Looking Back

Volatility 3 Public Beta: The Insider’s Preview

© 2019 The Volatility Foundation

February 2006
FATKit (Digital Investigation Journal)

February 2007
VolaTools (Black Hat - Initial Public Release)

August 2007
Volatility 1.1.1 (Scanning and VAD)

August 2008
Volatility 1.3 (DFRWS Contest, OMFW, and Plugins)

January 2009
Malfind 1.0

August 2011
Volatility Technology Preview (Internal Project)

December 2011
Volatility Technology Preview (Internal Project)

October 2012
OMFW: Volatility 3.0 on Roadmap (Technology Preview)

November 2013
"Volatility Past & Present" (Volatility 3.0: Python 3/Pagefile/Performance)

December 2013
Volatility Technology Preview → Rekall

November 2014
"Restructuring Memory" (Unified output) → 2.5

December, 2016
Volatility 2.6 (Windows 10 / Server 2016)

October 2015
Volatility 2.5 (Unified Output / Community)

September 2014
Volatility Foundation created

August 2014
Volatility 2.4 (Art of Memory Forensics)

October 2013
Volatility Plugin Contest

May 2013
Month of Volatility Plugins II

October 2012
Volatility 2.2 (Linux Support)

September 2012
Month of Volatility Plugins I

August 2012
Volatility 2.1 (Malware and 64-bits)

August 2011
Volatility 2.0 (Beyond XP)

October 2019
Volatility 3.0 Public Beta

Volatility 3

Volatility 2.x
Memory Forensics: 2006 vs. 2019

**Sample Size**
- **Then**: <= 4GB
  - **Now**: >= 16GB
  - >= 128GB common

**Sample Quantity**
- **Then**: 1
- **Now**: 5+

**Volatility Analysis Tasks**
- **Then**: <= 10
- **Now**: >= 50
Operating System Release Cycles in 2019 [3, 4]

• **Feature updates** will be released *twice per year* around March and September. As the name suggests, these will add new features to Windows 10, delivered in bite-sized chunks compared to the previous practice of Windows releases *every 3-5 years*.

• **Linux_5.3** Released Sun, 15 September 2019 (70 days)
• **Linux_5.2** Released Sun, 7 July 2019 (63 days)
• **Linux_5.1** Released Sun, 5 May 2019 (63 days)
• **Linux_5.0** Released Sun, 3 March 2019 (70 days)
The History of Vol3

• Many novel ideas attempted and refined before being put into the stable code base

• The goal: Meet the needs of the next decade of memory analysis
What is New in Volatility 3?

• **All** of it

• **Every** line of code

• **Entire framework** (backend, plugins, etc.) was completely rewritten and redesigned

• Written in Python 3

• Major performance boost!
  – Natively supports multi-processing and memory caches

• Much simpler integration into other libraries and user interfaces

• No more --profile for any OS!
  – Automatic detection of profiles
  – Extraction of known-good data from debug info vs hardcoded

• 32bit apps on 64bit kernels natively supported
  – Proper Wow64 analysis!

• Automated evaluation of in-memory code
What is New for Developers?

• Extensive API documentation

• Plugins can directly call other plugins

• Plugins are versioned

• Much easier to use custom data structures and symbols
$ time python3 vol.py -f turlanew.raw -r pretty windows.pssl
Volatility 3 Framework 1.0.0-beta.1
<table>
<thead>
<tr>
<th>PID</th>
<th>PPID</th>
<th>ImageFileName</th>
<th>Offset(V)</th>
<th>Threads</th>
<th>Handles</th>
<th>SessionID</th>
<th>Wow64</th>
<th>CreateTime</th>
<th>ExitTime</th>
</tr>
</thead>
</table>
* 4  | 0   | System        | 0xfa8030eb5840 | 94     | 413     | N/A       | False | 2018-07-31 17:39:31.000000 | N/A      |
* 276 | 4   | smss.exe      | 0xa80325e9700 | 2      | 30      | N/A       | False | 2018-07-31 17:39:31.000000 | N/A      |
* 372 | 360 | csrss.exe     | 0xa803185710 | 10     | 553     | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 428 | 360 | wininit.exe   | 0xa803351e060 | 3      | 79      | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 512 | 428 | services.exe  | 0xa80335a6060 | 9      | 239     | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 544 | 428 | lsass.exe     | 0xa80335d3710 | 8      | 585     | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 552 | 428 | lsm.exe       | 0xa80335d42d0 | 11     | 149     | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 648 | 512 | svchost.exe   | 0xa8033821a70 | 13     | 372     | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 708 | 512 | vmmachtlp.exe | 0xa8033856910 | 3      | 56      | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 1004 | 812 | audiodg.exe   | 0xa8033951630 | 6      | 136     | 0         | False | 2018-07-31 17:39:32.000000 | N/A      |
* 1632 | 2568 | notepad.exe  | 0xa80311e7d0 | 2      | 75      | 2         | False | 2019-04-11 19:35:28.000000 | N/A      |
* 2920 | 2568 | wordpad.exe  | 0xa8031a66060 | 5      | 132     | 2         | False | 2019-04-11 19:35:31.000000 | N/A      |

