rawdisk Documentation

Release 0.2.1

Darius Bakunas-Milanowski

Sep 07, 2018

Contents

Contents:

CHAPTER 1

rawdisk

Experimental python code to explore different volume formats

- Free software: BSD license
- Documentation: https://rawdisk.readthedocs.org.

1.1 Features

- TODO

CHAPTER 2

Installation

At the command line:

```
$ easy_install rawdisk
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv rawdisk
$ pip install rawdisk
```

CHAPTER 3

Basic Usage

3.1 Loading data file

In order to start filesystem analysis, you need to create `Session` instance:

```
from rawdisk.session import Session

session = Session()
session.load('sample_images/ntfs_mbr.vhd')
```

Last line looks through available filesystem plugins in *rawdisk/plugins/filesystem*. If filesystem is matched, it initializes plugin's volume object. In order to print a list of available partitions (will only show those that were matched), type:

```
for volume in session.volumes:
    print(volume)
```

```
Type: NTFS, Offset: 0x10000, Size: 7.00MB, MFT Table Offset: 0x265000
```

3.2 Show selected volume information

To print selected volume information:

```
ntfs_vol = session.volumes[0]
ntfs_vol.dump_volume()
```

Output:

```
Volume Information
    Volume Name: NTFS Volume
    Volume Version: 3.1
    Volume Size: 7.00MB
```

(continues on next page)

(continued from previous page)

```
    Volume Offset: 0x10000
    Total Sectors: 14335
    Total Clusters: 1791
    MFT Offset: 0x265000 (from beginning of volume)
    MFT Mirror Offset: 0x2000
    MFT Record Size: 1.00KB
    MFT Size: 0.87MB (12% of drive)
```

3.3 Analysing selected partition

r.partitions is a list that contains matched volume objects. For example to get NTFS volume object (`NtfsVolume`)from the listing above:

```
ntfs_vol = session.volumes[0]
```

To get $MFT entry (index: 0):

```
mft = ntfs_vol.mft_table.get_entry(0)

mft.hexdump()
```

Output:

```
00000000: 46 49 4C 45 30 00 03 00  82 4D 10 00 00 00 00 00  FILE0....M......
00000010: 01 00 01 00 38 00 01 00  A0 01 00 00 00 04 00 00  ....8...........
00000020: 00 00 00 00 00 00 00 00  07 00 00 00 00 00 00 00  ................
00000030: 02 00 FF 00 00 00 00 00  10 00 00 00 60 00 00 00  ............`...
00000040: 00 00 18 00 00 00 00 00  48 00 00 00 18 00 00 00  ........H.......
00000050: F8 58 8A 44 11 01 D0 01  F8 58 8A 44 11 01 D0 01  .X.D.....X.D....
00000060: F8 58 8A 44 11 01 D0 01  F8 58 8A 44 11 01 D0 01  .X.D.....X.D....
00000070: 06 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000080: 00 00 00 00 00 01 00 00  00 00 00 00 00 00 00 00  ................

<...>
```

To print a list of attributes belonging to this $MFT entry:

```
for attr in mft.attributes:
    print attr
```

Output:

```
Type: $STANDARD_INFORMATION Name: N/A Resident Size: 96
Type: $FILE_NAME Name: N/A Resident Size: 104
Type: $DATA Name: N/A Non-Resident Size: 80
Type: $BITMAP Name: N/A Non-Resident Size: 72
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/dariusbakunas/rawdisk/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

rawdisk could always use more documentation, whether as part of the official rawdisk docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/dariusbakunas/rawdisk/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *rawdisk* for local development.

1. Fork the *rawdisk* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/rawdisk.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv rawdisk
$ cd rawdisk/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 rawdisk tests
$ python setup.py test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5. Check https://travis-ci.org/dariusbakunas/rawdisk/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_rawdisk
```

CHAPTER 5

Filesystem Diagrams

5.1 Master Boot Record

Offset	Size	Description
0	446 (0x1BE)	MBR Bootstrapper (executable code)
0x1BE	64 (0x40)	MBR Partition Table (4 entries, 16 bytes each)
0x1FE	2	Boot Signature (0x55, 0xAA)

