBA[ 0]: 0x01168000 0x01029000 0x00c74000 0x00ble400 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 0x009d3c00 0x00aa0000 0x00d3000 0x008eb000 0x00ac7000
LBA[30]: 0x0001f1800 0x001fa000 0x003dc800 0x00077800 0x0107c400 0x01100000 0x005f5f000 0x00cccd00 0x00a30000
LBA[40]: 0x000f31000 0x00891400 0x0040c000 0x00ba1c00 0x00eb000 0x009fa400 0x0052f000 0x01010400 0x01138000 0x0105c400
LBA[50]: 0x00e5c000 0x00417000 0x001cc400 0x00a30000 0x011de000 0x0095d000 0x002e1400 0x0130fc00 0x00ecc800 0x00176000
LBA[60]: 0x00fb5000 0x001cc400 0x000d49000 0x00898000 0x00bb6c00 0x0013c00 0x00192800 0x00f46400 0x00013c00 0x00192800
LBA[70]: 0x0015a800 0x0094c000 0x000bd6000 0x000f92c00 0x003e0000 0x0001c000 0x000ca800 0x0032a000 0x0010f4800
LBA[80]: 0x002a8000 0x00cal000 0x011c4000 0x000cb000 0x007e8000 0x000e34c00 0x000509c00 0x00513800 0x00e16400 0x008c6800 0x011e5000
LBA[90]: 0x00664000 0x00e34c00 0x00ab5800 0x000768400 0x00d77000 0x00901c00 0x00e16000 0x00499000 0x00a80000 0x0007e8000

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。

```
[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: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 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 0x00afcd00
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 0x00e5b000 0x0057e000 0x003df400 0x00f95000 0x00b21800 0x004b2400 0x00acb800 0x00ceb800 0x01057000
LBA[40]: 0x00e00000 0x00804000 0x00c4f800 0x00e20000 0x0114b800 0x00492000 0x0105b000 0x00266800 0x00122cc0 0x008e6400
LBA[50]: 0x006c5000 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: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 1 WRITE: 70 SECTOR: 0 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:11:55
```

```
Buffer addr: 0xfffffb0740000 | 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]: 0x00b0ec800 0x00e98000 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400
LBA[50]: 0x00d69000 0x00fc7800 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00b89800 0x0101b000 0x00207000 0x00430000
LBA[60]: 0x00c11000 0x00d5d000 0x0057a800 0x00bd3400 0x00753c00 0x007c400 0x00ed6800 0x00933000 0x00d77400 0x00258400
LBA[70]: 0x00b7e000 0x00b44000 0x00e28e00 0x00958800 0x004f1800 0x00128000 0x0042c000 0x00aa0c00 0x00539800
LBA[80]: 0x006c4000 0x00375000 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000
LBA[90]: 0x0085c000 0x00e5bc00 0x0025f400 0x00af5800 0x00ca3800 0x00ff1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800
```

```
PRE LBA LIST:
LBA[ 0]: 0x0012cc00 0x00266800 0x0105b000 0x00492000 0x0114b800 0x00e20000 0x00c4f800 0x00804000 0x00e00000 0x01057000
```

```
=====
```

```
[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 0x00aa5800 disk/file
hd_write_verify_dump -c -D -L 0x00dc4c00 disk/file
```

```
CORRECT LBA[0]: 0xf24000 | HF: 0x04001275 F1: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 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: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 1 WRITE: 82 SECTOR: 234 / NUM: 1024 / ZERO: 208]
INCORRECT IO Time: 2023-06-12 17:12:00
```

```
CORRECT LBA[235]: 0xf240eb | HF: 0x04001275 F1: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 1 WRITE: 82 SECTOR: 235 / NUM: 1024 / ZERO: 208]
CORRECT IO Time: 2023-06-12 17:12:00
```

```
LBA: 0xf240e9 | HF: 0x04001275 F1: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 1 WRITE: 82 SECTOR: 233 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:12:00
```

```
Buffer addr: 0xfffff9327d200 | Physical addr: 0x30325eld200
```

```
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 0x006f9800
LBA[20]: 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x0069800 0x00757000
LBA[30]: 0x00e3000 0x006fc000 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x006f9800
LBA[40]: 0x00b0ec800 0x00e98000 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400 0x006f9800
LBA[50]: 0x00d69000 0x00fc7800 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00b89800 0x0101b000 0x00207000 0x00430000 0x006f9800
LBA[60]: 0x00c11000 0x00d5d000 0x0057a800 0x00bd3400 0x00753c00 0x007c400 0x00ed6800 0x00933000 0x00d77400 0x00258400
LBA[70]: 0x00b7e000 0x00b44000 0x00e28e00 0x00958800 0x004f1800 0x00128000 0x0042c000 0x00aa0c00 0x00539800 0x006f9800
LBA[80]: 0x006c4000 0x00375000 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000
LBA[90]: 0x0085c000 0x00e5bc00 0x0025f400 0x00af5800 0x00ca3800 0x00ff1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800
```

```
PRE LBA LIST:
LBA[ 0]: 0x00a05800 0x00673000 0x00ae1000 0x00b11800 0x00d40400 0x005a4400 0x00f63000 0x0091c000 0x010c0800 0x012ab800
```

```
=====
```

```
LBA: 0xf240eb | HF: 0x04001275 F1: 0xalb5ec90 F2: 0x29d3a78e [L0OP: 1 WRITE: 82 SECTOR: 234 / NUM: 1024 / ZERO: 208]
IO Time: 2023-06-12 17:12:00
```

```
Buffer addr: 0xfffff9327d400 | Physical addr: 0x30325eld400
```

```
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 0x006f9800
LBA[20]: 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x0069800 0x00757000
LBA[30]: 0x00e3000 0x006fc000 0x0072e000 0x01220000 0x011df000 0x01105000 0x00d14400 0x007b5000 0x0068c800 0x01144000 0x0069800
LBA[40]: 0x00b0ec800 0x00e98000 0x00a43000 0x01002800 0x0115a800 0x00d0b800 0x003c2800 0x011c3400 0x00cd1800 0x01256400 0x0069800
LBA[50]: 0x00d69000 0x00fc7800 0x007a8000 0x00dc6000 0x0100f800 0x012a8400 0x00b89800 0x0101b000 0x00207000 0x00430000 0x0069800
LBA[60]: 0x00c11000 0x00d5d000 0x0057a800 0x00bd3400 0x00753c00 0x007c400 0x00ed6800 0x00933000 0x00d77400 0x00258400
LBA[70]: 0x00b7e000 0x00b44000 0x00e28e00 0x00958800 0x004f1800 0x00128000 0x0042c000 0x00aa0c00 0x00539800 0x0069800
LBA[80]: 0x006c4000 0x00375000 0x00b98400 0x00271400 0x00846000 0x002d2800 0x0033e400 0x00b48800 0x013e8000 0x00782000
LBA[90]: 0x0085c000 0x00e5bc00 0x0025f400 0x00af5800 0x00ca3800 0x00ff1c000 0x0043f000 0x010c6000 0x00cf5800 0x00c31800
```

```
=====
```

```
LBA LIST:
LBA[ 0]: 0x00a05800 0x00673000 0x00ae1000 0x00b11800 0x00d40400 0x005a4400 0x00f63000 0x0091c000 0x010c0800 0x012ab800
```

```
PRE LBA LIST:
LBA[ 0]: 0x00a05800 0x00673000 0x00ae1000 0x00b11800 0x00d40400 0x005a4400 0x00f63000 0x0091c000 0x010c0800 0x012ab800
```

