bring2lite: a structural Concept and Tool for Forensic Data Analysis and Recovery of Deleted SQLite Records

Chr. Meng, **H. Baier**

Hochschule Darmstadt, CRISP, da/sec Security Group

2019-07-15

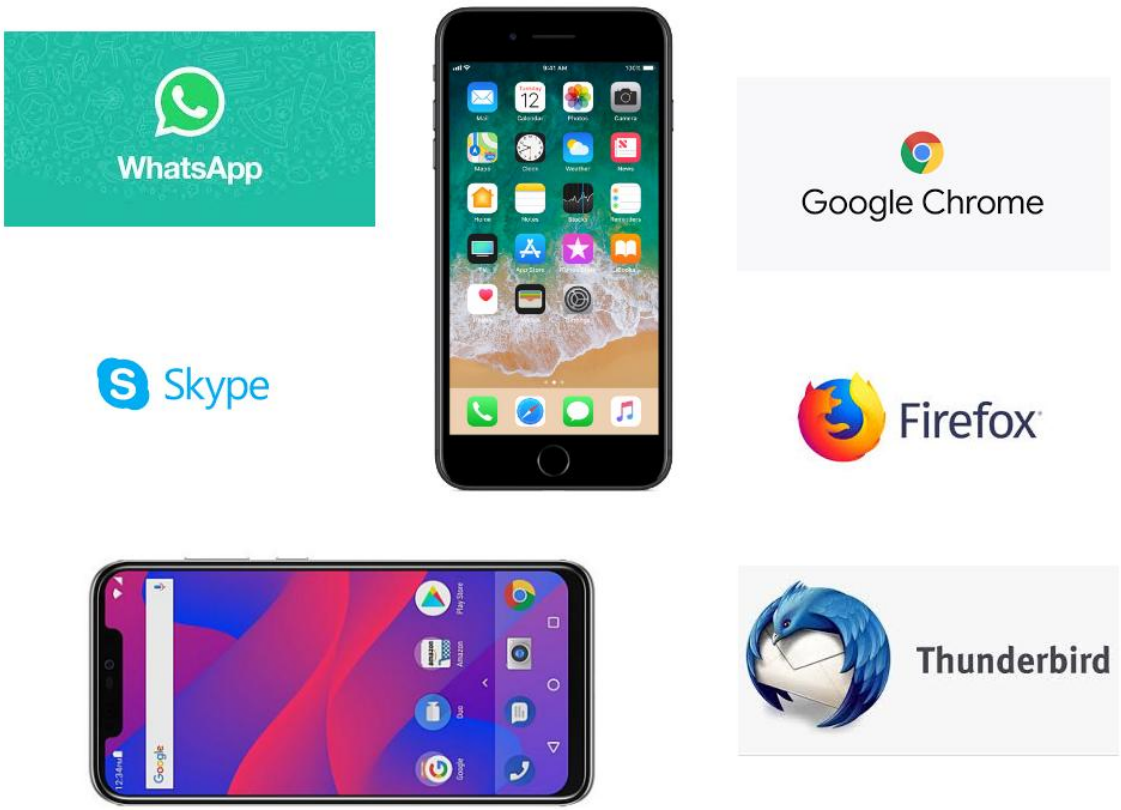
Location

Darmstadt

Agenda

Motivation

Widespread use



Widespread use $\Longrightarrow$ forensically highly relevant

Goals of this paper

1. Structural approach:
   - Analyse deletion behaviour of SQLite depending on different database parameters, which affect the erasure of database data
   - Relevant pragmas: `secure_delete`, `auto_vacuum`, `journal_mode`
2. Develop a concept to parse and process deleted SQLite records
3. Provide a proof of concept implementation: `bring2lite`
4. Evaluation with respect to a common test corpus