Example:

```
 hexdump -C -n 512 sample_images/ntfs_mbr.vhd

00000000  33 c0 8e d0 bc 00 7c 8e  c0 8e d8 be 00 7c bf 00  |3.....|......|..|
00000010  06 b9 00 02 fc f3 a4 50  68 1c 06 cb fb b9 04 00  |.......Ph.......|
00000020  bd be 07 80 7e 00 00 7c  0b 0f 85 0e 01 83 c5 10  |....~..|........|
00000030  e2 f1 cd 18 88 56 00 55  c6 46 11 05 c6 46 10 00  |.....V.U.F...F..|
00000040  b4 41 bb aa 55 cd 13 5d  72 0f 81 fb 55 aa 75 09  |.A..U..]r...U.u.|
```

(continues on next page)

(continued from previous page)

```
00000050  f7 c1 01 00 74 03 fe 46  10 66 60 80 7e 10 00 74  |....t..F.f`.~..t|
00000060  26 66 68 00 00 00 00 66  ff 76 08 68 00 00 68 00  |&fh....f.v.h..h.|
00000070  7c 68 01 00 68 10 00 b4  42 8a 56 00 8b f4 cd 13  ||h..h...B.V.....|
00000080  9f 83 c4 10 9e eb 14 b8  01 02 bb 00 7c 8a 56 00  |............|.V.|
00000090  8a 76 01 8a 4e 02 8a 6e  03 cd 13 66 61 73 1c fe  |.v..N..n...fas..|
000000a0  4e 11 75 0c 80 7e 00 80  0f 84 8a 00 b2 80 eb 84  |N.u..~..........|
000000b0  55 32 e4 8a 56 00 cd 13  5d eb 9e 81 3e fe 7d 55  |U2..V...]...>.}U|
000000c0  aa 75 6e ff 76 00 e8 8d  00 75 17 fa b0 d1 e6 64  |.un.v....u.....d|
000000d0  e8 83 00 b0 df e6 60 e8  7c 00 b0 ff e6 64 e8 75  |......`.|....d.u|
000000e0  00 fb b8 00 bb cd 1a 66  23 c0 75 3b 66 81 fb 54  |.......f#.u;f..T|
000000f0  43 50 41 75 32 81 f9 02  01 72 2c 66 68 07 bb 00  |CPAu2....r,fh...|
00000100  00 66 68 00 02 00 00 66  68 08 00 00 00 66 53 66  |.fh....fh....fSf|
00000110  53 66 55 66 68 00 00 00  00 66 68 00 7c 00 00 66  |SfUfh....fh.|..f|
00000120  61 68 00 00 07 cd 1a 5a  32 f6 ea 00 7c 00 00 cd  |ah.....Z2...|...|
00000130  18 a0 b7 07 eb 08 a0 b6  07 eb 03 a0 b5 07 32 e4  |..............2.|
00000140  05 00 07 8b f0 ac 3c 00  74 09 bb 07 00 b4 0e cd  |......<.t.......|
00000150  10 eb f2 f4 eb fd 2b c9  e4 64 eb 00 24 02 e0 f8  |......+..d..$...|
00000160  24 02 c3 49 6e 76 61 6c  69 64 20 70 61 72 74 69  |$..Invalid parti|
00000170  74 69 6f 6e 20 74 61 62  6c 65 00 45 72 72 6f 72  |tion table.Error|
00000180  20 6c 6f 61 64 69 6e 67  20 6f 70 65 72 61 74 69  | loading operati|
00000190  6e 67 20 73 79 73 74 65  6d 00 4d 69 73 73 69 6e  |ng system.Missin|
000001a0  67 20 6f 70 65 72 61 74  69 6e 67 20 73 79 73 74  |g operating syst|
000001b0  65 6d 00 00 00 63 7b 9a  83 a1 56 0d 00 00 00 02  |em...c{...V.....|
000001c0  03 00 07 e5 25 00 80 00  00 00 00 38 00 00 00 00  |....%......8....|
000001d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
000001f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 55 aa  |..............U.|
00000200
```

