# The Art of Memory Forensics :

# Simple Volatility Plugin Development Guide

DONG HYUN KIM / KITRI BoB Digital Forensic Track / 2017. 01.

There are many private data in the presentation

Please do not leak outside!

# Agenda

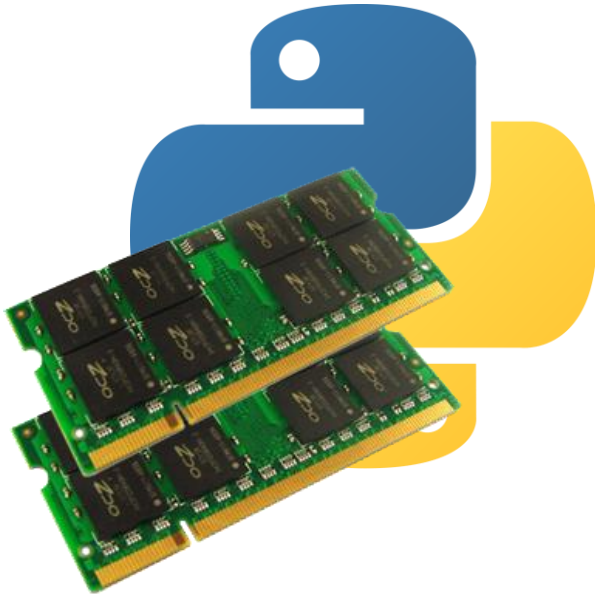
- Tutorial
- Introduction to Volatility
- Plugin Architecture
- Volatility Plugin Development Practice
- Case Study #1 (Malcom)
- Case Study #2 (Facebook)
- Plugin Development Tip

# Tutorial

A Simple description of Memory Forensics

Today, we focus on **Volatility** rather than memory forensics.

And we have experience with simple memory forensics.



# Necessity and Advantages of Memory Forensics

- **Necessity**

Malicious code that loads directly into memory

Delete for privacy

- **Advantages**

Repeatable investigation possible

Various methods available

# Memory is essential for running program

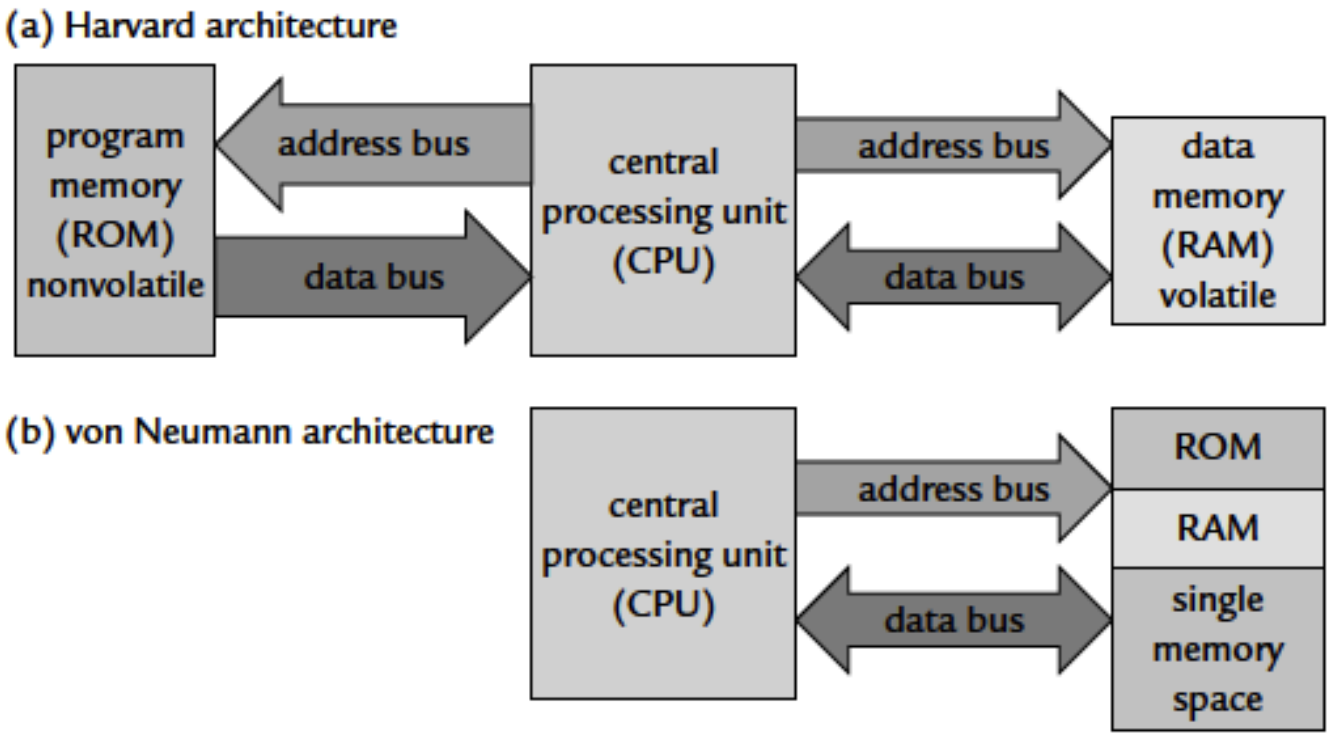
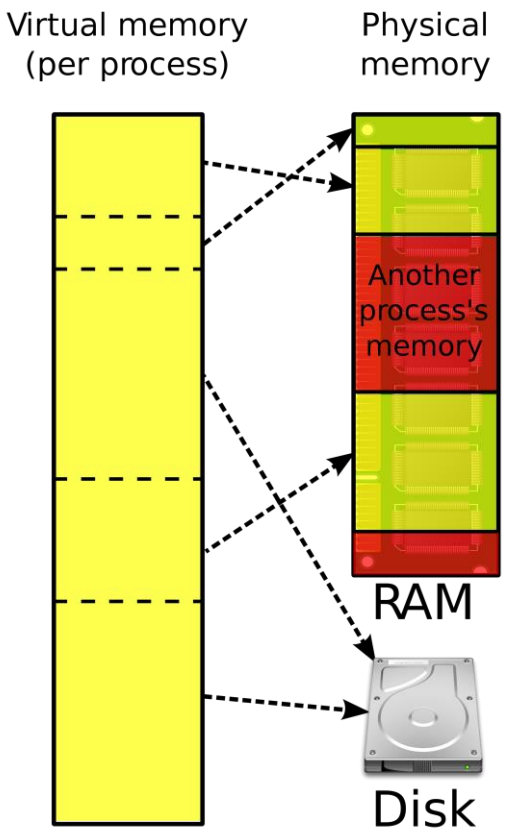
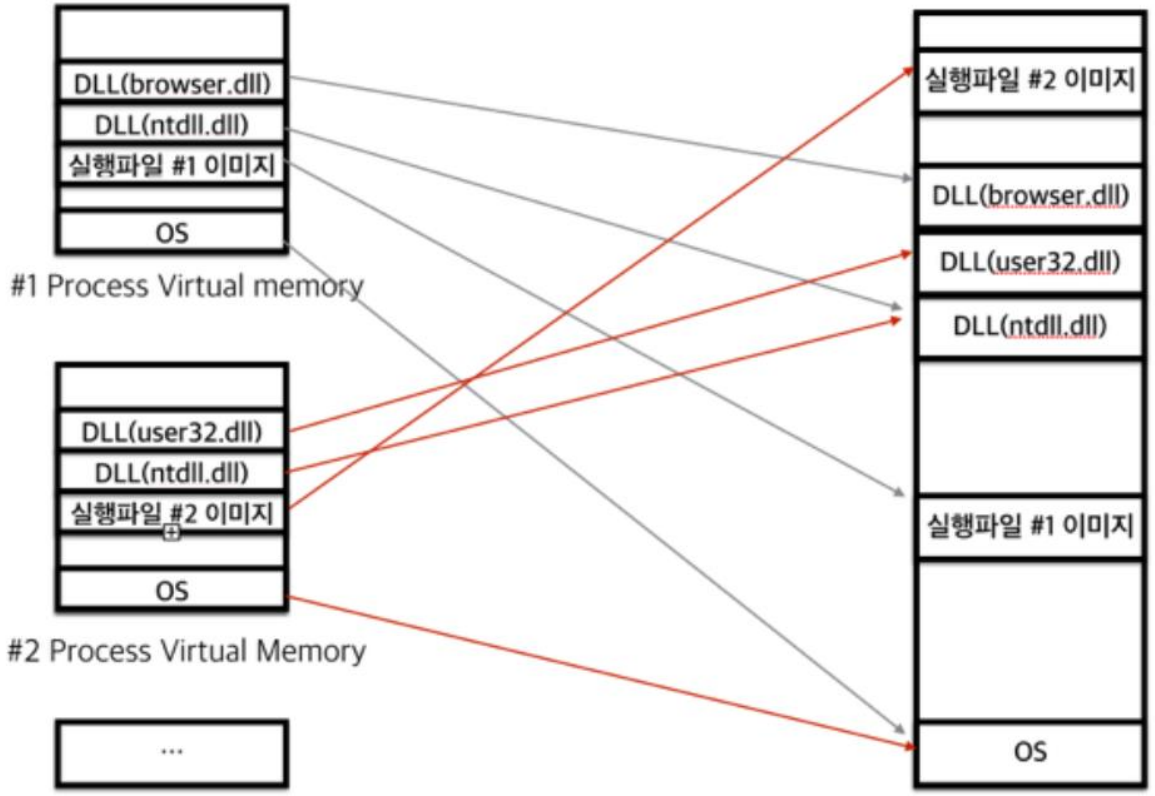