# 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: 0x1b5ec90 F2: 0x29701e70 [L0OP: 1 WRITE: 103 SECTOR: 688 / NUM: 1024 / ZERO: 282]
IO Time: 2023-06-12 17:12:00
```

```
Buffer addr: 0xfffffa1926000 | Physical addr: 0x682a12b6000
```

```
LBA LIST:
LBA[ 0]: 0x00a89800 0x0133b000 0x013c0400 0x00bae800 0x00928000 0x007b1000 0x0068a000 0x00b03000 0x01131000
LBA[10]: 0x00a94800 0x0074b000 0x00b89000 0x01250c00 0x0123800 0x00fa7400 0x00e86400 0x005d0400 0x00de5800 0x00046400
LBA[20]: 0x00a1ec00 0x001db000 0x000720400 0x00ce6000 0x006f3000 0x001c5000 0x010ea800 0x00050000 0x009d2000 0x00256800
LBA[30]: 0x00ae5000 0x01078c00 0x001b4800 0x00666000 0x000d8f000 0x00788000 0x0012f800 0x0004e1800 0x0136dc00 0x008f0800
LBA[40]: 0x005d2000 0x00326000 0x01111800 0x00223800 0x013cc800 0x01375800 0x0096b400 0x00735000 0x011cb000 0x00cec000
LBA[50]: 0x00be0800 0x01118c00 0x00314000 0x002ea000 0x00876000 0x000aa000 0x0071d000 0x00583c00 0x00853000 0x010db000
LBA[60]: 0x00cc1800 0x0010c400 0x002b3800 0x00ce8800 0x00e3b000 0x00c50000 0x00444400 0x00456000 0x01211400 0x00ca4800
LBA[70]: 0x00a56400 0x00879400 0x0051ec00 0x00abbb00 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 0x00ff12000
```

```
PRE LBA LIST:
LBA[ 0]: 0x01172000 0x01206000 0x002e8000 0x00810000 0x00a86400 0x0043e000 0x00493800 0x01le4000 0x010e4000 0x00f08c00
```

```
LBA: 0xa89ab1 | HF: 0x04001270 F1: 0x1b5ec90 F2: 0x29701e70 [L0OP: 1 WRITE: 103 SECTOR: 689 / NUM: 1024 / ZERO: 282]
IO Time: 2023-06-12 17:12:00
```

```
Buffer addr: 0xfffffa1926200 | Physical addr: 0x682a12b6200
```

```
LBA LIST:
LBA[ 0]: 0x00a89800 0x0133b000 0x013c0400 0x00bae800 0x00928000 0x007b1000 0x0068a000 0x00b03000 0x01131000
LBA[10]: 0x00a94800 0x0074b000 0x00b89000 0x01250c00 0x0123800 0x00fa7400 0x00e86400 0x005d0400 0x00de5800 0x00046400
LBA[20]: 0x00a1ec00 0x001db000 0x000720400 0x00ce6000 0x006f3000 0x001c5000 0x010ea800 0x00050000 0x009d2000 0x00256800
LBA[30]: 0x00ae5000 0x01078c00 0x001b4800 0x00666000 0x000d8f000 0x00788000 0x0012f800 0x0004e1800 0x0136dc00 0x008f0800
LBA[40]: 0x005d2000 0x00326000 0x01111800 0x00223800 0x013cc800 0x01375800 0x0096b400 0x00735000 0x011cb000 0x00cec000
LBA[50]: 0x00be0800 0x01118c00 0x00314000 0x002ea000 0x00876000 0x000aa000 0x0071d000 0x00583c00 0x00853000 0x010db000
LBA[60]: 0x00cc1800 0x0010c400 0x002b3800 0x00ce8800 0x00e3b000 0x00c50000 0x00444400 0x00456000 0x01211400 0x00ca4800
LBA[70]: 0x00a56400 0x00879400 0x0051ec00 0x00abbb00 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 0x00ff12000
```

```
PRE LBA LIST:
LBA[ 0]: 0x01172000 0x01206000 0x002e8000 0x00810000 0x00a86400 0x0043e000 0x00493800 0x01le4000 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 [L0OP: 1 WRITE: 629 SECTOR: 24 / NUM: 1024 / ZERO: 4]
IO Time: 2023-07-12 15:37:04
```

```
Buffer addr: 0xfffff9d133000 | Physical addr: 0x182b4613000
```

```
LBA LIST:
LBA[ 0]: 0x00649400 0x000ec800 0x00557c00 0x005fcfc00 0x003c6800 0x0017e400 0x00285400 0x00158c00 0x0058f000 0x0018b400
LBA[10]: 0x007b000 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 0x0021ec00 0x0033e000 0x005d0400 0x003b8800 0x0072f800
LBA[50]: 0x00389800 0x00202400 0x0017ec00 0x007fe800 0x0021f400 0x0027f400 0x00795c00 0x003bf000 0x005c4c00 0x002d4400
LBA[60]: 0x00191c00 0x0049d000 0x001d6400 0x005ca400 0x00333c00 0x004df400 0x0064b00 0x005eb800 0x0076b800 0x002da400
LBA[70]: 0x000db800 0x0030a000 0x007db800 0x00212800 0x002ec400 0x000b5800 0x003f1400 0x004fd400 0x0066a000 0x007dd400
LBA[80]: 0x0064cc00 0x003d9c00 0x007de800 0x0038cc00 0x0034e000 0x001fc00 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 [L0OP: 976894522 WRITE: 976894522 SECTOR: 14906 / NUM: 149
IO Time: 976896422-976894523-976894522 976894522:976894522:976894522
```