5.2 MBR Partition Entry

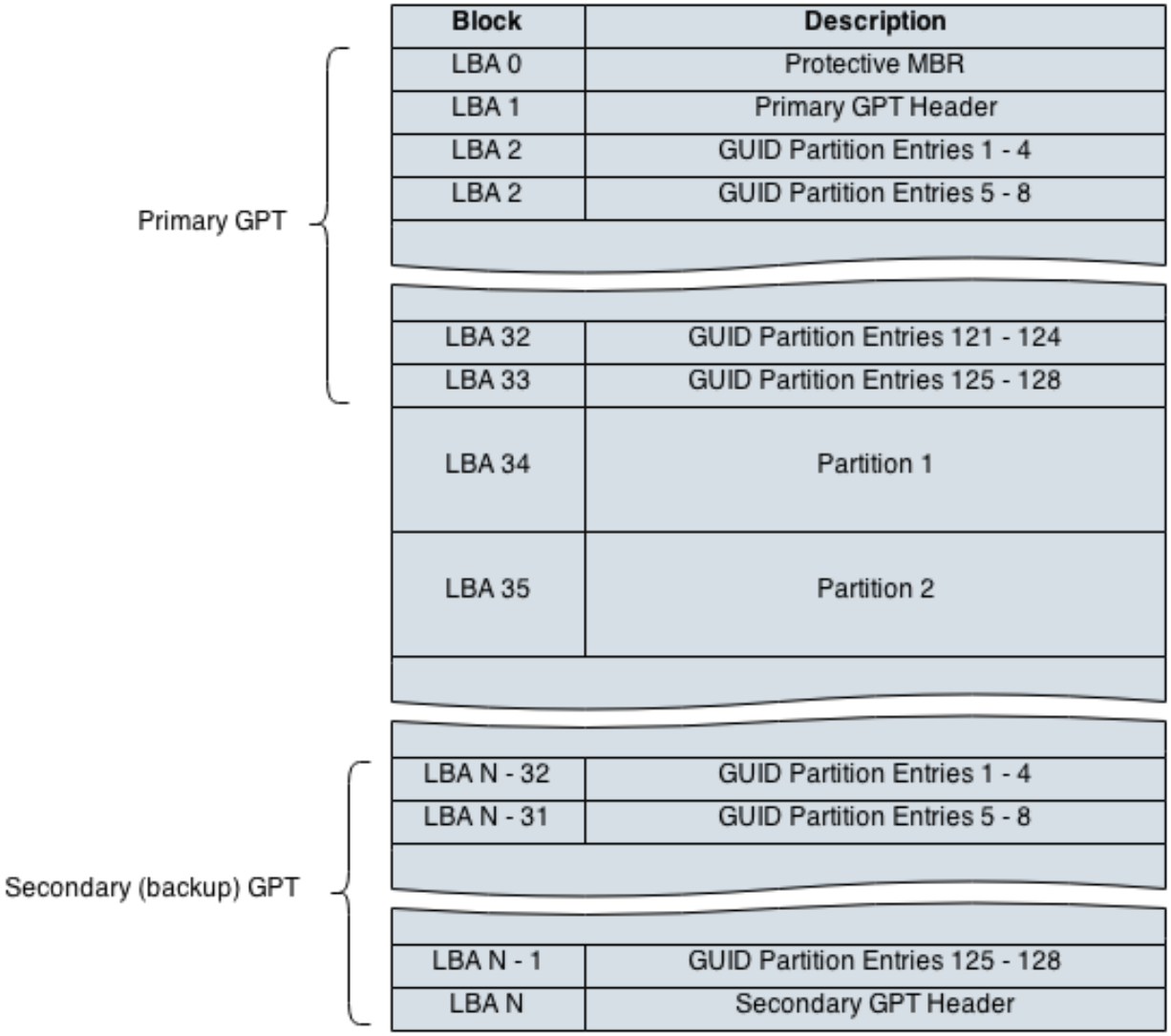
Relative Offset	Size (bytes)	Description
0	1	Boot Indicator (0x80 = active)
1	3	Starting CHS values
4	1	Partition Type Descriptor
5	3	Ending CHS values
8	4	Starting Sector
12 (0xC)	4	Total Sectors