Figure 1.3: Harvard and von Neumann architectures for memory.

# Virtual Memory & Physical Memory



# _EPROCESS Structure

```
0: kd> dt nt!_EPROCESS
   +0x000 Pcb              : _KPROCESS
   +0x06c ProcessLock      : _EX_PUSH_LOCK
   +0x070 CreateTime       : _LARGE_INTEGER
   +0x078 ExitTime         : _LARGE_INTEGER
   +0x080 RundownProtect   : _EX_RUNDOWN_REF
   +0x084 UniqueProcessId  : Ptr32 Void
   +0x088 ActiveProcessLinks : _LIST_ENTRY
   +0x090 QuotaUsage       : [3] Uint4B
   +0x09c QuotaPeak        : [3] Uint4B
   +0x0a8 CommitCharge     : Uint4B
   +0x0ac PeakVirtualSize  : Uint4B
   +0x0b0 VirtualSize      : Uint4B
   +0x0b4 SessionProcessLinks : _LIST_ENTRY
   +0x0bc DebugPort        : Ptr32 Void
   +0x0c0 ExceptionPort    : Ptr32 Void
   +0x0c4 ObjectTable      : Ptr32 _HANDLE_TABLE
   +0x0c8 Token            : _EX_FAST_REF
   +0x0cc WorkingSetLock   : _FAST_MUTEX
   +0x0ec WorkingSetPage   : Uint4B
   +0x0f0 AddressCreationLock : _FAST_MUTEX
```

- 주요 EPROCESS 항목

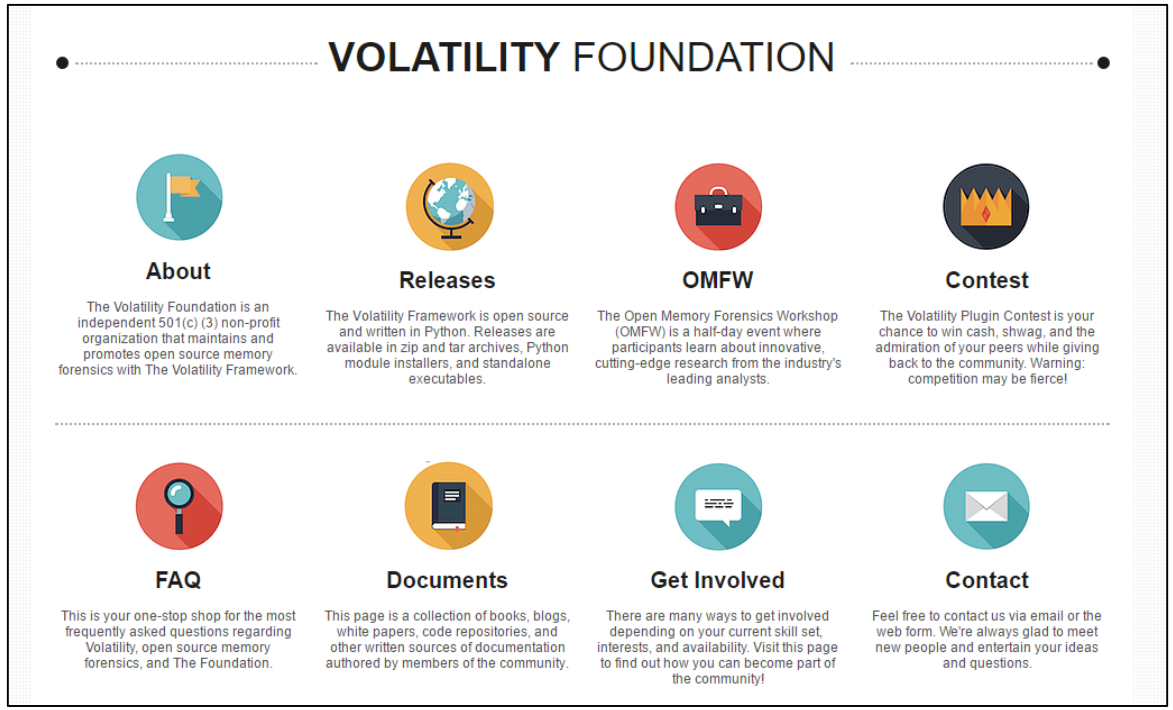
| 항목 | 데이터 타입 | 설명 |
|---|---|---|
| PCB<br>(Process Control Block) | _KPROCESS | DISPATCHER_HEADER, 디렉터리 테이블 주소, KTHREAD 목록, 우선 순위, 커널/유저 CPU 시간 등 |
| CreateTime | _LARGE_INTEGER | 64비트 윈도우 시간 형식의 프로세스 시작 시간 |
| ExitTime | _LARGE_INTEGER | 64비트 윈도우 시간 형식의 프로세스 종료 시간 |
| UniqueProcessId | Ptr32 Void | 프로세스 ID |
| ActiveProcessLinks | _LIST_ENTRY | ActiveProcess List를 구성하는 이중 링크드 리스트 |
| ObjectTable | Ptr32_HANDLE_TABLE | 오브젝트 핸들 테이블의 위치를 가르키는 포인터 |
| WorkingSetPage | Uint4B | 프로세스 WorkingSetPage |
| Peb<br>(Process Environment Block) | Ptr32_PEB | 프로세스 실행에 필요한 정보, 실행파일/DLL 경로, 베이스 어드레스, 모듈 리스트, 힙/스택 정보 등 |