```
Buffer addr: 0xfffff9d133200 | Physical addr: 0x182b4613200
```

```
LBA LIST:
LBA[ 0]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[10]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[20]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[30]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[40]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[50]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[60]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[70]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[80]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
LBA[90]: 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a 0x3a3a3a3a
```

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

dump.log

```
[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: 0xfffff92ad0000 | 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 0x0021lc00 0x0023f400 0x000a3800 0x006ee800 0x00409800 0x006c8c00 0x0069d800 0x0035e000 0x0070a000
  LBA[30]: 0x00092800 0x00370000 0x0034ac00 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: 0xfffff92ad0200 | 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 0x0021lc00 0x0023f400 0x000a3800 0x006ee800 0x00409800 0x006c8c00 0x0069d800 0x0035e000 0x0070a000
  LBA[30]: 0x00092800 0x00370000 0x0034ac00 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]#
```

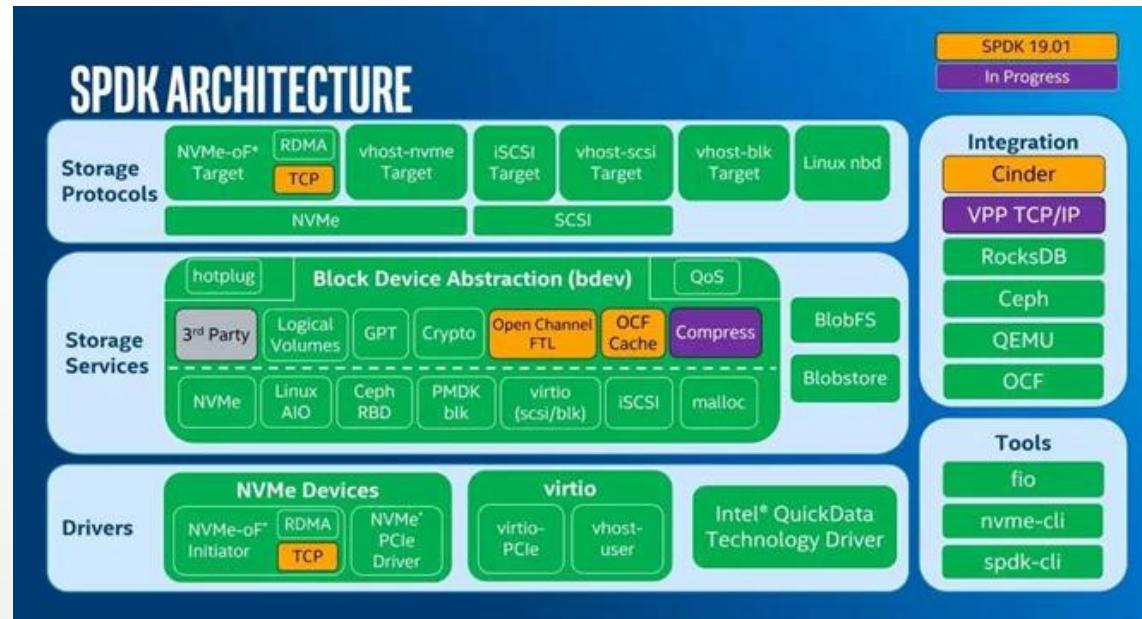
# 目 录



- 
- 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



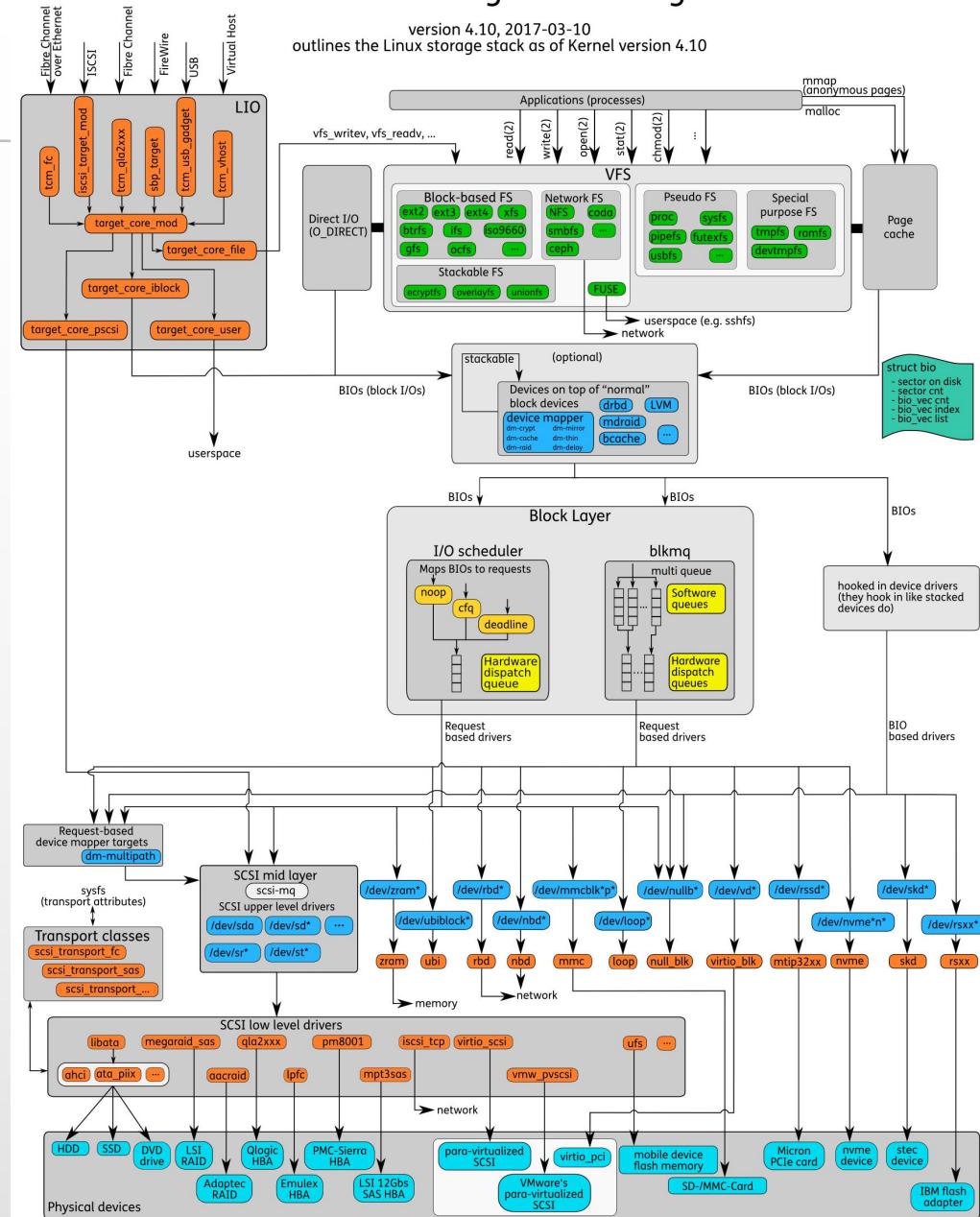
## 存储：

1. 内核存储协议栈
2. spdk
3. ceph
4. glusterfs
5. linstor
6. ipsan/fcsan/multipath/nas/.....



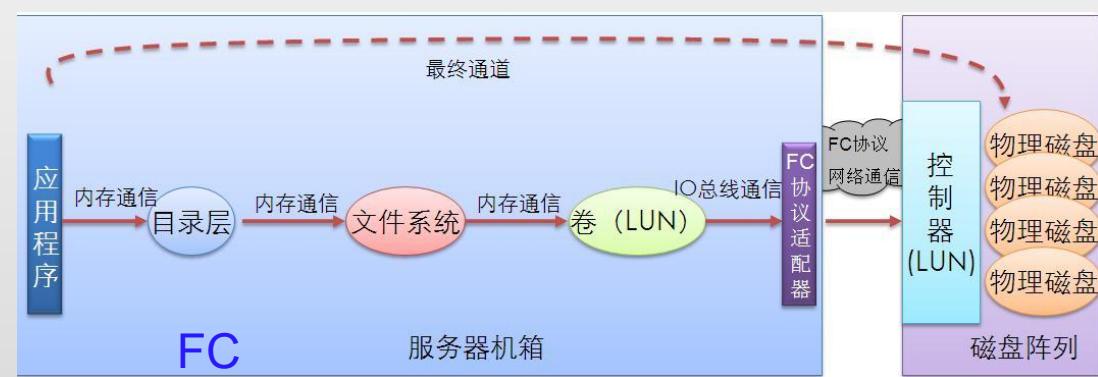