File format

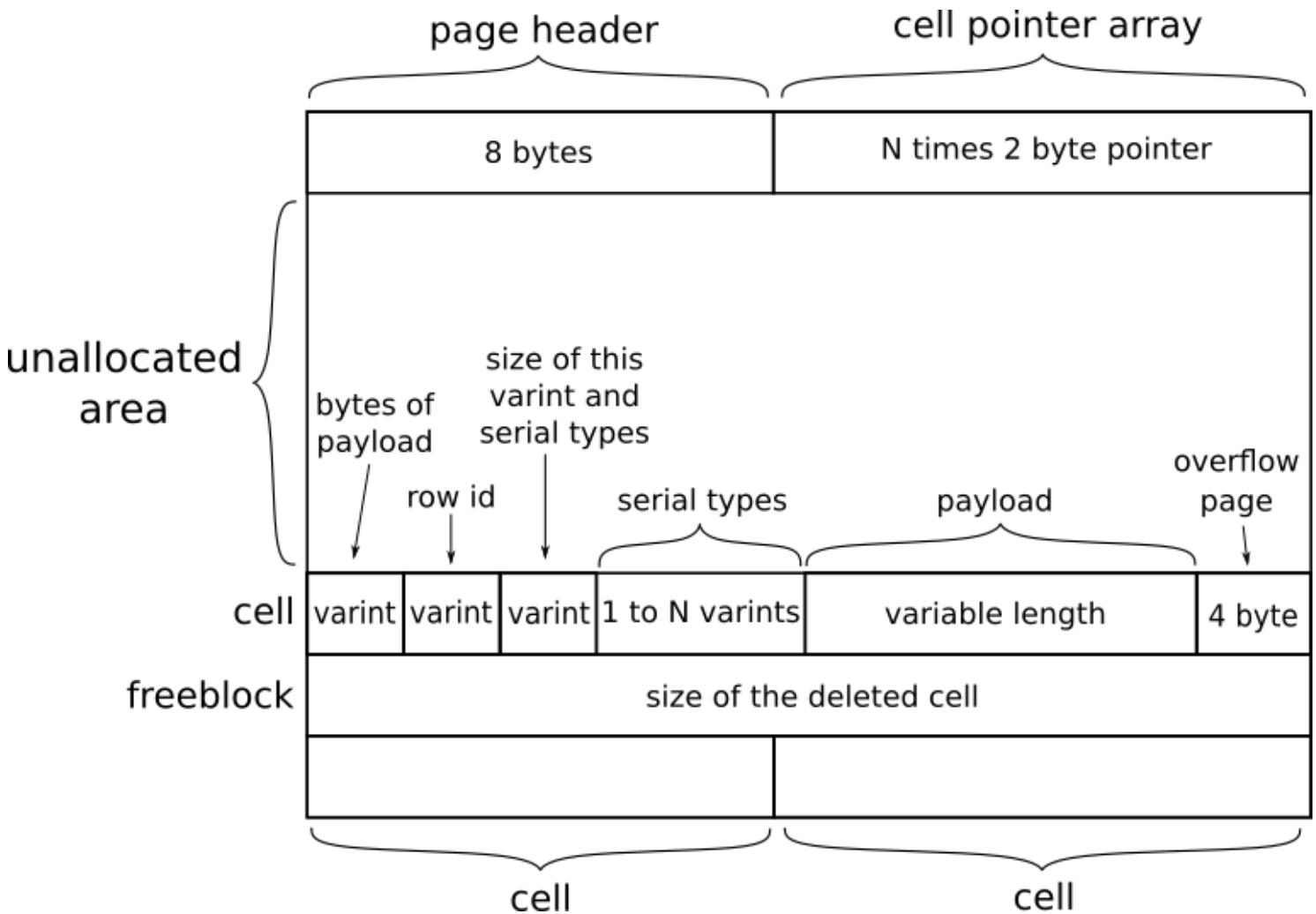
1. SQLite file organised in pages:
   - Page numbers start with 1
   - Page 1 contains SQLite file header (100 bytes) and root page of `sqlite_master` table
   - SQLite header defines page size (e.g., 4 KiB)
2. Five different page types:
   - b-tree pages for tables / indexes (interior, leaf pages)
   - Overflow pages
   - Freelist pages
   - Further two forensically irrelevant page types

