[1、问题环境 2](#_Toc889671890)

[2、问题描述 3](#_Toc1939806759)

[3、问题分析 3](#_Toc1193312712)

[3.1 sosreport-cq2t300178-20240313155241 3](#_Toc78477853)

[4、问题小结 8](#_Toc1059118651)

**平安xfs问题**

# 1、问题环境

|  |  |
| --- | --- |
| 物理机/虚拟机/云/容器 | 物理机 |
| 外网/私有网络/无网络 | 私有网络 |
| 处理器： | Hygon C86 7265 |
| 内存： | DDR4 512 GiB |
| 主板型号： | xsps |
| 整机类型/架构： | Suma H620-G30 Dhyana+ |
| BIOS版本： | Suma PQL1031220-U05 02/13/2023 |
| 固件版本： |  |
| 系统硬盘： |  |
| 网卡： |  |
| 具体操作系统版本 | V10SP2 |
| 内核版本 | 4.19.90-25.25.v2101.ky10.x86\_64 |
| 第三方模块 |  |
| 第三方应用 |  |

# 2、问题描述

日志中有大量的xfs报错堆栈。

# 3、问题分析

## 3.1 sosreport-cq2t300178-20240313155241

查看日志，存在报错：

|  |
| --- |
| Mar 8 02:46:57 cq2t300178 kernel: [283635.682471] WARNING: CPU: 26 PID: 192191 at fs/xfs/libxfs/xfs\_bmap.c:722 xfs\_bmap\_extents\_to\_btree+0x533/0x5b0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682519] CPU: 26 PID: 192191 Comm: rocksdb:low Kdump: loaded Not tainted 4.19.90-25.25.v2101.ky10.x86\_64 #1 Mar 8 02:46:57 cq2t300178 kernel: [283635.682521] Hardware name: Suma H620-G30/65N32-US, BIOS PQL1031220-U05 02/13/2023 Mar 8 02:46:57 cq2t300178 kernel: [283635.682555] RIP: 0010:xfs\_bmap\_extents\_to\_btree+0x533/0x5b0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682572] CR2: 00007f98696bc000 CR3: 0000002fe4cb0000 CR4: 00000000003406e0 Mar 8 02:46:57 cq2t300178 kernel: [283635.682573] Call Trace: Mar 8 02:46:57 cq2t300178 kernel: [283635.682614] ? xfs\_btree\_del\_cursor+0x28/0x60 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682647] ? xfs\_alloc\_ag\_vextent\_near+0x633/0xb20 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682682] xfs\_bmap\_add\_extent\_hole\_real+0x67f/0x910 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682720] xfs\_bmapi\_allocate+0x1a8/0x2f0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682755] xfs\_bmapi\_write+0x471/0x640 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682795] xfs\_alloc\_file\_space+0x153/0x2e0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682834] xfs\_file\_fallocate+0x1f5/0x370 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682874] ? xfs\_file\_read\_iter+0x6e/0xd0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.682880] ? new\_sync\_read+0xfa/0x160 Mar 8 02:46:57 cq2t300178 kernel: [283635.682884] vfs\_fallocate+0x13f/0x270 Mar 8 02:46:57 cq2t300178 kernel: [283635.682888] ksys\_fallocate+0x3c/0x70 Mar 8 02:46:57 