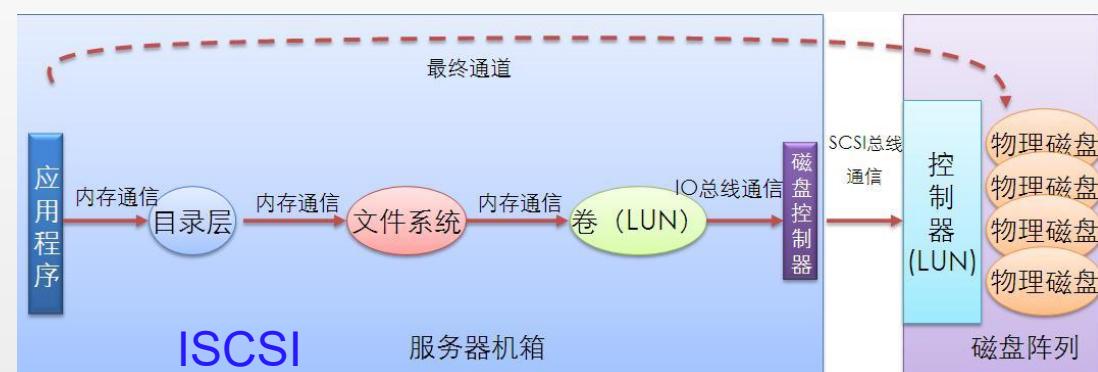
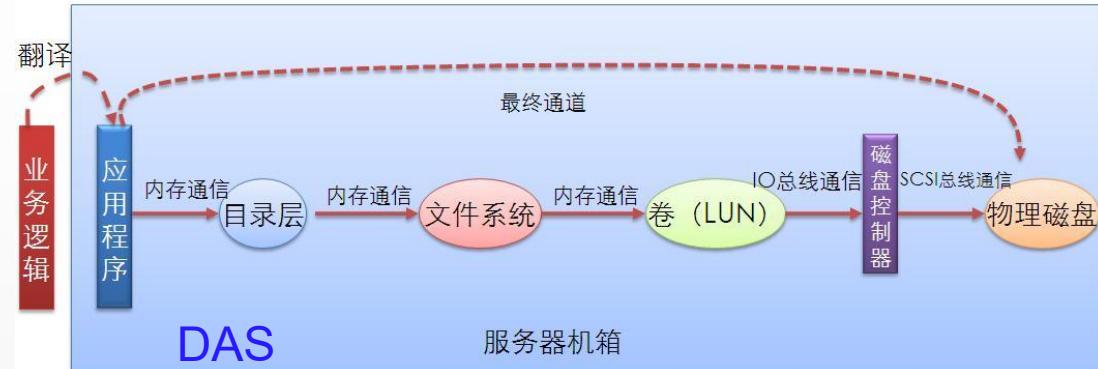
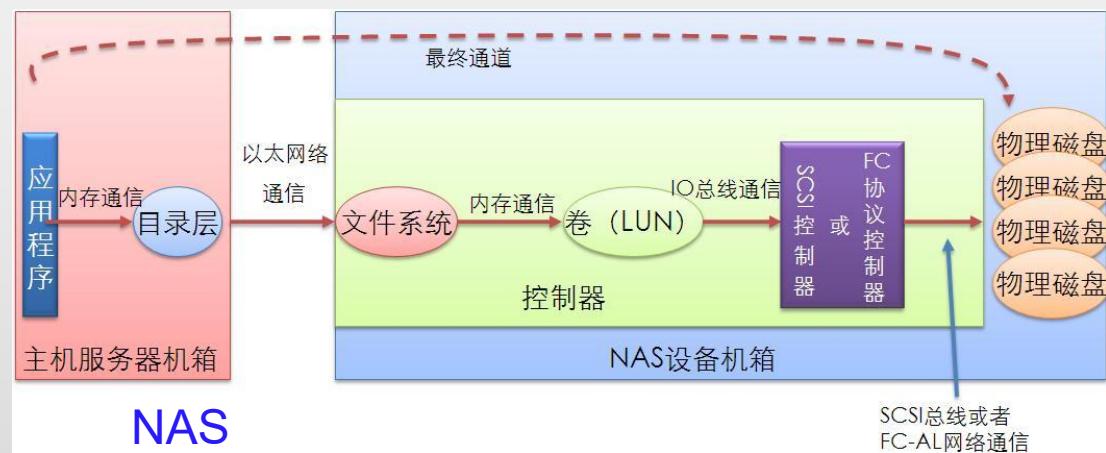
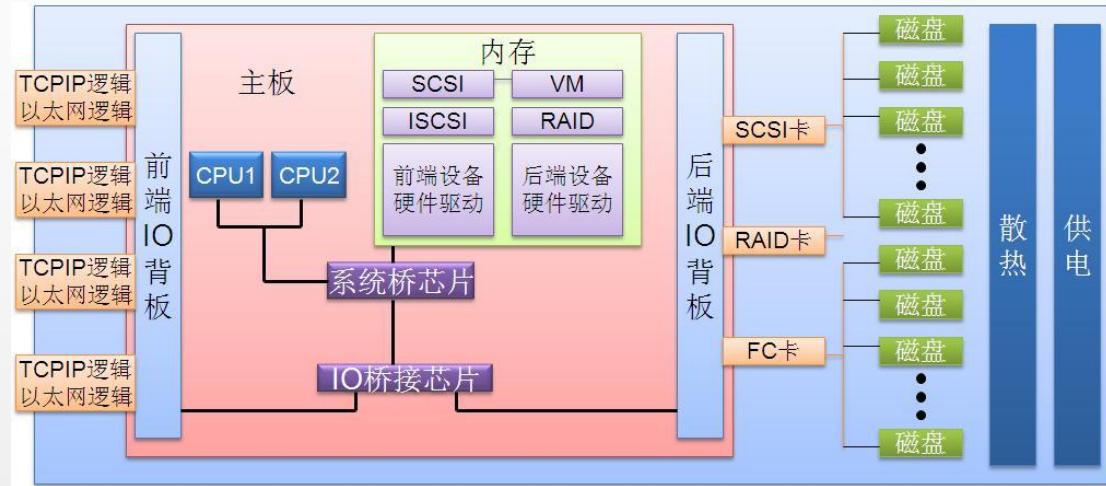
## 文件系统：

1. 本地文件系统: ext2/ext3/ext4/xfs/.....
2. 分布式文件系统: gfs2/.....
3. 内存文件系统: tmpfs/.....



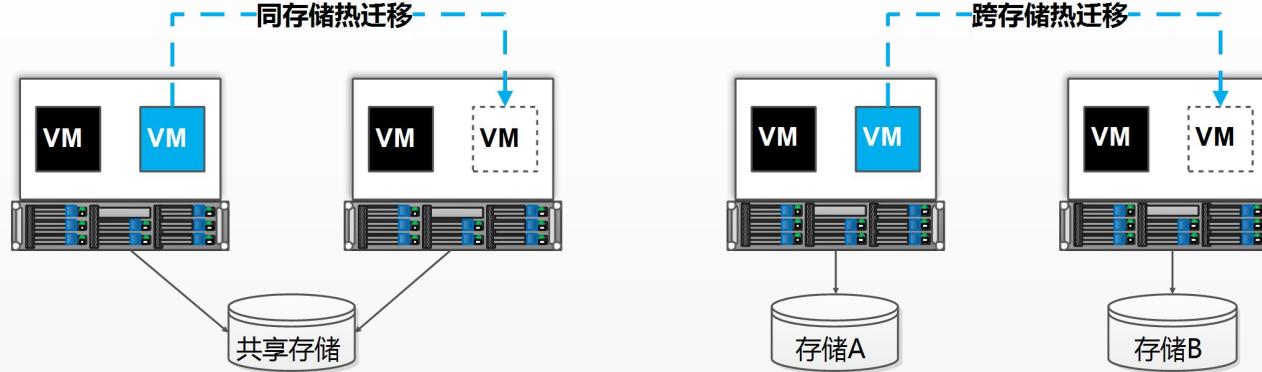
# LBA工具典型应用场景：存储、OS

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)

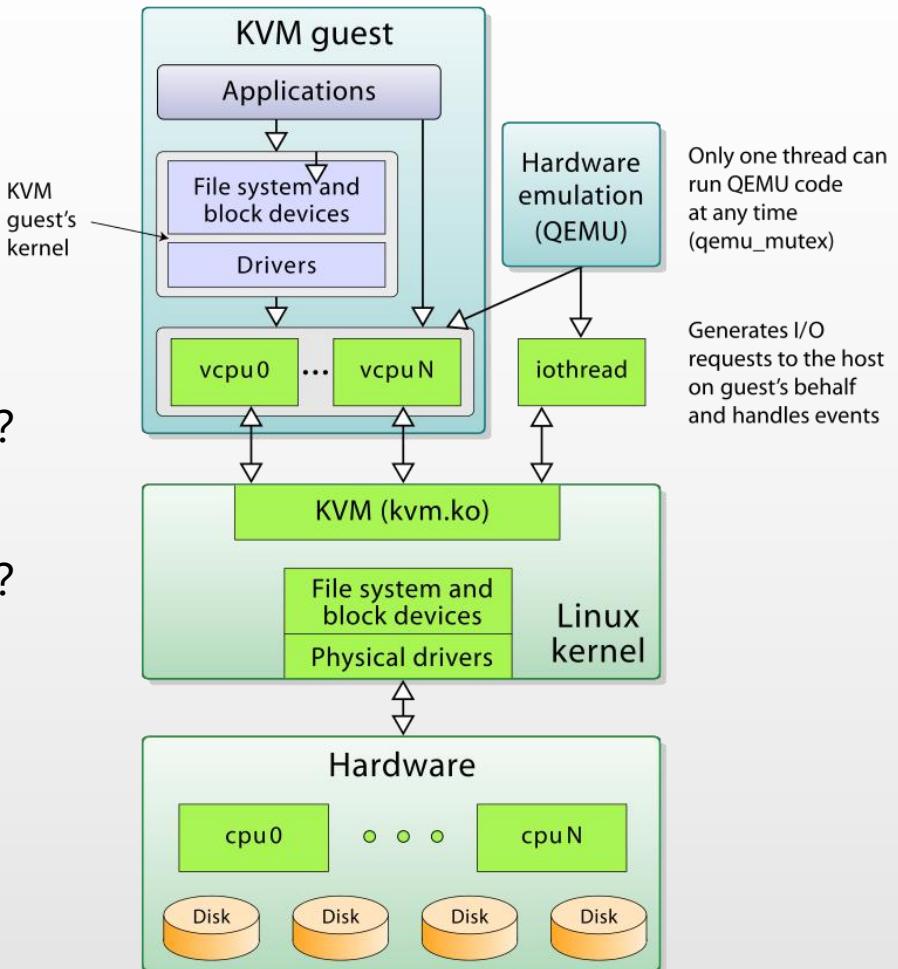
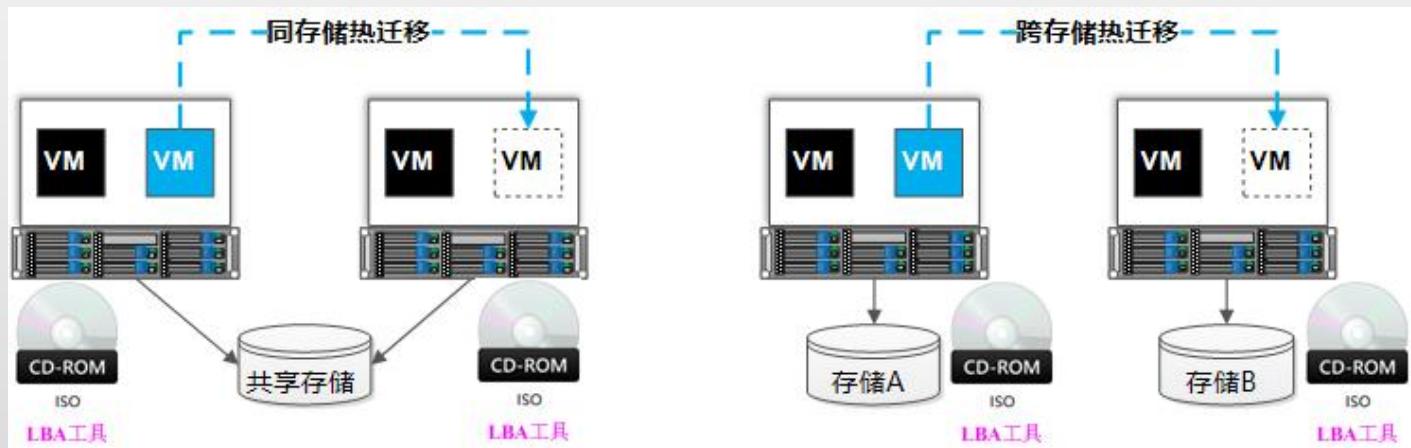


# LBA工具典型应用场景：计算

## ➤ 虚拟机热迁移存储/内存数据一致性校验



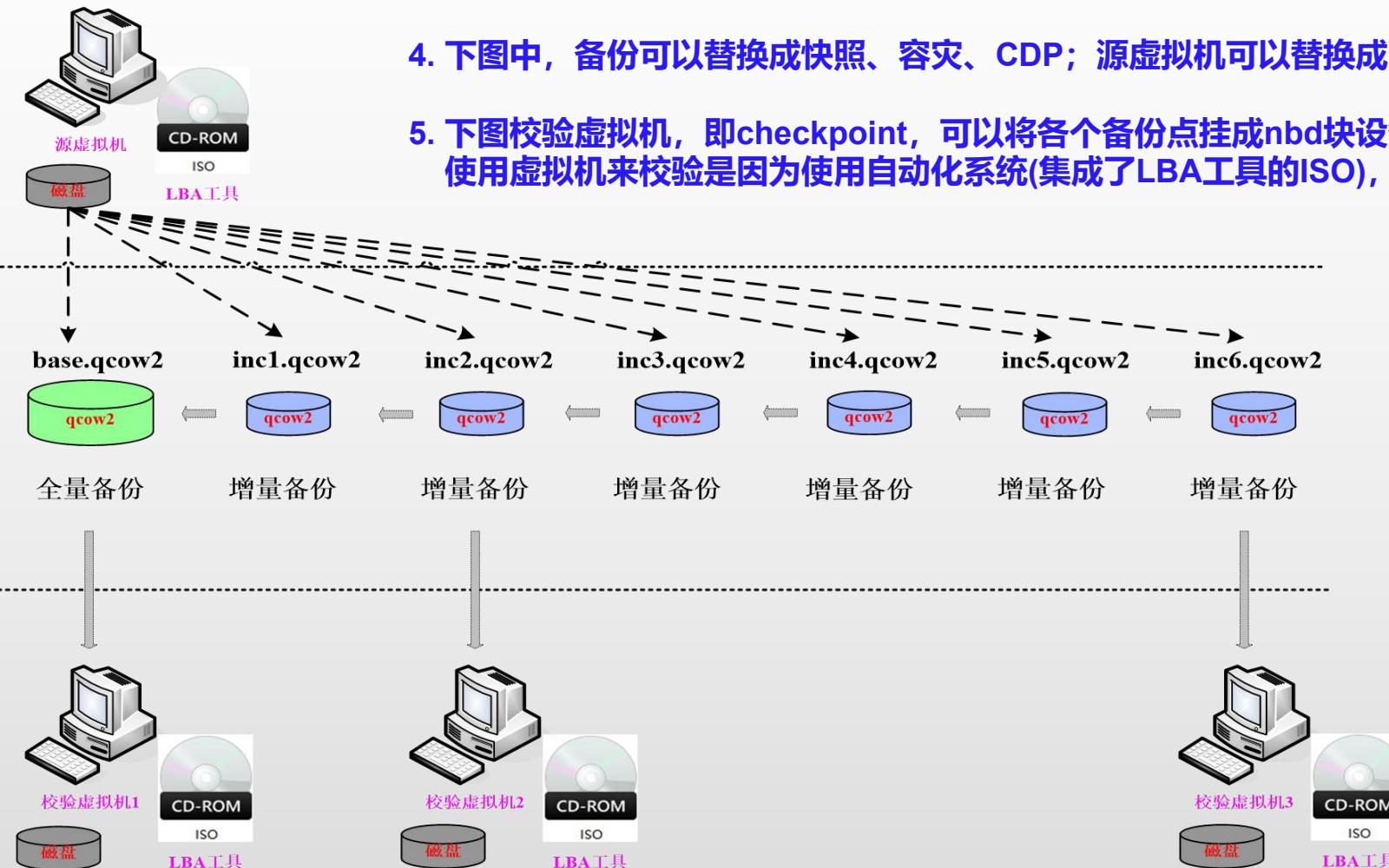
- (1) 虚拟机热迁移，怎么验证内存数据迁移到目标端虚拟机后，数据没有出错？
- (2) 虚拟机热迁移，怎么验证内存数据迁移，源端虚拟机没有漏迁数据？
  
- (3) 虚拟机热迁移，怎么验证磁盘数据迁移到目标端虚拟机后，数据没有出错？
- (4) 虚拟机热迁移，怎么验证磁盘数据迁移，源端虚拟机没有漏迁数据？



# 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；



# 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	名称	修改日期	类型	大小
hd_write_verify.with.stripe.split.64K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:22	hd_write_verify.with.stripe.split.64K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:33	ISO 文件	64,442 KB
hd_write_verify.with.stripe.split.128K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:20	hd_write_verify.with.stripe.split.128K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:33	ISO 文件	64,442 KB
hd_write_verify.with.stripe.split.256K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:18	hd_write_verify.with.stripe.split.256K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:33	ISO 文件	64,442 KB
hd_write_verify.with.stripe.split.512K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:16	hd_write_verify.with.stripe.split.512K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:32	ISO 文件	64,442 KB