**Windbg using** : http://www.ahnlab.com/kr/site/securityinfo/secunews/secuNewsView.do?&seq=17602&menu_dist=1

**Windows Forensic Guide** : https://drive.google.com/file/d/0B4ueNC1Cr70VTHpUaFk5OE9jOTA/view?usp=sharing

# Introduction to Volatility

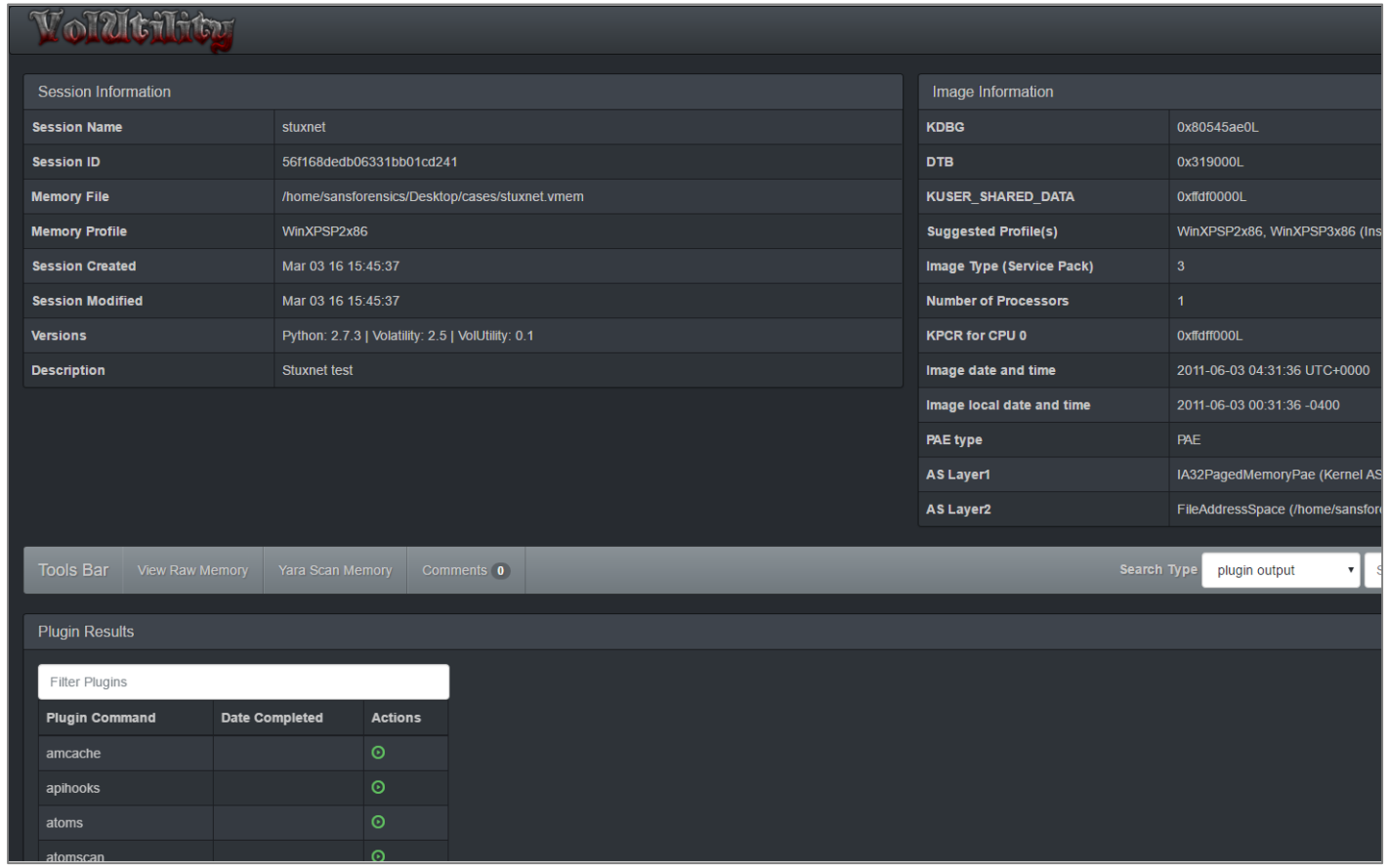
What is Volatility? And Where to use?




**Download :** http://www.volatilityfoundation.org/26

# Volatility

- Memory Forensic Framework
- Open Source
- Based on Python
- Support to Various OS
- Release New version 2.6




# Function

- Process Info
- VAD Info
- Registry Hive Info
- File System Info
- Connection Info

# Volatility Labs

Monday, December 5, 2016

## Results from the 2016 Volatility Plugin Contest are in!

Congratulations to all the participants! This year we received more submissions than ever before (21 to be exact, from 16 different authors), so judging took longer than we expected. Sorry about that! The good news is...there's a LOT of new and exciting functionality available to law enforcement agents, DF/IR practitioners, malware analysts, and researchers around the globe, which can immediately be transitioned into their workflows. That's the whole spirit of open source memory forensics with Volatility, and we're once again very proud to sponsor a contest with such impressive results.