cq2t300178 kernel: [283635.682890] \_\_x64\_sys\_fallocate+0x1a/0x20 Mar 8 02:46:57 cq2t300178 kernel: [283635.682894] do\_syscall\_64+0x5b/0x1d0 Mar 8 02:46:57 cq2t300178 kernel: [283635.682900] entry\_SYSCALL\_64\_after\_hwframe+0x44/0xa9  Mar 8 02:46:57 cq2t300178 kernel: [283635.682915] ---[ end trace 57d73c68851bd836 ]--- Mar 8 02:46:57 cq2t300178 kernel: [283635.682959] XFS (dm-5): Internal error xfs\_trans\_cancel at line 1046 of file fs/xfs/xfs\_trans.c. Caller xfs\_alloc\_file\_space+0x1ef/0x2e0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.698032] CPU: 26 PID: 192191 Comm: rocksdb:low Kdump: loaded Tainted: G W 4.19.90-25.25.v2101.ky10.x86\_64 #1 Mar 8 02:46:57 cq2t300178 kernel: [283635.698033] Hardware name: Suma H620-G30/65N32-US, BIOS PQL1031220-U05 02/13/2023 Mar 8 02:46:57 cq2t300178 kernel: [283635.698034] Call Trace: Mar 8 02:46:57 cq2t300178 kernel: [283635.698045] dump\_stack+0x66/0x8b Mar 8 02:46:57 cq2t300178 kernel: [283635.698103] xfs\_trans\_cancel+0x131/0x150 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.698138] xfs\_alloc\_file\_space+0x1ef/0x2e0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.698172] xfs\_file\_fallocate+0x1f5/0x370 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.698206] ? xfs\_file\_read\_iter+0x6e/0xd0 [xfs] Mar 8 02:46:57 cq2t300178 kernel: [283635.698210] ? new\_sync\_read+0xfa/0x160 Mar 8 02:46:57 cq2t300178 kernel: [283635.698213] vfs\_fallocate+0x13f/0x270 Mar 8 02:46:57 cq2t300178 kernel: [283635.698215] ksys\_fallocate+0x3c/0x70 Mar 8 02:46:57 cq2t300178 kernel: [283635.698218] \_\_x64\_sys\_fallocate+0x1a/0x20 Mar 8 02:46:57 cq2t300178 kernel: [283635.698221] do\_syscall\_64+0x5b/0x1d0 Mar 8 02:46:57 cq2t300178 kernel: [283635.698227] entry\_SYSCALL\_64\_after\_hwframe+0x44/0xa9 Mar 8 02:46:57 cq2t300178 kernel: [283635.698234] RIP: 0033:0x7fa019c4c455  Mar 8 02:46:57 cq2t300178 kernel: [283635.698251] XFS (dm-5): xfs\_do\_force\_shutdown(0x8) called from line 1047 of file fs/xfs/xfs\_trans.c. Return address = 00000000cc808cfe Mar 8 02:46:57 cq2t300178 kernel: [283635.719893] XFS (dm-5): Corruption of in-memory data detected. Shutting down filesystem Mar 8 02:46:57 cq2t300178 kernel: [283635.729604] XFS (dm-5): Please umount the filesystem and rectify the problem(s) Mar 8 02:47:16 cq2t300178 bash[2221186]: /bin/bash: /paic/t0ucccis/td5523/deploy/scripts/run\_tikv.sh: Input/output error |