hd_write_verify.with.stripe.split.1M.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:14	hd_write_verify.with.stripe.split.1M.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:32	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.4K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:12	hd_write_verify.with.stripe.robin.4K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:32	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.8K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:11	hd_write_verify.with.stripe.robin.8K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:32	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.16K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:09	hd_write_verify.with.stripe.robin.16K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:31	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.64K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:07	hd_write_verify.with.stripe.robin.64K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:31	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.128K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:05	hd_write_verify.with.stripe.robin.128K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:31	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.256K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:03	hd_write_verify.with.stripe.robin.256K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:31	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.512K.整体校验+批量校验.x86_64.V10.iso	2023/8/30 18:00	hd_write_verify.with.stripe.robin.512K.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:31	ISO 文件	64,442 KB
hd_write_verify.with.stripe.robin.1M.整体校验+批量校验.x86_64.V10.iso	2023/8/30 17:58	hd_write_verify.with.stripe.robin.1M.整体校验+批量校验.aarch64.V10.iso	2023/9/1 19:30	ISO 文件	64,442 KB
hd_write_verify.disk_lba_check_only.with.stripe.split.x86_64.V10.iso	2023/8/30 17:56	hd_write_verify.disk_lba_check_only.with.stripe.split.aarch64.V10.iso	2023/9/1 19:30	ISO 文件	64,442 KB
hd_write_verify.disk_lba_check_only.with.stripe.robin.x86_64.V10.iso	2023/8/30 17:54	hd_write_verify.disk_lba_check_only.with.stripe.robin.aarch64.V10.iso	2023/9/1 19:30	ISO 文件	64,442 KB
hd_write_verify.mem_lba.with.stripe.robin.1M.x86_64.V10.iso	2023/8/30 17:52	hd_write_verify.mem_lba.with.stripe.robin.1M.aarch64.V10.iso	2023/9/1 19:30	ISO 文件	64,442 KB
hd_write_verify.mem_disk_lba.with.stripe.robin.1M.x86_64.V10.iso	2023/8/30 17:50	hd_write_verify.mem_disk_lba.with.stripe.robin.1M.aarch64.V10.iso	2023/9/1 19:30	ISO 文件	64,442 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.V10.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)



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/
[REDACTED]

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   avgrrq-sz  avgqu-sz   await  r_await  w_await  svctm  %util
vda       0.00     0.00   38.61   41.95  39073.19  26445.41    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    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    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   avgrrq-sz  avgqu-sz   await  r_await  w_await  svctm  %util
vda       0.00     0.00   34.00   33.67  34816.00  22186.67    0.61     10.85    17.30     4.34    8.93   60.40