It may sound cheesy, but everyone is a winner in this contest. Although a few developers will walk away with prizes, they all solved a problem that they (and inevitably others) faced, gained experience writing Python plugins, and learned some intricacies of memory analysis internals. The capability to program around technical issues and design/implement solutions is a gift. You can applaud by following the authors on Twitter/GitHub/LinkedIn, providing feedback on their ideas, and helping to improve their code with testing, documentation, or contributing patches.

We also want to thank Airbnb for donating $999 to the cash prizes! When looking for a new job, we definitely recommend considering employers that support open source forensics and value the importance of memory analysis. Maybe you can be their next Security CSIRT Engineer!

---

Here is a break down of the placements and prizes

---

1st place and $1800 USD cash or a Free Seat at Malware and Memory Forensics Training by the Volatility Project goes to:

> Monnappa for Hollow Process Detection and Analysis. Monnappa also participated in the 2015, so this is his second consecutive contest. Aside from the code itself, Monnappa's corresponding documentation was very impressive. Given Monnappa has already taken our training class, he'll likely take the cash prize!

2nd place and $800 USD cash goes to:

> Kevin Breen for VolUtility and LastPass Credential Recovery. Although we've seen web interfaces in the past, Kevin's take on it has a lot of unique and helpful features. He's already integrated quite a number of new capabilities since the contest closed in October and he's showing no signs of slowing down. He's got Volatility plugin fever!

# Volatility Labs

Tuesday, October 2, 2012

## MoVP 4.2 Taking Screenshots from Memory Dumps

**Month of Volatility Plugins**

Open Memory Forensics Workshop 2012 is currently in progress, thus today's MoVP post will be short and sweet. However, it will still introduce an exciting new capability exclusive to Volatility.

One of Brendan Dolan Gavitt's early GDI utilities for Volatility included a screenshot plugin. The plugin drew "wire-frame" rectangles of windows according to their positions on the desktop. Its far from a real screenshot, but nonetheless is very exciting from a memory forensics perspective. BDG also wrote a plugin using VM introspection and PyGame to actively trace user's mouse movements and window interactions based on the changes they made in physical memory. These are both major developments that show abstract ways you can leverage the power of Volatility.

For the upcoming release of Volatility 2.2, I took the liberty of updating BDG's screenshot plugin to work with the latest core code base and include support for all major windows versions. Since the original plugin only worked on XP x86, there was no need to examine multiple desktops at that time (it was before session 0 isolation). However, nowadays, the plugin will output one screenshot for each desktop – including the desktops seen via RDP and multiple logged-on users.

The Screenshots Plugin

The inner workings of the plugin are quite simple. It enumerates windows for each desktop in their Z-Order (front-to-back focus) just as described in MoVP 2.2 Malware In Your Windows. It takes the left, right, top and bottom coordinates of each window from the `tagWND` structure and draws rectangles with PIL (Python Imaging Library).

To demonstrate, two users logged into the same Windows 7 box with fast-user switching. Each user left various windows open. Then memory was acquired and the screenshots plugin was run. As shown below, you just pass it a `-D/--dump-dir` parameter for the PNG files to be saved.

```
$ python vol.py -f users.vmem --profile=Win7SP1x86 screenshot -D shots/
Volatile Systems Volatility Framework 2.1_alpha
Wrote shots/session_0.Service-0x0-3e4$.Default.png
Wrote shots/session_0.Service-0x0-3e5$.Default.png
Wrote shots/session_0.msswindowstation.mssrestricteddesk.png
Wrote shots/session_0.Service-0x0-3e7$.Default.png
Wrote shots/session_1.WinSta0.Default.png
Wrote shots/session_1.WinSta0.Disconnect.png
Wrote shots/session_1.WinSta0.Winlogon.png
```

# We enable to use standalone version.

# But if you want to develop Plugin, Please install it!

- Source Code of Volatility 2.5 or 2.6 version (Github or Foundation)

  **Github Source Download :** https://github.com/volatilityfoundation/volatility/releases

- Memory Dump Image File (.vmem, Real Dump … , DC 301)
- Python and Essential Package (Github)
- Text Editor (Self, Sublime Text)

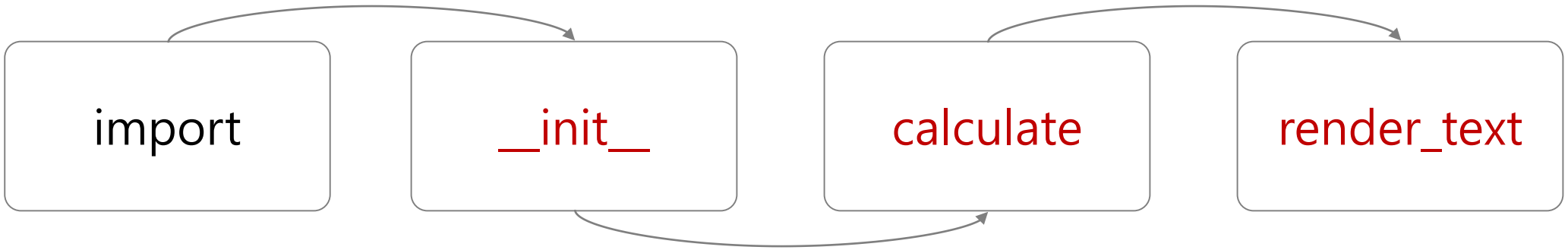
# Plugin Architecture

Sketching is important in painting

# Many functions exist. But we need three functions.

- Class *Plugin-Name() : Input Plugin name and help text
- def __init__() : Initializing functions and specifying options
- def calculate() : Calculations required for running plugin
- def render_text : Output the result of the calculation

# Overall progress of the plugin




**Reconstructing a binary from memory**

- Source Code of Volatility 2.5 or 2.6 version (Github or Foundation)
- Github Source Download :
- Memory Dump Image File (.vmem, Real Dump … , DC 301)