报错发生于函数xfs\_bmap\_extents\_to\_btree，此时正将存储结构由extent格式转化为btree格式，需要分配一个块的空间来放置btree相关的数据结构。

不过在试图分配这个块时，发现文件系统的元数据存在异常，所以准备shutdown掉文件系统，以防这个错误进一步扩散。

上述中报错的栈并未指明文件系统的数据究竟是出现了什么异常。所以可以通过对磁盘用xfs\_repair -n进行扫描，找到文件系统元数据的异常。

具体首先展开metadump文件为磁盘镜像，然后用xfs\_repair扫描：

|  |
| --- |
| xfs\_metadump -o /dev/mapper/VG\_tidb-LV\_t0ucccis\_k06\_data 故障日志20240308.metadump.metadump xfs\_mdrestore 故障日志20240308.metadump dm.img xfs\_repair -n dm.img 2>&1 | tee scan\_log.txt |

扫描输出为：

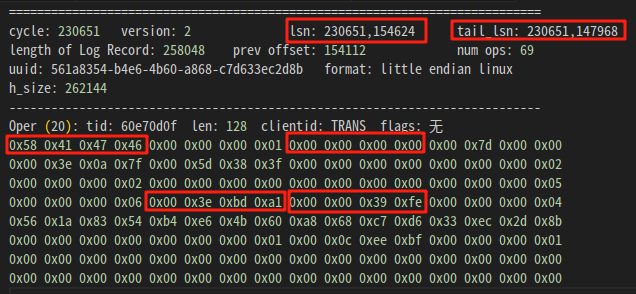
|  |
| --- |
| # **阶段 1 - 查找并验证超级块** # 检查文件系统的超级块是否完好。超级块是文件系统的关键元数据之一，包含了文件系统的整体信息。 Phase 1 - find and verify superblock...  - reporting progress in intervals of 15 minutes # **阶段 2 - 使用内部日志** # 此阶段涉及清零日志并扫描文件系统的空闲空间和inode映射。输出中的不一致表明实际计算的空闲块数和最长连续空闲块数与分配组（AG）0的超级块（agf）记录值不匹配。此外，全文件系统的可用块数也与超级块（sb）记录的不匹配。 Phase 2 - using internal log  - zero log...  - scan filesystem freespace and inode maps...   # agf\_freeblks是分配组文件（AGF）记录的空闲块数量，这里的值是3997090。 # counted的值是实际检测到的空闲块数量，为4111777。 # 这表示在ag0中，实际的空闲块数量比记录的多出114687块。 agf\_freeblks 3997090, counted 4111777 in ag 0  #agf\_longest指的是AGF记录中最长的连续空闲块序列的长度，这里的值是14556。 #counted的值是实际检测到的最长连续空闲块序列的长度，为14846。 #实际上在ag0中存在的最长连续空闲块序列比记录的要长，多出290块。 **agf\_longest 14556, counted 14846 in ag 0**:  #sb\_fdblocks是超级块（superblock）记录的全文件系统的空闲块总数，这里的值是37433544。 #counted的值是实际检测到的全文件系统空闲块总数，为37548231。 #实际的文件系统空闲块总数比记录的多出114687块。 sb\_fdblocks 37433544, counted 37548231  - 16:06:45: scanning filesystem freespace - 16 of 16 allocation groups done  - found root inode chunk # **阶段 3 - 对每个分配组（AG）**  Phase 3 - for each AG...  - scan (but don't clear) agi unlinked lists...  - 16:06:45: scanning agi unlinked lists - 16 of 16 allocation groups done  - process known inodes and perform inode discovery...  - agno = 15  - agno = 0  - agno = 1  - agno = 2  - agno = 3  - agno = 4  - agno = 5  - agno = 6  - agno = 7  - agno = 8  - agno = 9  - agno = 10  - agno = 11  - agno = 12  - agno = 13  - agno = 14  - 16:06:46: process known inodes and inode discovery - 30400 of 30400 inodes done  - process newly discovered inodes...  - 16:06:46: process newly discovered inodes - 16 of 16 allocation groups done # **阶段 4 - 检查重复块** # 建立重复的扩展列表，然后检查inode是否声称有重复的块。这一阶段没有报告具体的重复块问题，意味着没有发现inode间的块重复使用情况。 Phase 4 - check for duplicate blocks...  - setting up duplicate extent list...  - 16:06:46: setting up duplicate extent list - 16 of 16 allocation groups done  - check for inodes claiming duplicate blocks...  - agno = 0  - agno = 2  - agno = 13  - agno = 3  - agno = 5  - agno = 4  - agno = 6  - agno = 7  - agno = 8  - agno = 9  - agno = 10  - agno = 11  - agno = 12  - agno = 15  - agno = 1  - agno = 14  - 16:06:46: check for inodes claiming duplicate blocks - 30400 of 30400 inodes done No modify flag set, skipping phase 5 **阶段 5 - 被跳过** # **阶段 6 - 检查inode连通性** Phase 6 - check inode connectivity...  - traversing filesystem ...  - traversal finished ...  - moving disconnected inodes to lost+found ... # **阶段 7 - 验证链接计数** Phase 7 - verify link counts...  - 16:06:46: verify and correct link counts - 16 of 16 allocation groups done |