Example:

```
% hexdump -C -s 0x1be -n 16 sample_images/ntfs_mbr.vhd

000001be  00 02 03 00 07 e5 25 00  80 00 00 00 00 38 00 00  |......%......8..|
```

5.3 GUID Partition Table



5.4 GPT Header

Relative Offset	Size (bytes)	Description
0	8	Signature ("EFI PART")
8	4	GPT Revision
12 (0x0C)	4	Header size in little endian
16 (0x10)	4	CRC32 for the header
20 (0x14)	4	Reserved (must be set to 0)
24 (0x18)	8	Current LBA
32 (0x20)	8	Backup LBA
40 (0x28)	8	First usable LBA for partitions (primary partition table last LBA + 1)
48 (0x30)	8	Last usable LBA for partitions (secondary partition table first LBA - 1)
56 (0x38)	16	Disk GUID also referred as UUID
72 (0x48)	8	Starting LBA of array of partition entries (always 2 in primary table)
80 (0x50)	4	Number of partition entries in array
84 (0x54)	4	Size of single partition entry (usually 128)
88 (0x58)	4	CRC32 of partition array
92 (0x5C)	*	Rserved. Must be zeroes for the rest of the block (420 for 512 byte sectors, but can be more for larger sectors)

Example:

```
>>> from rawdisk.scheme.gpt import Gpt
>>> gpt = Gpt()
>>> gpt.load('sample_images/ntfs_primary_gpt.bin')
>>> gpt.header.hexdump()

00000000: 45 46 49 20 50 41 52 54  00 00 01 00 5C 00 00 00  EFI PART....\...
00000010: 99 24 C1 58 00 00 00 00  01 00 00 00 00 00 00 00  .$.X............
00000020: FF FF 03 00 00 00 00 00  22 00 00 00 00 00 00 00  ........".......
00000030: DE FF 03 00 00 00 00 00  E5 66 99 AF FB 00 CD 45  .........f.....E
00000040: BE 63 26 2D 91 88 DC E7  02 00 00 00 00 00 00 00  .c&-............
00000050: 80 00 00 00 80 00 00 00  62 5A F4 F0              ........bZ..

>>> gpt.header.size
92
```

5.5 GPT Partition Entry

Relative Offset	Size (bytes)	Description
0	16	Partition type GUID
16 (0x10)	16	Unique partition GUID
32 (0x20)	8	First LBA (little endian)
40 (0x28)	8	Last LBA (inclusive)
48 (0x30)	8	Attribute flags (bit 60 denotes read-only)
56 (0x38)	72	Partition name (36 UTF-16LE symbols)