# Save your plugin inside the plugin path.

**Path :** C:₩Users₩[username]₩Desktop₩volatility-2.6₩volatility-master₩volatility₩plugins

| | | | |
|---|---|---|---|
| contrib | 2017-01-02 오후... | 파일 폴더 | |
| pyinstaller | 2017-01-02 오후... | 파일 폴더 | |
| resources | 2017-01-02 오후... | 파일 폴더 | |
| tools | 2017-01-02 오후... | 파일 폴더 | |
| volatility | 2017-01-02 오후... | 파일 폴더 | |
| .gitattributes | 2016-12-27 오전... | GITATTRIBUTES 파... | 1KB |
| .gitignore | 2016-12-27 오전... | GITIGNORE 파일 | 1KB |
| AUTHORS | 2016-12-27 오전... | 텍스트 문서 | 1KB |
| CHANGELOG | 2016-12-27 오전... | 텍스트 문서 | 24KB |
| CREDITS | 2016-12-27 오전... | 텍스트 문서 | 4KB |
| LEGAL | 2016-12-27 오전... | 텍스트 문서 | 1KB |
| LICENSE | 2016-12-27 오전... | 텍스트 문서 | 15KB |
| Makefile | 2016-12-27 오전... | 파일 | 1KB |
| MANIFEST.in | 2016-12-27 오전... | IN 파일 | 1KB |
| PKG-INFO | 2016-12-27 오전... | 파일 | 1KB |
| pyinstaller.spec | 2016-12-27 오전... | SPEC 파일 | 1KB |
| README | 2016-12-27 오전... | 텍스트 문서 | 32KB |
| setup.py | 2016-12-27 오전... | PY 파일 | 4KB |
| vol.py | 2016-12-27 오전... | PY 파일 | 7KB |

| | | | |
|---|---|---|---|
| plugins | 2017-01-03 오전... | 파일 폴더 | |
| renderers | 2017-01-02 오후... | 파일 폴더 | |
| win32 | 2017-01-02 오후... | 파일 폴더 | |
| __init__.py | 2016-12-27 오전... | PY 파일 | 1KB |
| addrspace.py | 2016-12-27 오전... | PY 파일 | 16KB |
| cache.py | 2016-12-27 오전... | PY 파일 | 24KB |
| commands.py | 2016-12-27 오전... | PY 파일 | 13KB |
| conf.py | 2016-12-27 오전... | PY 파일 | 15KB |
| constants.py | 2016-12-27 오전... | PY 파일 | 2KB |
| debug.py | 2016-12-27 오전... | PY 파일 | 3KB |
| dwarf.py | 2016-12-27 오전... | PY 파일 | 14KB |
| exceptions.py | 2016-12-27 오전... | PY 파일 | 2KB |
| fmtspec.py | 2016-12-27 오전... | PY 파일 | 4KB |
| obj.py | 2016-12-27 오전... | PY 파일 | 47KB |
| poolscan.py | 2016-12-27 오전... | PY 파일 | 16KB |
| protos.py | 2016-12-27 오전... | PY 파일 | 7KB |
| registry.py | 2016-12-27 오전... | PY 파일 | 7KB |
| scan.py | 2016-12-27 오전... | PY 파일 | 7KB |
| timefmt.py | 2016-12-27 오전... | PY 파일 | 5KB |
| utils.py | 2016-12-27 오전... | PY 파일 | 6KB |
| validity.py | 2016-12-27 오전... | PY 파일 | 2KB |

| | | | |
|---|---|---|---|
| addrspaces | 2017-01-02 오후... | 파일 폴더 | |
| gui | 2017-01-02 오후... | 파일 폴더 | |
| linux | 2017-01-02 오후... | 파일 폴더 | |
| mac | 2017-01-02 오후... | 파일 폴더 | |
| malware | 2017-01-02 오후... | 파일 폴더 | |
| overlays | 2017-01-02 오후... | 파일 폴더 | |
| registry | 2017-01-02 오후... | 파일 폴더 | |
| __init__.py | 2016-12-27 오전... | PY 파일 | 2KB |
| bigpagepools.py | 2016-12-27 오전... | PY 파일 | 8KB |
| bioskbd.py | 2016-12-27 오전... | PY 파일 | 3KB |
| cmdline.py | 2016-12-27 오전... | PY 파일 | 3KB |
| common.py | 2016-12-27 오전... | PY 파일 | 4KB |
| connections.py | 2016-12-27 오전... | PY 파일 | 4KB |
| connscan.py | 2016-12-27 오전... | PY 파일 | 4KB |
| crashinfo.py | 2016-12-27 오전... | PY 파일 | 9KB |
| dlldump.py | 2016-12-27 오전... | PY 파일 | 7KB |
| drivermodule.py | 2016-12-27 오전... | PY 파일 | 5KB |
| dumpcerts.py | 2016-12-27 오전... | PY 파일 | 11KB |
| dumpfiles.py | 2016-12-27 오전... | PY 파일 | 58KB |
| envars.py | 2016-12-27 오전... | PY 파일 | 6KB |
| evtlogs.py | 2016-12-27 오전... | PY 파일 | 14KB |
| fileparam.py | 2016-12-27 오전... | PY 파일 | 2KB |
| filescan.py | 2016-12-27 오전... | PY 파일 | 17KB |
| getservicesids.py | 2016-12-27 오전... | PY 파일 | 41KB |
| getsids.py | 2016-12-27 오전... | PY 파일 | 11KB |
| handles.py | 2016-12-27 오전... | PY 파일 | 6KB |
| heaps.py | 2016-12-27 오전... | PY 파일 | 2KB |
| hibinfo.py | 2016-12-27 오전... | PY 파일 | 3KB |
| hpakinfo.py | 2016-12-27 오전... | PY 파일 | 3KB |
| iehistory.py | 2016-12-27 오전... | PY 파일 | 12KB |
| imagecopy.py | 2016-12-27 오전... | PY 파일 | 5KB |
| imageinfo.py | 2016-12-27 오전... | PY 파일 | 6KB |
| joblinks.py | 2016-12-27 오전... | PY 파일 | 7KB |
| kdbgscan.py | 2016-12-27 오전... | PY 파일 | 12KB |
| kpcrscan.py | 2016-12-27 오전... | PY 파일 | 8KB |
| machoinfo.py | 2016-12-27 오전... | PY 파일 | 2KB |
| mbrparser.py | 2016-12-27 오전... | PY 파일 | 23KB |
| mftparser.py | 2016-12-27 오전... | PY 파일 | 44KB |
| moddump.py | 2016-12-27 오전... | PY 파일 | 6KB |
| modscan.py | 2016-12-27 오전... | PY 파일 | 7KB |
| modules.py | 2016-12-27 오전... | PY 파일 | 5KB |
| multiscan.py | 2016-12-27 오전... | PY 파일 | 3KB |
| netscan.py | 2016-12-27 오전... | PY 파일 | 11KB |

# Simple Practice

Simple exercises before running

```python
import volatility.commands as commands
import volatility.utils and utils
import volatility.win32.tasks and tasks

class BoBPlugin(commands.Command):
    """This is BoB's Test Plugin"""

    def calculate(self):
        kernel_space = utils.load_as(self._config)

        for process in tasks.pslist(kernel_space):
            yield process
```