vdb       0.00     0.00   25.83   31.67  25941.33  19235.67    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    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   avgrrq-sz  avgqu-sz   await  r_await  w_await  svctm  %util
vda       0.00     0.00   40.33   20.00  41301.33  11946.67    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    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    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 14:06:03 GMT-8 2023 on ttys1
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

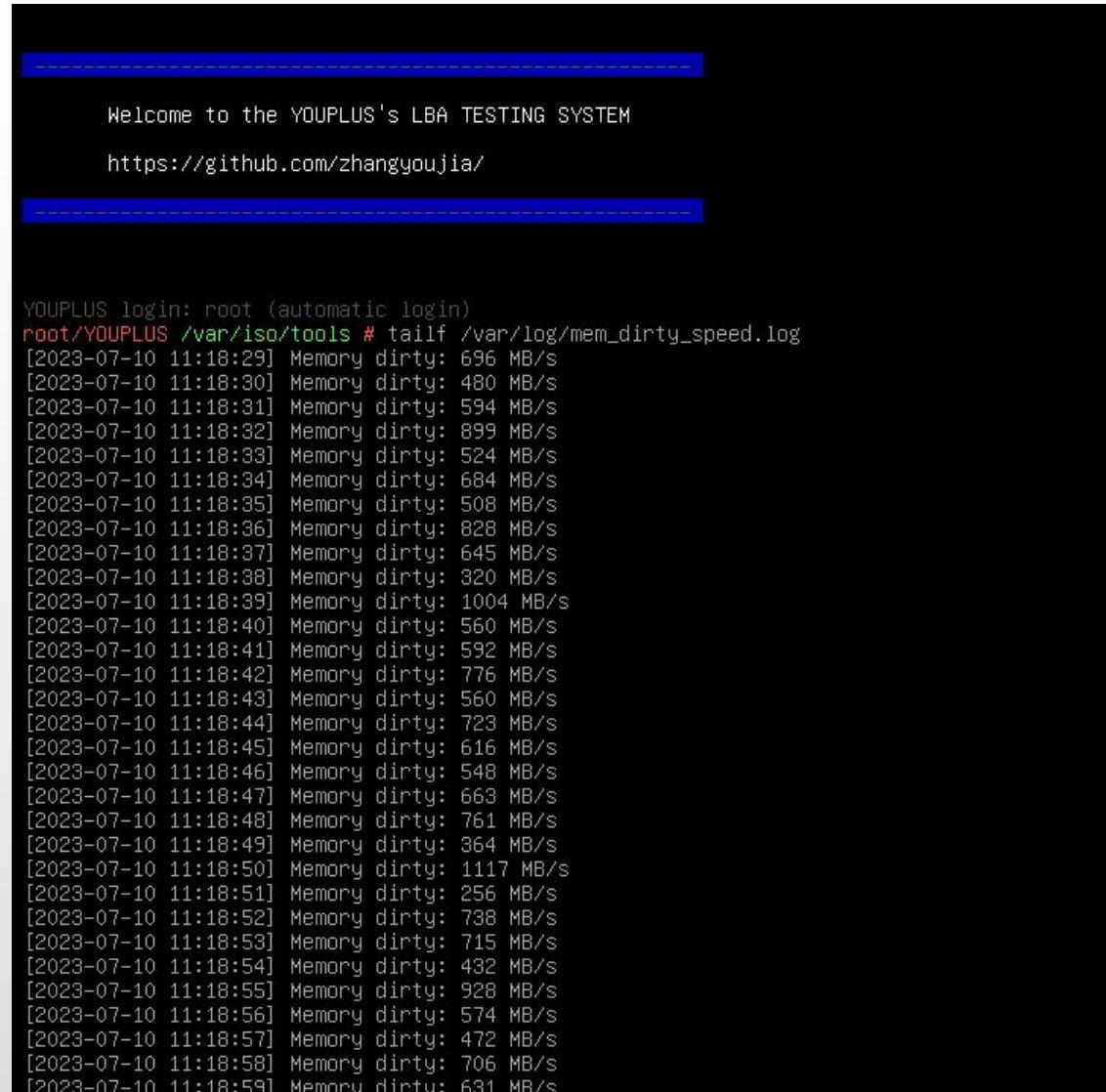
Current Time: 2023-07-10 14:06:05
Thread 5 [tid: 1122]: Starting check disk ...
Thread 6 [tid: 1123]: Starting check disk ...
Thread 1 [tid: 1118]: Starting check disk ...
Thread 9 [tid: 1126]: Starting check disk ...
Thread 4 [tid: 1121]: Starting check disk ...
Thread 2 [tid: 1119]: Starting check 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 |
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev type tmpfs (rw,relatime,size=32768k,mode=755) Current Time: 2023-07-10 14:06:39
/dev/sr0 on /var/iso type iso9660 (ro,relatime,norock,check=)
tmpfs on /var/mem type tmpfs (rw,relatime,size=3329024k) Thread 8 [tid: 1125]: Starting check disk ...
root/YOUPlus /var/iso/tools # Thread 4 [tid: 1121]: Starting check disk ...
root/YOUPlus /var/iso/tools # losetup -a Thread 9 [tid: 1126]: Starting check disk ...
/dev/loop1: [0018]:19412 (/var/mem/mem_test.raw) Thread 2 [tid: 1119]: Starting check disk ...
root/YOUPlus /var/iso/tools # 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



Welcome to the YOUPlus's LBA TESTING SYSTEM  
<https://github.com/zhangyoujia/>