real  0m1.116s
user  0m0.760s
sys   0m0.076s
Call help() to see available functions

Volshell mode: Generic
Current Layer: primary2

(primary2) >>> hh()

Methods:
* dt, display_type
  Display Type describes the members of a particular object in alphabetical order
* db, display_bytes
  Displays byte values and ASCII characters
* dw, display_words
  Displays word values (2 bytes) and corresponding ASCII characters
* dd, display_doublewords
  Displays double-word values (4 bytes) and corresponding ASCII characters
* dq, display_quadwords
  Displays quad-word values (8 bytes) and corresponding ASCII characters
* dis, disassemble
  Disassembles a number of instructions from the code at offset
* cl, change_layer
  Changes the current default layer
* dpo, display_plugin_output
  Displays the output for a particular plugin (with keyword arguments)
* gt, generate_treegrid
  Generates a TreeGrid based on a specific plugin passing in kwarg configuration values
* rt, render_treegrid
  Renders a treegrid as produced by generate_treegrid
* ds, display_symbols
  Prints an alphabetical list of symbols for a symbol table
Supporting Modern and Advanced Analytics

• Automating (where possible) operating system and application support

• Automating analysis decisions beyond simply presenting data structures and raw disassembly listings

• Automating analysis of multiple samples at once
Automated Kernel Module Analysis – NDIS & Netfilter [5, 6]

With the hooks installed, whenever the network adapter driver attempts to register to NDIS, or whenever there is an attempt to install NDIS intermediate driver or NDIS filter driver, the hook handlers will register Snake’s own MiniportXxx functions with the NDIS library.

With its own miniport handler functions, it can send/receive data by using a private TCP/IP stack, bypassing all firewall hooks, and making its open ports invisible to scanners.

```
1007    magic_packet_hook_options.hook = (void *)magic_packet_hook;
1008    magic_packet_hook_options.hooknum = 0;
1009    magic_packet_hook_options.pf = PF_INET;
1010    magic_packet_hook_options.priority = NF_IP_PRI_FIRST;
```
Automated Version Analysis – TrueCrypt vs VeraCrypt [7, 8]

Tuesday, January 14, 2014

TrueCrypt Master Key Extraction And Volume Identification

- Windows:
  - Installer: VeraCrypt Setup 1.24.exe (34.2 MB) (PGP Signature)
  - Portable version: VeraCrypt Portable 1.24.exe (34 MB) (PGP Signature)
  - Debugging Symbols: VeraCrypt_1.24_Windows_Symbols.zip (9.45 MB) (PGP Signature)
### Automatic Symbol Inclusion

```bash
$ python3 vol.py -f sample.vmem windows.ssdtt
Volatility 3 Framework 1.0.0-beta.1

<table>
<thead>
<tr>
<th>Index</th>
<th>Address</th>
<th>Module</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0xf800034dea50</td>
<td>ntoskrnl</td>
<td>NtMapUserPhysicalPagesScatter</td>
</tr>
<tr>
<td>1</td>
<td>0xf800033c40a0</td>
<td>ntoskrnl</td>
<td>NtWaitForSingleObject</td>
</tr>
<tr>
<td>2</td>
<td>0xf800030c68a0</td>
<td>ntoskrnl</td>
<td>NtCallbackReturn</td>
</tr>
<tr>
<td>3</td>
<td>0xf800033b7210</td>
<td>ntoskrnl</td>
<td>NtReadFile</td>
</tr>
</tbody>
</table>
```

```bash
$ python3 vol.py -f sample.vmem windows.callbacks
Volatility 3 Framework 1.0.0-beta.1

<table>
<thead>
<tr>
<th>Type</th>
<th>Callback</th>
<th>Module</th>
<th>Symbol</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>PspLoadImageNotifyRoutine</td>
<td>0xf800033ea520</td>
<td>ntoskrnl</td>
<td>EtwpTraceLoadImage</td>
<td>N/A</td>
</tr>
<tr>
<td>PspCreateProcessNotifyRoutine</td>
<td>0xf80003099590</td>
<td>ntoskrnl</td>
<td>ViCreateProcessCallback</td>
<td>N/A</td>
</tr>
</tbody>
</table>
```