**Virtual Base Address Return,**

**Add astype='physical'**

**pslist is EPROCESS Address Return**

Public Member Functions

| | |
|---|---|
| def | __init__ (self, config, _args, _kwargs) |
| def | help (cls) |
| def | calculate (self) |
| def | execute (self) |
| def | format_value (self, value, fmt) |
| def | table_header (self, outfd, title_format_list=None) |
| def | table_row (self, outfd, args) |
| def | text_cell_renderers (self, columns) |
| def | unified_output (self, data) |
| def | render_text (self, outfd, data) |
| def | render_greptext (self, outfd, data) |
| def | render_json (self, outfd, data) |
| def | render_sqlite (self, outfd, data) |
| def | render_dot (self, outfd, data) |
| def | render_html (self, outfd, data) |
| def | render_xlsx (self, outfd, data) |

◆ pslist()

```
def volatility.win32.tasks.pslist (  addr_space )
```

A Generator for _EPROCESS objects

Definition at line 85 of file tasks.py.

References volatility.win32.tasks.get_kdbg().

◆ load_as()

```
def volatility.utils.load_as (  config,
                                astype = 'virtual',
                                kwargs
                              )
```

Loads an address space by stacking valid ASes on top of each other (priority order first)

Definition at line 31 of file utils.py.

References volatility.win32.rawreg.values().

# EPROCESS Structure member Print

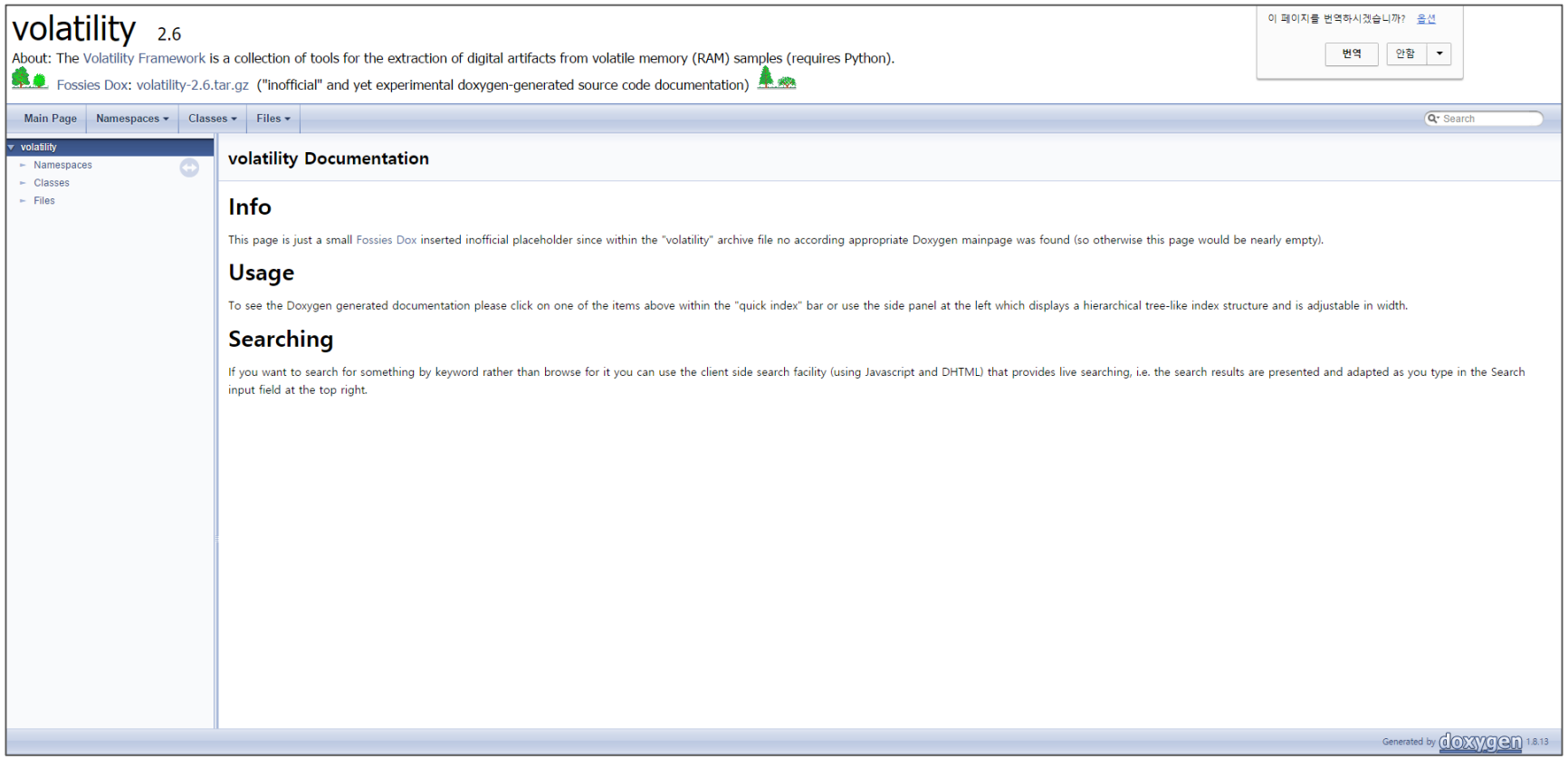
```
def render_text(self, outfd, data):
    for process in data:
        outfd.write("Process:{0}, PID{1} \n".format(process.ImageFileName, process.UniqueProcessId))
```

```
+0x154 VadFreeHint          : Ptr32 Void
+0x158 VdmObjects           : Ptr32 Void
+0x15c DeviceMap            : Ptr32 Void
+0x160 PhysicalVadList      : _LIST_ENTRY
+0x168 PageDirectoryPte     : _HARDWARE_PTE
+0x168 Filler               : Uint8B
+0x170 Session              : Ptr32 Void
+0x174 ImageFileName        : [16] UChar
+0x184 JobLinks             : _LIST_ENTRY
+0x18c LockedPagesList      : Ptr32 Void
+0x190 ThreadListHead       : _LIST_ENTRY
+0x198 SecurityPort         : Ptr32 Void
+0x19c PaeTop               : Ptr32 Void
+0x1a0 ActiveThreads        : Uint4B
```

```
kd> dt _EPROCESS
nt!_EPROCESS
   +0x000 Pcb              : _KPROCESS
   +0x06c ProcessLock      : _EX_PUSH_LOCK
   +0x070 CreateTime       : _LARGE_INTEGER
   +0x078 ExitTime         : _LARGE_INTEGER
   +0x080 RundownProtect   : _EX_RUNDOWN_REF
   +0x084 UniqueProcessId  : Ptr32 Void
   +0x088 ActiveProcessLinks : _LIST_ENTRY
   +0x090 QuotaUsage       : [3] Uint4B
   +0x09c QuotaPeak        : [3] Uint4B
   +0x0a8 CommitCharge     : Uint4B
   +0x0ac PeakVirtualSize  : Uint4B
   +0x0b0 VirtualSize      : Uint4B
```

?