```
YOUPlus login: root (automatic login)
root@YOUPlus /var/iso/tools # tailf /var/log/mem_dirty_speed.log
[2023-07-10 11:18:29] Memory dirty: 696 MB/s
[2023-07-10 11:18:30] Memory dirty: 480 MB/s
[2023-07-10 11:18:31] Memory dirty: 594 MB/s
[2023-07-10 11:18:32] Memory dirty: 899 MB/s
[2023-07-10 11:18:33] Memory dirty: 524 MB/s
[2023-07-10 11:18:34] Memory dirty: 684 MB/s
[2023-07-10 11:18:35] Memory dirty: 508 MB/s
[2023-07-10 11:18:36] Memory dirty: 828 MB/s
[2023-07-10 11:18:37] Memory dirty: 645 MB/s
[2023-07-10 11:18:38] Memory dirty: 320 MB/s
[2023-07-10 11:18:39] Memory dirty: 1004 MB/s
[2023-07-10 11:18:40] Memory dirty: 560 MB/s
[2023-07-10 11:18:41] Memory dirty: 592 MB/s
[2023-07-10 11:18:42] Memory dirty: 776 MB/s
[2023-07-10 11:18:43] Memory dirty: 560 MB/s
[2023-07-10 11:18:44] Memory dirty: 723 MB/s
[2023-07-10 11:18:45] Memory dirty: 616 MB/s
[2023-07-10 11:18:46] Memory dirty: 548 MB/s
[2023-07-10 11:18:47] Memory dirty: 663 MB/s
[2023-07-10 11:18:48] Memory dirty: 761 MB/s
[2023-07-10 11:18:49] Memory dirty: 364 MB/s
[2023-07-10 11:18:50] Memory dirty: 1117 MB/s
[2023-07-10 11:18:51] Memory dirty: 256 MB/s
[2023-07-10 11:18:52] Memory dirty: 738 MB/s
[2023-07-10 11:18:53] Memory dirty: 715 MB/s
[2023-07-10 11:18:54] Memory dirty: 432 MB/s
[2023-07-10 11:18:55] Memory dirty: 928 MB/s
[2023-07-10 11:18:56] Memory dirty: 574 MB/s
[2023-07-10 11:18:57] Memory dirty: 472 MB/s
[2023-07-10 11:18:58] Memory dirty: 706 MB/s
[2023-07-10 11:18:59] Memory dirty: 631 MB/s
```

# 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)



# 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、spdk等开源社区和国内存储(OS)、云计算、备份/容灾等厂商推荐试用；
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工具；
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)

Q & A

# Thanks