Example:

```
>>> from rawdisk.scheme.gpt import Gpt
>>> gpt = Gpt()
>>> gpt.load('sample_images/ntfs_primary_gpt.bin')
>>> gpt.partition_entries[1].hexdump()

00000000: A2 A0 D0 EB E5 B9 33 44  87 C0 68 B6 B7 26 99 C7  ......3D..h..&..
00000010: 19 67 C5 5C 2B 2E 46 4A  B4 55 C5 C2 6E 74 67 5C  .g.\+.FJ.U..ntg\
00000020: 80 00 01 00 00 00 00 00  7F F0 03 00 00 00 00 00  ................
00000030: 00 00 00 00 00 00 00 00  42 00 61 00 73 00 69 00  ........B.a.s.i.
00000040: 63 00 20 00 64 00 61 00  74 00 61 00 20 00 70 00  c. .d.a.t.a. .p.
00000050: 61 00 72 00 74 00 69 00  74 00 69 00 6F 00 6E 00  a.r.t.i.t.i.o.n.
00000060: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000070: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

CHAPTER 6

NTFS Diagrams

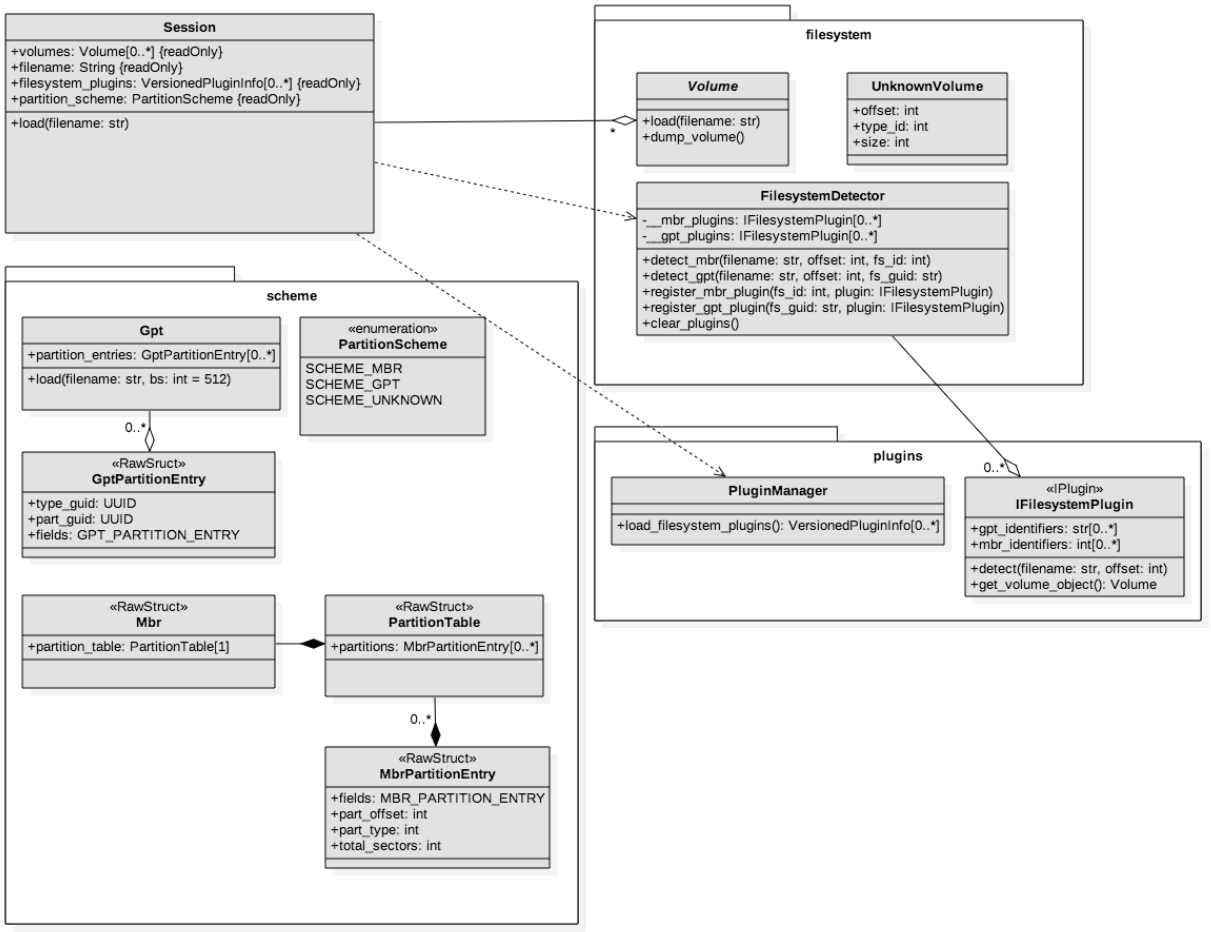
6.1 Bootsector

Relative Offset	Size (bytes)	Description
0	3	Jump Instruction
3	8	OEM ID
11 (0x0B)	25	BPB
36 (0x24)	48	Extended BPB
84 (0x54)	426	Bootstrap Code
510 (0x01FE)	2	End of Sector Marker (0x55, 0xAA)