Pragmas

1. A pragma is a configuration option.
2. Relevant pragmas in the scope of deletion are:
   - 2.1 `secure_delete`:
     - Settings 0, 1, FAST
     - `secure_delete` = 1: deleted content overwritten with zero bytes in database file
   - 2.2 `auto_vacuum`: if turned on, deletes unused pages (does not keep them in a freelist)
   - 2.3 `journal_mode`:
     - WAL journal (Write-ahead log)
     - Rollback journal file

Background

Table b-tree leaf page



Structural Analysis

Goal and scenarios

- Main goal: observe how SQLite removes records under different conditions, i.e. to learn the reality of SQLite deletion.
- Six scenarios, e.g.,

  Scenario S1 Insert 1 record, delete it.
  Scenario S2 Insert 3 records, delete 1 record.
  Scenario S5 Insert records until a second page will be created, delete all records on the first page.
  Scenario S6 Insert records until a second page will be created, delete all records.
- 12 pragma combinations for each scenario:

$$6 \cdot 12 = 72 \text{ test cases.}$$

- Generation of test files on our own.

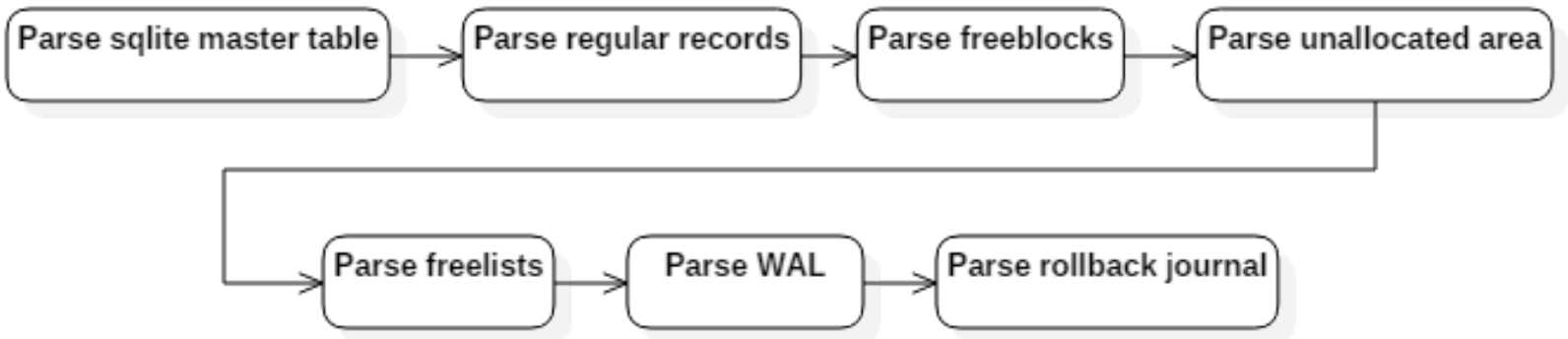
Structural Analysis

Sample results of structural analysis

Scenarios	secure_delete=0 / auto_vacuum=0 / journal_mode=OFF	secure_delete=FAST / auto_vacuum=0 / journal_mode=OFF	secure_delete=1 / auto_vacuum=0 / journal_mode=WAL	secure_delete=1 / auto_vacuum=0 / journal_mode=PERSIST
S1	+	-	+	+
S2	+	-	+	+
S3	+	-	+	+
S4	+	-	+	+
S5	+	0	+	+
S6	+	0	+	+

