Twoskip – Cyrus database format

Crash-safe, 64 bit, transactional
key-value store

brong@opera.com

Cyrus

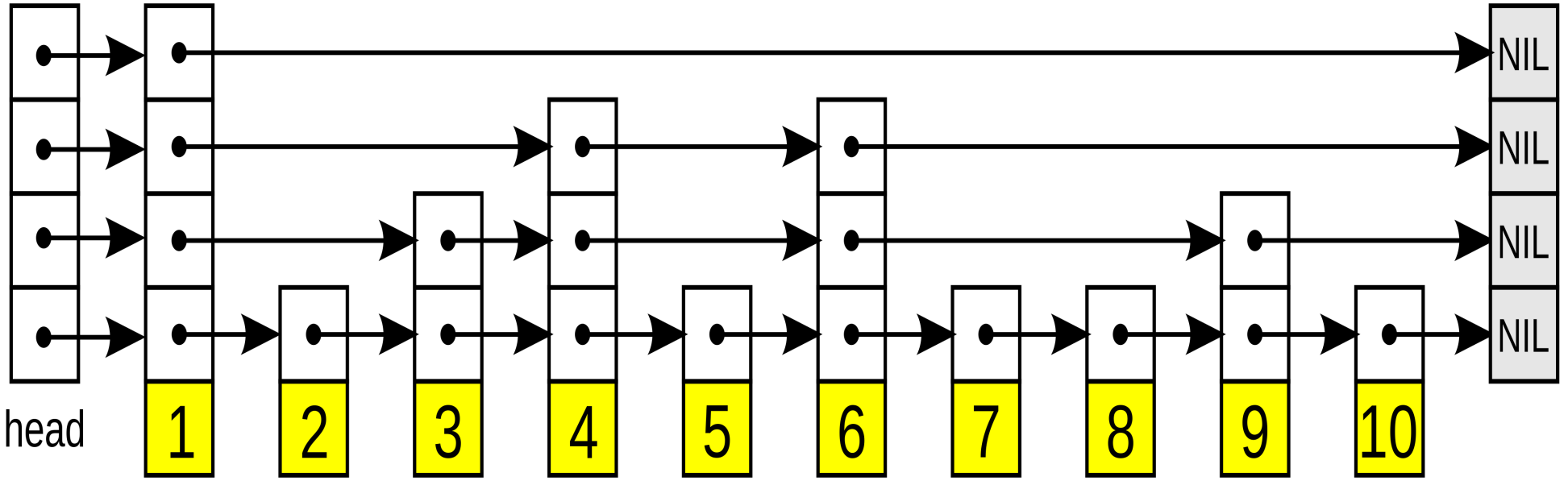
- Email storage server (IMAP/POP/LMTP)
- Old codebase, written in C.
- Data formats (custom binary)
- Databases (key/value – transactional)

Databases

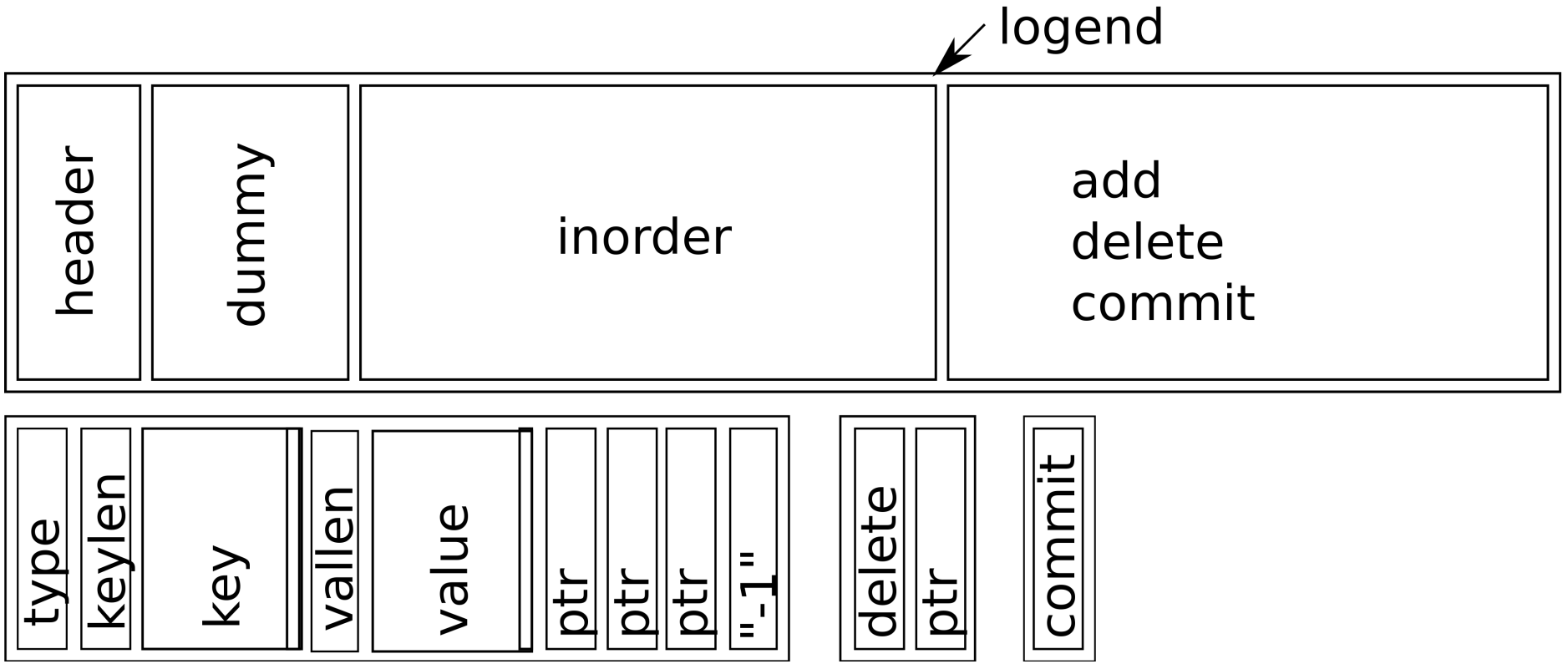
- Berkeley DB
  - Environment issues (config required)
  - Versioning/upgrade pain
  - Deadlocks possible
- Skiplist
  - Corrupty until 2.3.7(ish)
  - Otherwise, pretty good
- Crappy flat-file rubbish