# Volatility API Reference




**Link :** https://fossies.org/dox/volatility-2.6/index.html

# Volatility API Reference

## Public Member Functions

```
def __init__ (self, config, args, kwargs)
def dump_pe (self, space, base, dump_file)
def calculate (self)
def unified_output (self, data)
def generator (self, data)
def render_text (self, outfd, data)
```

### ◆ dump_pe()

```
def volatility.plugins.procdump.ProcDump.dump_pe (  self,
                                                    space,
                                                    base,
                                                    dump_file
                                                 )
```

```
Dump a PE from an AS into a file.

@param space: an AS to use
@param base: PE base address
@param dump_file: dumped file name

@returns a string status message
```

Definition at line 51 of file procdump.py.

References volatility.commands.Command._config, and volatility.addrspace.BaseAddressSpace._config.

Referenced by volatility.plugins.moddump.ModDump.generator(), volatility.plugins.dlldump.DLLDump.generator(), volatility.plugins.procdump.ProcDump.generator(), volatility.plugins.moddump.ModDump.render_text(), volatility.plugins.dlldump.DLLDump.render_text(), and volatility.plugins.procdump.ProcDump.render_text().

# Case Study #1

Malcom Plugin


# Malcom

- Use Procdump plugin
- Use Malwares.com API
- Simple & Easy code
- Support to Various OS
- Support to Version 2.6

```
import os
import sys
import shutil
import requests
import json
import volatility.plugins.procdump as procdump
```

**Use Procdump Plugin**

| | | |
|---|---|---|
| objtypescan | 2017-01-09 오전... | Compiled Python... |
| patcher | 2016-12-27 오전... | Python File |
| patcher | 2017-01-09 오전... | Compiled Python... |
| patchguard | 2016-12-27 오전... | Python File |
| patchguard | 2017-01-09 오전... | Compiled Python... |
| pooltracker | 2016-12-27 오전... | Python File |
| pooltracker | 2017-01-09 오전... | Compiled Python... |
| privileges | 2016-12-27 오전... | Python File |
| privileges | 2017-01-09 오전... | Compiled Python... |
| procdump | 2016-12-27 오전... | Python File |
| procdump | 2017-01-09 오전... | Compiled Python... |
| pstree | 2016-12-27 오전... | Python File |
| pstree | 2017-01-09 오전... | Compiled Python... |
| raw2dmp | 2016-12-27 오전... | Python File |
| raw2dmp | 2017-01-09 오전... | Compiled Python... |
| sockets | 2016-12-27 오전... | Python File |
| sockets | 2017-01-09 오전... | Compiled Python... |

| Name | Size | Type |
|---|---|---|
| executable.224.exe | 224 KB | Application |
| executable.408.exe | 52 KB | Application |
| executable.412.exe | 84 KB | Application |
| executable.464.exe | 84 KB | Application |
| executable.516.exe | 60 KB | Application |
| executable.588.exe | 20 KB | Application |
| executable.612.exe | 512 KB | Application |
| executable.648.exe | 196 KB | Application |
| executable.656.exe | 112 KB | Application |
| executable.668.exe | 24 KB | Application |
| executable.872.exe | 232 KB | Application |
| executable.888.exe | 24 KB | Application |
| executable.984.exe | 24 KB | Application |
| executable.1020.exe | 24 KB | Application |
| executable.1048.exe | 24 KB | Application |
| executable.1056.exe | 40 KB | Application |
| executable.1232.exe | 24 KB | Application |
| executable.1304.exe | 24 KB | Application |
| executable.1516.exe | 64 KB | Application |
| executable.1876.exe | 196 KB | Application |
| executable.1928.exe | 1,020 KB | Application |
| executable.1932.exe | 196 KB | Application |
| procdumpinfo.txt | 34 KB | Text Document |

## Reconstructing a binary from memory

- http://computer.forensikblog.de/en/2006/04/reconstructing-a-binary-1.html
- http://computer.forensikblog.de/en/2006/04/reconstructing-a-binary-2.html

```
malcom_key = "--- Your Malwares.com API Key ---"

class malcom(procdump.ProcDump):
    """Process Dump & Malwares.com Scan Plugin."""
```

## Set Plugin name & Help text

## Set option –o, -p

```
def __init__(self, config, *args, **kwargs):
    procdump.ProcDump.__init__(self, config, *args, **kwargs)
    config.add_option('OFFSET', short_option = 'o', default = None, help = 'EPROCESS offset (in hex) in the physical address space', action = 'store', type = 'int')
    config.add_option('PID', short_option = 'p', default = None, help = 'Operate on these Process IDs (comma-separated)', action = 'store', type = 'str')
```