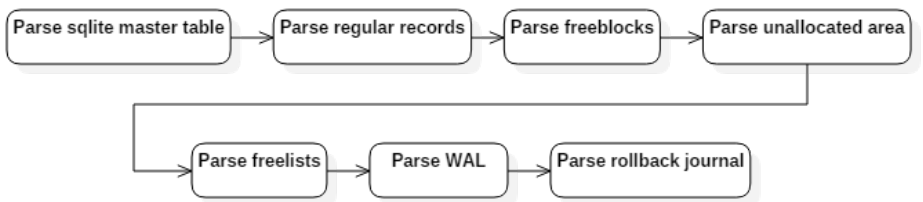
Concept

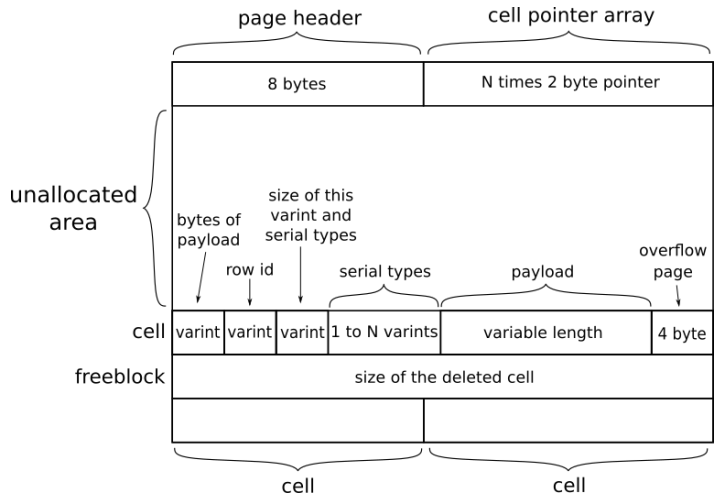
Concept (1/4)



- Start with processing SQLite database header
- Parse `sqlite_master` table:
  - Its root page is page 1 (directly following the database header)
  - Extract all schemas and connect every database page to its schema
- Parse regular records (i.e. all active database entries)

Concept (2/4)

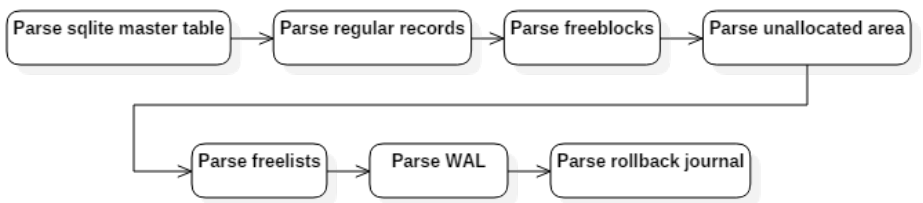


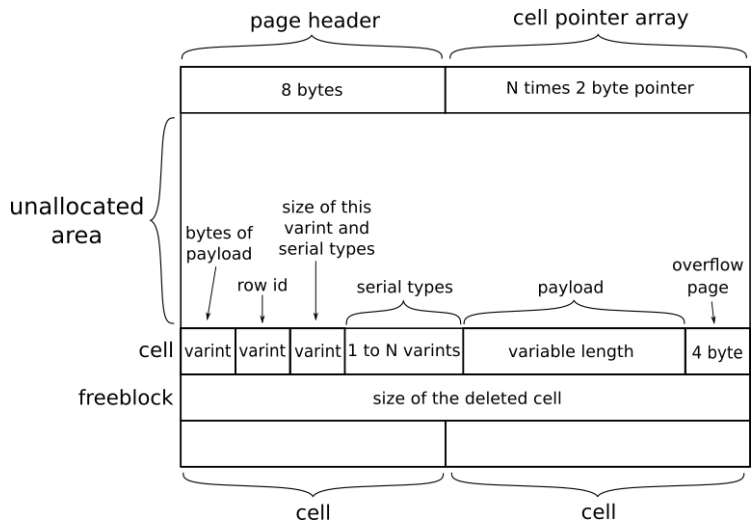


- Parse freeblocks (in an active page):
  - Entry point is page header entry
  - Linked list containing at least 4 bytes:
    - 2 bytes: pointer to subsequent freeblock (if there is any)
    - 2 bytes: length of freeblock
  - Cell header and payload header information partly overwritten