Skiplist

- http://en.wikipedia.org/wiki/Skip_list



Skiplist file format



- Changes: append new record, rewrite pointers
- Repack – stream to “inorder” records, rename

Bug fixes through 2.3.x

- Locking fds, flock, re-opening DB
- Segfaults on reading corrupted DBs
- Random Linux kernel bug
- Type size problems

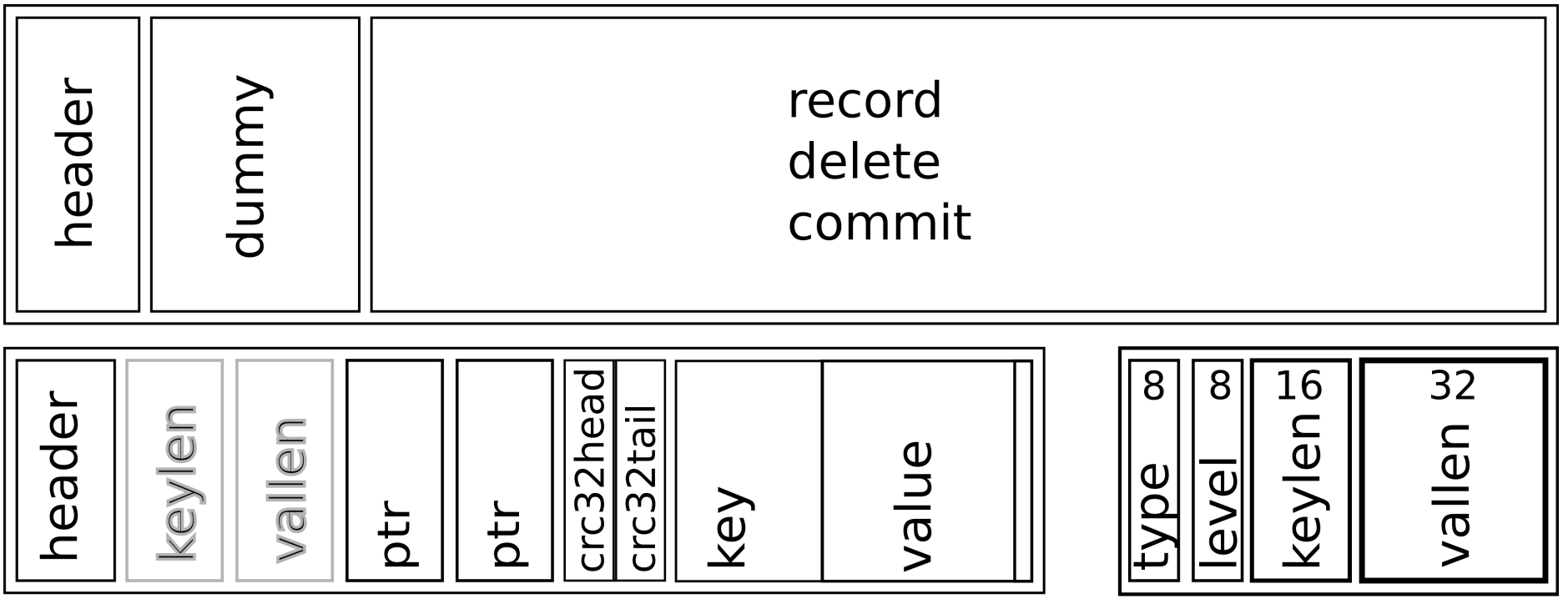
Unfixable issues

- 32 bit size limit
- “skipstamp” - need write lock to read data so you can update “last accessed”.
- Slow rebuild – can't know for sure if file is consistent after a restart
- (workaround: “skipclean” file on shutdown)
- Wrote a “skiplist2”, but it didn't solve the crash problem – you can't trust write ordering on arbitrary filesystems.