扫描输出的结果告诉我们，异常来自与superblock,agf中的统计字段的值与实际不符。比如：

agf\_freeblks值是3997090，实际检测到的值为4111777。

agf\_longest值是14556，实际检测到值为14846。

通过在xfs日志里面查找，我们看到实际上文件系统已经对相应的值进行了正确的更新，如下：



简短介绍下xfs日志。所有元数据写入时，会首先写入到xfs日志。

所以agf在被更新时，agf的数据会先写入到日志。

上图中16进制数据就是即将写入到磁盘上的数据，具体意思为：

0x58 0x41 0x47 0x46, 对应的是‘X’‘A’‘G’‘F’。代表agf数据结构。

0x00 0x00 0x00 0x00, 代表的是第0号AG。

0x00 0x3e 0xbd 0xa1 = 4111777，代表agf\_freeblks。

0x00 0x00 0x39 0xfe = 14846，代表agf\_longest。

注意这两个值与前文中扫描到的实际数值一致。也就是说，**正确的应当被写入到磁盘agf数据块已经在日志当中出现了。**

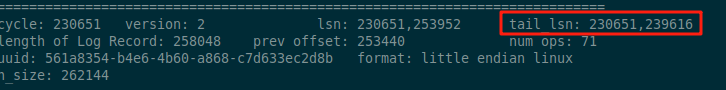
**接下来证明这一则数据已经成功发送到磁盘。**

上图中lsn为（230651，154624）。这个代表本次写入的lsn序号,可将lsn序号等价为时间点。

这里tail\_lsn为（230651，147968）。这个代表在时间点（230651，147968）以前记录对应的io均已完成。每当相应的日志记录对应buf io 完成函数完成时最终会体现到该字段的更新里。

由于（230651，154624）> （230651，147968）。所以这里并不能看出这条记录对应的io完成函数已经被调用。

可以通过之后的其它某条记录的tail\_lsn字段来判断这一点。例如：



该字段tail\_lsn为（230651，239619）。这个代表在时间点（230651，239619）以前记录对应的io均已完成。

而（230651，154624）< （230651，239619），

这说明那条关于AGF的记录对应的io已经完成返回。

随后查看磁盘中agf结构最近一次写入的时间序号lsn是：

|  |
| --- |
| xfs\_db> agf 0  xfs\_db> p  magicnum = 0x58414746  versionnum = 1  seqno = 0  length = 8192000  bnoroot = 4065919  cntroot = 6109247  rmaproot =  refcntroot = 847551  bnolevel = 2  cntlevel = 2  rmaplevel = 0  refcntlevel = 1  rmapblocks = 0  refcntblocks = 1  flfirst = 0  fllast = 5  flcount = 6  freeblks = 3997090  longest = 14556  btreeblks = 4  uuid = 561a8354-b4e6-4b60-a868-c7d633ec2d8b  lsn = 0x383dc00078a00  crc = 0xdf78498b (correct) |

lsn = 0x383dc00078a00 = （0x383dc，0x00078a00）= (230364, 494080) <（230651，154624）

即

磁盘中的agf最近一次更新发生在时刻（230364，494080）。

但是日志中记载在之后时刻（230651，154624）又发生了更新写入。只要完成这一则io，那么agf数据就是完好一致的。

系统显示该io已经完成，但是实际上磁盘上并未看到该数据。

# 4、问题小结

日志中已经记载了正确的数据，并且记载该数据已经发往硬件。

最后磁盘上未看到相应数据，所以数据在是在硬件中丢失了。

但是具体是硬件中的哪一步丢失了呢？

首先通过**关闭了阵列卡缓存后问题仍然存在**这个现象看，排除阵列卡缓存这一步造成的问题。

接下来，可以进一步改变硬件中的一些条件，看能否进一步缩小范围。比如：

可以将出问题的盘由raid5变成raid1。之所以建议为raid1，是因为目前为止出问题的都不包括raid1。