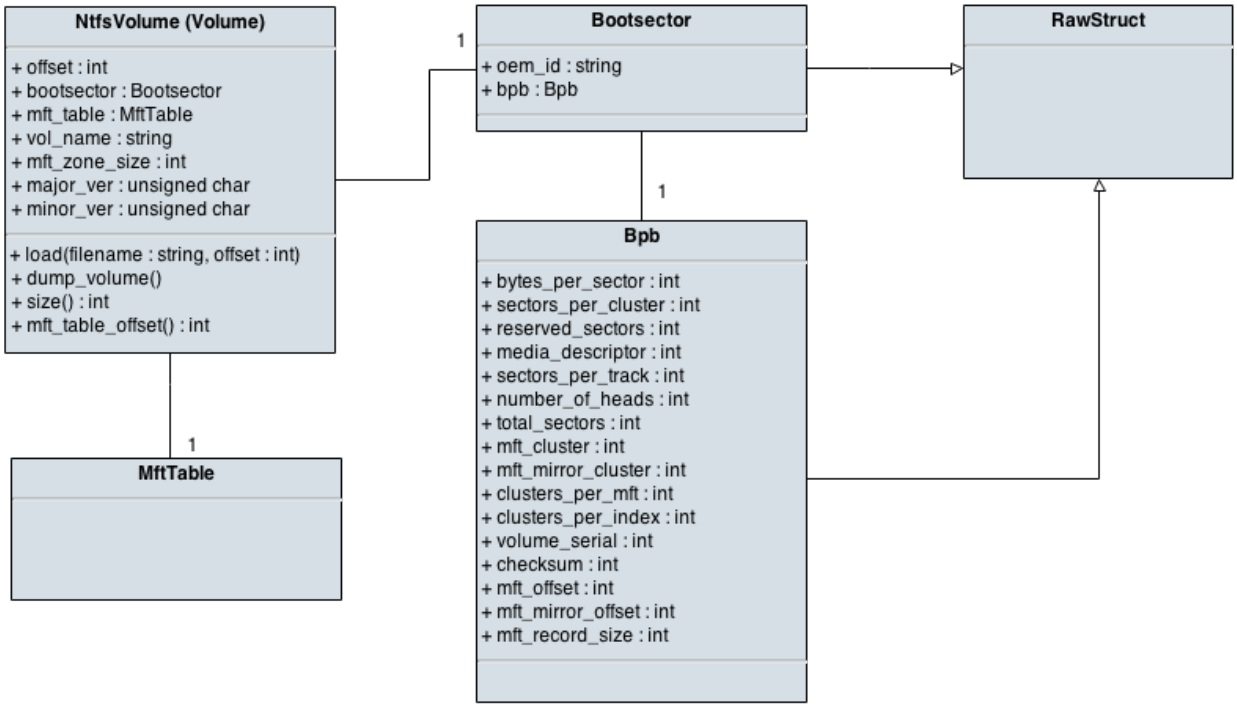
CHAPTER 7

Rawdisk Diagrams

7.1 Main Class Diagram



7.2 NTFS Volume Class Diagram



CHAPTER 8

References

- All about BIOS parameter blocks
- GUID Partition Table
- How Basic Disks and Volumes Work
- NTFS
- NTFS Partition Boot Sector
- Secrets of the GPT

CHAPTER 9

Credits

9.1 Development Lead

- Darius Bakunas-Milanowski <bakunas@gmail.com>

9.2 Contributors

None yet. Why not be the first?

CHAPTER 10

History

10.1 0.2.0 (2017-05-13)

- Cleared spiderwebs
- Fixed failing builds
- Converted project to Python 3.5
- Updated usage examples

10.2 0.1.2 (2014-11-21)

- Added couple filesystem binary samples
- New filesystem diagrams
- New rawdisk class diagrams
- Enabled code coverage for Travis CI
- Added waffle.io badge
- Added more tests
- Couple fixes for plugin files
- Removed hurry.filesize dependency
- Removed Py2.6 compatibility

10.3 0.1.1 (2014-11-10)

- Moved to the project template generated by cookiecutter.

CHAPTER 11

Indices and tables

- genindex
- modindex
- search