New database goals

- 64 bit throughout
- Checksums on all content
- Single on-disk file
- Fully embedded, no daemons or external helpers
- Readable without causing writes (if clean)
- May require that an aligned 512 bytes either fully writes or doesn't write anything.
- May require fsync to work

Twoskip format



Twoskip header

- “Dirty” flag
- File size at last commit (rather than logend)
- Predicted size at next repack (still have to repack to remove stale records and improve locality)
- Generation number (for efficient locality reads in the usual case)

Safe changes

- Rewrite header with “dirty” flag set
- Append data – update pointers
- Commit: rewrite header with new length and dirty off again

3 fsyncs

- Header: dirty=1, fsync
- Append, update pointers

  Append, update pointers

  ...

  fsync
- Header dirty = 0, fsync

Safe changes

- Rewrite header with “dirty” set, fsync
- Append data – update pointers
- Commit: fsync, rewrite header clean, fsync.

- **Problem: host crash during append, pointers might point past EOF – the chain is broken, can't rebuild without a full parse.**

Two complete linked lists

- Always keep a crash-safe unbroken chain of linked nodes.
- Write algorithm: update the link with the lowest offset UNLESS the higher is part of this transaction.
- Read algorithm: use the highest value which is not past the “length” in the header (works even if corrupted – but NOTE: higher level links may be bogus if the file is dirty)
- Repair algorithm – walk at level zero – wiping any bogus zero level pointers. For higher levels, keep back-pointers and update them when you find the next record of at least that height.

```
HEADER: v=1 fl=0 num=0 sz=(00000150/00000150)
00000040 DUMMY kl=0 dl=0 lvl=31 ()
    00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

```
HEADER: v=1 fl=0 num=1 sz=(00000198/00000180)
00000040 DUMMY kl=0 dl=0 lvl=31 ()
    00000000
    00000150 00000150 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000150 RECORD kl=1 dl=1 lvl=2 (a)
    00000000
    00000000 00000000
00000180 COMMIT start=00000150
```

```
HEADER: v=1 fl=0 num=2 sz=(00000238/000001C0)
00000040 DUMMY kl=0 dl=0 lvl=31 ()
    000001E0
    00000150 000001E0 000001E0 000001E0 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000150 RECORD kl=1 dl=1 lvl=2 (a)
    00000000
    00000198 00000198
00000180 COMMIT start=00000150
00000198 RECORD kl=1 dl=1 lvl=2 (b)
    00000000
    00000000 00000000
000001C8 COMMIT start=00000198
000001E0 RECORD kl=1 dl=1 lvl=4 (a)
    00000000
    00000198 00000198 00000000 00000000
00000220 COMMIT start=000001E0
```

```
HEADER: v=1 fl=0 num=3 sz=(00000288/000001F8)
00000040 DUMMY kl=0 dl=0 lvl=31 ()
    000001E0
    00000150 000001E0 000001E0 000001E0 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000150 RECORD kl=1 dl=1 lvl=2 (a)
    00000000
    00000198 00000198
00000180 COMMIT start=00000150
00000198 RECORD kl=1 dl=1 lvl=2 (b)
    00000000
    00000238 00000238
000001C8 COMMIT start=00000198
000001E0 RECORD kl=1 dl=1 lvl=4 (a)
    00000000
    00000198 00000198 00000238 00000000
00000220 COMMIT start=000001E0
00000238 RECORD kl=1 dl=1 lvl=3 (c)
    00000000
    00000000 00000000 00000000
00000270 COMMIT start=00000238
```

```
HEADER: v=1 fl=0 num=4 sz=(000002D0/00000228)
00000040 DUMMY kl=0 dl=0 lvl=31 ()
    000001E0
    00000150 000001E0 000001E0 000001E0 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000150 RECORD kl=1 dl=1 lvl=2 (a)
    00000000
    00000198 00000198
00000180 COMMIT start=00000150
00000198 RECORD kl=1 dl=1 lvl=2 (b)
    00000000
    00000238 00000238
000001C8 COMMIT start=00000198
000001E0 RECORD kl=1 dl=1 lvl=4 (a)
    00000288
    00000198 00000288 00000238 00000000
00000220 COMMIT start=000001E0
00000238 RECORD kl=1 dl=1 lvl=3 (c)
    00000000
    00000000 00000000 00000000
00000270 COMMIT start=00000238
00000288 RECORD kl=2 dl=1 lvl=2 (aa)
    00000000
    00000198 00000198
000002B8 COMMIT start=00000288
```

Source code

http://git.cyrusimap.org/cyrus-imapd/tree/lib/cyrusdb_twoskip.c

- BSD Licence
- ~2000 lines of C code including db-driver
- Easy to rewrite in other languages (e.g. Perl!)