Automated Emulation of In-Memory Hooks [9]

Hook mode: Usermode
Hook type: Inline/Trampoline
Process: 3068 (iexplore.exe)
Victim module: ntdll.dll (0x77640000 - 0x7777c000)
Function: ntdll.dll!LdrLoadDll at 0x776a22b8
Hook address: 0x74c601f8
Hooking module: <unknown>

Disassembly(0):
0x776a22b8 e93bdf5bf0d  JMP 0x74c601f8

Disassembly(1):
0x74c601f8 e9c3daaeb  JMP 0x6071d0c0

3068 iexplore.exe ntdll.dll!LdrLoadDll at 0x776a22b8
PAGE_EXECUTE_READWRITE <Non-File Backed Region: 0x74c60000
0x74c6afff> (4)
PAGE_EXECUTE_WRITECOPY \Device\HarddiskVolume1\Program Files\AVG\Antivirus\snxhk.dll (2)
PAGE_EXECUTE_READWRITE <Non-File Backed Region: 0x74c60000
0x74c6afff> (46)
PAGE_EXECUTE_WRITECOPY \Device\HarddiskVolume1\Program Files\AVG\Antivirus\aswhookx.dll (2)
PAGE_EXECUTE_READWRITE <Non-File Backed Region: 0x6f670000
0x6f67ff00> (4)
PAGE_EXECUTE_WRITECOPY \Device\HarddiskVolume1\Windows\System32\ntdll.dll (2)
Automatically Analyzing Multiple Samples

Volatility 2
1. Run kdbgscan (or imageinfo)
2. <wait>
3. Set --profile
4. Run plugin

Volatility 3
1. Run plugin

```bash
$ for sample in $(ls samples//*.mem); do python3 vol.py -f $sample windows.psl|grep System;done
```

<table>
<thead>
<tr>
<th>Sample</th>
<th>Offset</th>
<th>Size</th>
<th>Type</th>
<th>N/A</th>
<th>Valid Date</th>
<th>Start Time</th>
<th>End Time</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0</td>
<td>0x8a13f280</td>
<td>120</td>
<td>-</td>
<td>N/A</td>
<td>False</td>
<td>2019-09-30 03:16:14.000000</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0x86569040</td>
<td>109</td>
<td>-</td>
<td>N/A</td>
<td>False</td>
<td>2019-01-05 02:29:56.000000</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0x8954f040</td>
<td>121</td>
<td>-</td>
<td>N/A</td>
<td>False</td>
<td>2019-07-15 21:29:58.000000</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0x85f6c040</td>
<td>118</td>
<td>-</td>
<td>N/A</td>
<td>False</td>
<td>2019-08-02 00:39:33.000000</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Volatility 3.0 Licensing - Volatility Software License 1.0

- The Volatility Foundation will no longer release code under the GPL
- Short, simple, plain language copyleft license
  - No philosophical agendas or drama!
- Volatility Software License 1.0
  - Grants all rights of copyright (“Freedom Zero”)
  - Requires changes to the software are shared
  - Covers content and data
- Legal Advisor: Heather Meeker
- License Inspiration
  - API Copyleft License (Kyle E. Mitchell)
  - Nmap Public Source License
Looking Forward

August 2020 Volatility 3.0 Official Release

Volatility 3

Feature parity/new & unique capabilities

August 2021 – future Volatility development & support ONLY for 3.x

2020 2021

Plugin & operating system updates

Volatility 2.x

August 2021 2.x: Development & support for 2.x Ends
Start Using It and Get Involved!

- https://www.github.com/volatilityfoundation/volatility3
- https://volatility3.rtfd.io/
- https://www.volatilityfoundation.org/slack
- https://lists.volatilityfoundation.org/pipermail/vol-users/
References

[6] https://github.com/f0rb1dd3n/Reptile/