Concept

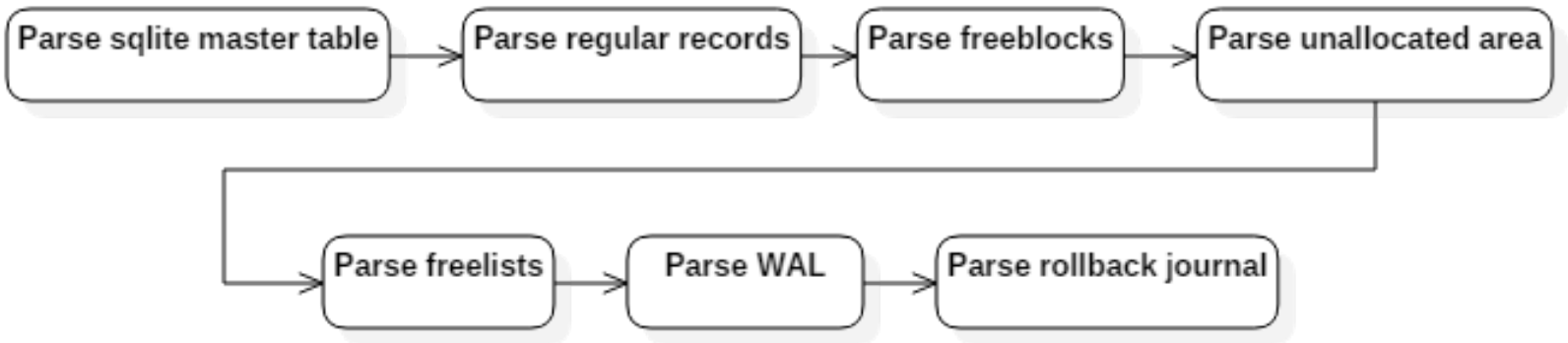
Concept (3/4)





- Parse unallocated area (in an active page):
  - Start after cell pointer area
  - Stop at cell area
  - Parse for bytes different from zero bytes

Concept (4/4)



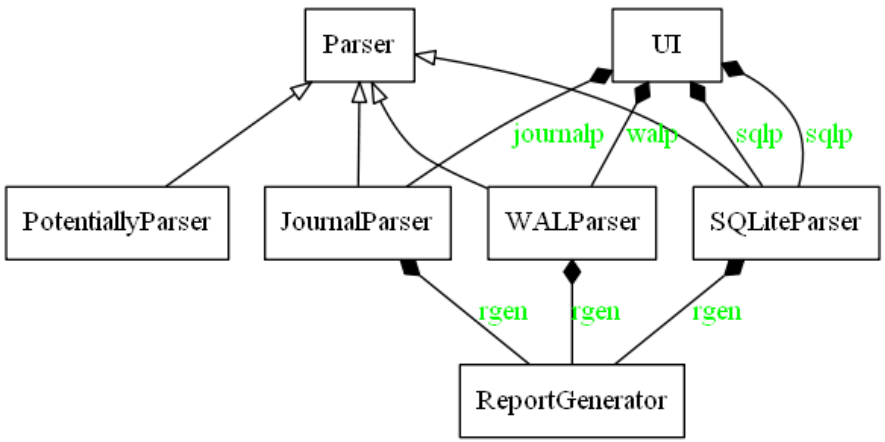
- Parse freelist pages:
  - Entry point is from SQLite database header entry
  - Freelist structure is easily processed
- If available, parse journal file (WAL, rollback journal)

The Tool bring2lite

The Tool bring2lite

Overview

- Proof of concept implemention in Python
- Class diagramme:



- Command line oriented
- Interprets SQLite data types (e.g., varints as integers)
- Source freely available via
  `https://github.com/bring2lite/bring2lite`

The Tool bring2lite

Sample usage

```
~/bring2lite$ python3.6 ./bring2lite/main.py \
  --filename ./db/database.sqlite --out ./results
```

```
~/bring2lite$ ls -l results
drwxr-xr-x cm cm 16384 Mr 27 20:34 freeblocks
drwxr-xr-x cm cm 16384 Mr 27 20:34 freelists
drwxr-xr-x cm cm 16384 Mr 27 20:34 regular-page-parsing
drwxr-xr-x cm cm 16384 Mr 27 20:34 schemas
drwxr-xr-x cm cm 16384 Mr 27 20:34 unalloc-parsing
```

