Cassandra Data Recovery

fixing broken sstables by fliping bits

Michael «hops» Sprecher

Cassandra

- NoSQL database
- SQL like Query Language (CQL)
- part of Apache Software Foundation

Hashes.org

- Founded by Sein Coray in 2012
- Public database of leaked password hashes
- Go to place for bored hashcrackers
- The reason for this talk

Story time

- Weird behaviour of Hashes.org ~3 months ago
- Random (non-reproduceable) segfaults of `sort`
- 7zip randomly failing to compress files
- Suspected recently updated libc
- Suddenly a corrupt database appears
- Repairs without lasting success

Story time

- Screw it! Take a backup and start from scratch
- Cassandra failed to load some SSTables
- What now?

Options

- Offline scrubbing – took ~2 days for a 1.1gb file

  Largest file is ~320gb – would take ~300 days
- Write our own SSTable parser

  We know the database schema

  Data is rather easy to verify for correctness
- SSTable format is well documented (sort of)

SSTable format

```
struct partition {
    struct partition_header header;
    optional<struct row> static_row; // Has IS_STATIC flag set
    struct unfiltered unfiltereds[];
};

struct partition_header {
    be16 key_length;
    byte key[key_length];
    struct deletion_time deletion_time;
};

struct deletion_time {
    be32 local_deletion_time;
    be64 marked_for_delete_at;
};
```

SSTable format

The special value LIVE = (MAX_BE32, MIN_BE64), i.e., the bytes 7F FF FF FF 80 00 00 00 00 00 00 00, is the default for live, undeleted, partitions.

https://docs.scylladb.com/architecture/sstable/sstable3/sstables_3_data_file_format

SSTable format

```
00000000: 0000 0100 f219 0020 3532 3232 3934 3038  ....... 52229408
00000010: 3137 3839 3162 6134 6162 6136 6164 3832  17891ba4aba6ad82
00000020: 6565 3964 6433 6462 7fff ffff 8000 0100  ee9dd3db........
00000030: f407 2080 8f2e 00fc 45b1 bcfa 1cc7 0956  .. .....E......V
00000040: 4255 4c4c 4554 494e 1200 0124 0003 1e00  BULLETIN...$....
00000050: 080c 0044 522c e38d 2400 44ff ffff f40c  ...DR,..$.D.....
00000060: 0008 2400 f402 1036 3236 3137 3236 3136  ..$....626172616
00000070: 3236 6636 6636 6425 0074 0637 3635 3034  26f6f6d%.t.76504
00000080: 620f 0046 0000 0008 0c00 fa1e 07df 0100  b..F............
00000090: 2834 3932 6662 6432 3733 3366 3734 6264  (492fbd2733f74bd
```

WOOT?

Compression

- Cassandra uses LZ4 (default) to save diskspace
- Data is sliced into 64kb blocks (called chunk)
- A chunk begins with chunk_length
- Each chunk is compressed using LZ4
- A CRC32 checksum of the compressed chunk is calculated and added at the end of a chunk
- Chunk offsets stored in *-CompressionInfo.db

Compressed chunk

```
00000000: 0000 0100 d222 3742 03fb d804 4e54 4c4d  ....."7B....NTLM
00000010: 00fc 0d00 445b 92b2 ce0c 0044 0000 0001  ....D[.....D....
…
00008d50: 0000 0000 fc68 acde df                   .....h…
```

Chunk length (int16 little endian)

Compressed data

CRC32 of chunk length + data (uint32 big endian)

Story time (continue)

- Identified 3 corrupt SSTables
- Meanwhile weird issues on Hashes.org server appeared again
- Running a memory test revealed a defect RAM module
- Server got replaced (start again from scratch)

CRC32

- Checksum for data
- Not collision resistent
- Good enough to detect single bitflips
- We do not know which bit is wrong

Flip all the bits!

- Loop through data in chunk
- For each byte flip on bit at a time
- After each flip check if CRC32 matches
- Rinse and repeat
- Does not work when multiple bits are wrong (unless we have a hint which bit fliped)

Progress (so far)

- Two (small-ish) SSTables repaired and imported
- 14/30 defect chunks in the big SSTable had a single bitflip. One had two bitflips and one had three
- Remaining 14 defect chunks «repaired» by simply patching the CRC32 checksum
- Sadly import of the big file was not (yet) successful :(

Questions?