Evaluation

Evaluation

Based on corpus by Nemetz, Schmitt, Freiling

A-01	create 1, insert 20, drop
A-02	create 1, insert 20, delete 20, drop
A-03	create 2, insert 10/each, drop each
A-04	create 2, insert 10/each, delete 10/1, drop each
A-05	create 2, insert 10/each, delete 10/each, drop each
B-01	create 1, insert 10, drop, create 1, insert 5
B-02	create 3, insert 10/each, drop 1, create 1, insert 5
C-01	create 1 (int cols), insert 20, delete 7
C-02	create 2 (int cols), insert 20/each, delete 5/each
C-03	create 1 (text cols), insert 20, delete 7
C-04	create 2 (text cols), insert 20/each, delete 5/each
C-05	create 2 (int/text), insert 20/each, delete 5/each
C-06	create 1 (float cols), insert 20, delete 7
C-07	create 2 (float cols), insert 20/each, delete 5/each
C-08	create 2 (float/text), insert 20/each, delete 5/each
C-09	create 1 (float/text), insert 10, delete 10
C-10	create 2 (float/text), insert 10/each, delete 10/each
D-01	create, insert 10, delete 5, insert 3
D-02	create, insert 10, delete 5, insert 5
D-03	create, insert 10, delete 5, insert 10: match 1
D-04	create, insert 10, delete 5, insert 3: match all
D-05	create, insert 10, delete 5, insert 5: match all
D-06	create, insert 10, delete 10, insert 5: match all
D-07	create 2, insert 10/each, delete 5/1, insert 5/2
D-08	create 2, insert 10/each (alt), delete 5/1, insert 5/2
E-01	records overflow like in 07-01, delete 7
E-02	records overflow like in 07-02, delete 5

Source: S. Nemetz, S. Schmitt, F. Freiling; A standardized corpus for SQLite database forensics; DFRWS EU 2018

Evaluation

Evaluation result

	Tools tested by the creators of the forensic corpus [15]						Tools tested by the authors of this paper		
Case	**Undark**	**SQLite Deleted Records Parser**	**SQLabs SQLite Doctor**	**Stellar Phoenix Repair for SQLite**	**SysTools SQLite Database Recovery**	**Sanderson Forensic Browser for SQLite**	**Sqlite Forensic Explorer**	**Autopsy SQLite Deleted Records Plugin**	**bring2lite**
0A-01	20/20*	0/20	0/20	0/20	0/20	0/20	0/20	0/20	20/20
0A-02	9/20*	20/20*	0/20	0/20	0/20	0/20	0/20	0/20	1/20
0A-03	20/20*	0/20	0/20	0/20	0/20	0/20	0/20	0/20	20/20
0A-04	15/20*	10/20*	0/20	0/20	0/20	0/20	0/20	0/20	13/20
0A-05	11/20*	20/20*	0/20	0/20	0/20	0/20	0/20	0/20	3/20
0B-01	0/10	0/10	0/10	0/10	0/10	0/10	0/10	0/10	4/10
0B-02	0/10	0/10	0/10	0/10	0/10	0/10	0/10	0/10	4/10
0C-01	0/7	0/7	0/7	0/7	0/7	7/7	7/7	0/7	6/7*
0C-02	0/10	0/10	0/10	0/10	0/10	10/10*	9/10	0/10	8/10*
0C-03	0/7	7/7	0/7	0/7	0/7	2/7	4/7	0/7	6/7*
0C-04	0/10	10/10*	0/10	0/10	0/10	1/10*	8/10	0/10	8/10*
0C-05	0/10	10/10*	0/10	0/10	0/10	10/10*	9/10	0/10	10/10
0C-06	0/7	0/7	0/7	0/7	0/7	0/7	5/7	0/7	6/7*
0C-07	0/10	0/10	0/10	0/10	0/10	0/10	10/10	0/10	9/10*
0C-08	0/10	10/10*	0/10	0/10	0/10	0/10	6/10	0/10	7/10*
0C-09	5/10*	10/10*	0/10	0/10	0/10	0/10	2/10	0/10	0/10
0C-10	11/20*	20/20*	0/20	0/20	0/20	0/20	2/20	0/20	5/20
0D-01	0/5	2/5*	0/5	0/5	0/5	0/5	1/5	0/5	1/5
0D-02	0/5	1/5*	0/5	0/5	0/5	0/5	1/5	0/5	1/5
0D-03	0/5	0/5	0/5	0/5	0/5	0/5	1/5	0/5	0/5
0D-04	0/5	2/5*	0/5	0/5	0/5	0/5	0/5	0/5	1/5*
0D-05	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5
0D-06	1/10*	5/10*	0/10	0/10	0/10	0/10	0/10	0/10	0/10
0D-07	0/5	5/5*	0/5	0/5	0/5	0/5	5/5	0/5	3/5*
0D-08	0/5	5/5*	0/5	0/5	0/5	0/5	3/5	0/5	3/5*
0E-01	3/7	2/7	0/7	0/7	0/7	3/7	0/7	0/7	5/7*
0E-02	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	3/5*
Sum	**95/278**	**139/278**	**0/278**	**0/278**	**0/278**	**33/278**	**73/278**	**0/278**	**147/278**

Sample database: 0B-02 (1/3)

Setting: create 3, insert 10/each, drop 1, create 1, insert 5

```
$ less 0B-02.sql

PRAGMA page_size=4096;
[REMOVED]

CREATE TABLE users ('id' INT UNSIGNED NOT NULL,
        'name' TEXT NOT NULL, 'surname' TEXT NULL,
        'codeA' INT NULL, 'codeB' FLOAT NULL
);

CREATE TABLE customers ('cid' INT UNSIGNED NOT NULL,
        'cname' TEXT NOT NULL, 'csurname' TEXT NULL,
        'ccodeA' INT NULL, 'ccodeB' FLOAT NULL
);
[REMOVED]
```

Sample database: 0B-02 (2/3)

- Files `0B-02.sql` and `0B-02.db` show:
  - Table `users` written to page 2 (10 entries)
  - Table `customers` written to page 3 (10 entries)
  - Table `supplier` written to page 4 (10 entries)
- Table `customers` dropped, i.e. page 3 released
- Table `members` written to page 3 (5 entries)
- Deleted content from table `customers` is in unallocated area of page 3 – if there is any:
  - Cell area in page 3 starts at offset `0x0f4f` of page 3
  - Deleted content is starting at offset `0x0ead` of page 3

Sample database: 0B-02 (3/3)

bring2lite retrieves these cells from dropped table customers

```
bring2lite/output$ ls
regular-page-parsing  schemas  unalloc-parsing

bring2lite/output$ ls unalloc-parsing
3-page.log

bring2lite/output$ less unalloc-parsing/3-page.log
INT,TEXT,TEXT,INT,REAL,
20010,Luisa,Kuhn,-1407291853,4892744407.93914,
20009,Christian,Schulze,527030628,4362154905.38727,
20008,Zoe,Schubert,-603005252,4007666590.16147,
20007,Luca,Scholz,1643805150,1166617011.72898,
++++++++++++++++++++++++++++++
```

Conclusion and Future Work

Conclusion and Future Work

- Scope: recovering deleted SQLite records
- Assess commercial and open source tools
- `bring2lite` performs best
- Future work:
  - Focus on test cases, where `bring2lite` performs poorly
  - Improve recovery performance of freeblocks and overflow pages
  - Sharpen the property 'meaningfulness' of recovered data
  - Consider anti-forensic measures

Contact

- harald.baier@h-da.de / meng.chr@googlemail.com
- Interested in internship at da/sec?