***malwares.com API Instruction***

*Version 2.0.14*

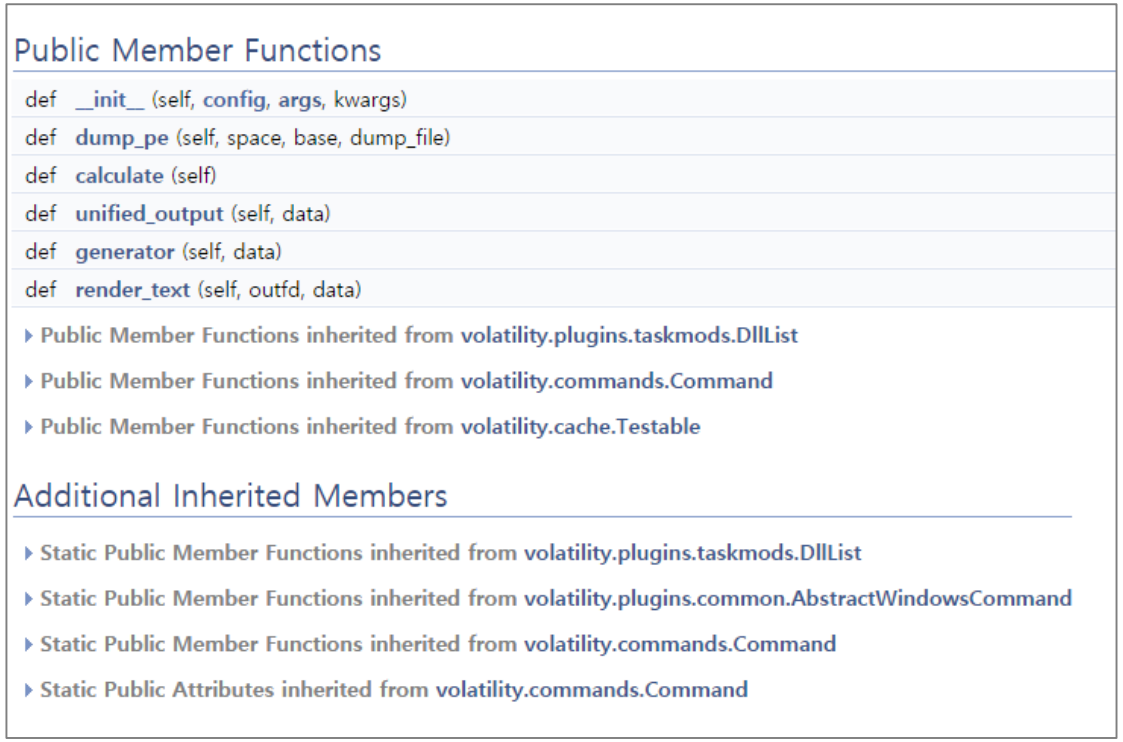
*Remarks*

- Saint Security Co., Ltd. owns the property rights of this document. This document is protected by copyright. Therefore, without a written permission, it is illegal to extract contents all or part or to copy it any form.
- The document is subject to change without notice in case of function improvement and error correction.
- Saint Security Co., Ltd. will not be liable for any damage or loss which can occur directly or indirectly by this document and its contents.

SAINT SECURITY

GLOBAL SECURITY SOLUTION PROVIDER

*Saint Security, Co., LTD.*
*Copyright 2014 Saint Security, Inc. All rights reserved.*
*Tel 82-2-704-7502*
*Fax 82-2-704-7508*
*www.stsc.com*
*www.malwares.com*



### Public Member Functions

- def __init__ (self, config, args, kwargs)
- def dump_pe (self, space, base, dump_file)
- def calculate (self)
- def unified_output (self, data)
- def generator (self, data)
- def render_text (self, outfd, data)

▸ Public Member Functions inherited from volatility.plugins.taskmods.DllList

▸ Public Member Functions inherited from volatility.commands.Command

▸ Public Member Functions inherited from volatility.cache.Testable

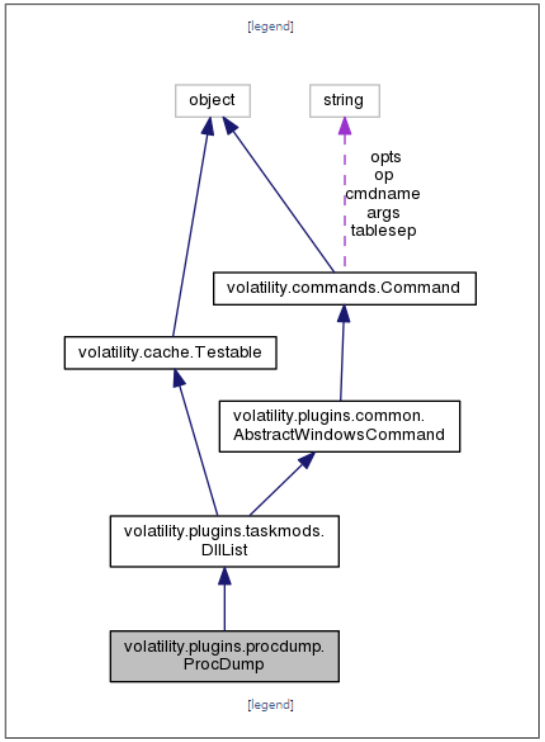
### Additional Inherited Members

▸ Static Public Member Functions inherited from volatility.plugins.taskmods.DllList

▸ Static Public Member Functions inherited from volatility.plugins.common.AbstractWindowsCommand

▸ Static Public Member Functions inherited from volatility.commands.Command

▸ Static Public Attributes inherited from volatility.commands.Command



```
def calculate(self):

    if self._config.DUMP_DIR == None:
        print "\n[!] Process to dump in the current directory."
        self._config.DUMP_DIR = os.getcwd()

    if self._config.PID != None:
        print "\n[+] To start a process dump.\n"
        result = procdump.ProcDump(self._config).execute()
        result2 = procdump.ProcDump.calculate(self)
        filepath = self._config.DUMP_DIR + "\executable.{0}.exe".format(self._config.PID)
        filename = "executable.{0}.exe".format(self._config.PID)
        copypath = self._config.DUMP_DIR + "\Volatility\plugins" + "\executable.{0}.exe".format(self._config.PID)

        shutil.copy(filepath, copypath)

        print "\n[+] Copying Dump File ..."
        print "[-] Copy Dump File Path : " + copypath

        return filename
```

**Get Current directory**

**Dump ₩Volatility₩Plugins₩executable.(PID).exe**

**Return Filename / executable.(PID).exe**

```
def render_text(self, outfd, data):

    print "[+] Upload File & File Analysis ..."

    params = {'api_key':malcom_key,'filename': data}
    files = {'file':(data,open(data,'rb'), 'application/octet-stream')}
    response = requests.post('https://www.malwares.com/api/v2/file/upload', files=files, data=params)
    json_response = response.json() # File_Upload

    md5 = json_response["md5"]

    params = {'api_key':malcom_key, 'hash':md5}
    response = requests.get('https://www.malwares.com/api/v2/file/mwsinfo', params=params)
    json_response = response.json() # File_Scan
```

**Get Filename**

**Upload malicious file**

**Upload MD5 & JSON Report**

**Malwares.com API Guide :** https://www.malwares.com/about/api

```
positives = avscan["positives"]
total = avscan["total"]
```

## JSON Report Parse

```
print "\n[!] Upload Information"
print "[-] Upload Result : " + result_msg
print "[-] Upload Date : " + date

print "\n[!] File Information"
print "[-] MD5 : " + md5
print "[-] SHA1 : " + sha1
print "[-] SHA256 : " + sha256

print "\n[!] File Advanced Information"
print "[-] View Count : " + str(view_count)
print "[-] First Seen : " + first_seen

if(black_white == 1):
    print "[-] Black & White List : Black List"

else:
    print "[-] Black & White List : White List"

print "[-] File Type : " + filetype
print "[-] File Size : " + str(filesize) + " Byte"
print "\n[!] AV Scan Result : " + str(positives) + " / " + str(total)
```

## Print File info

malwares.com™

| | | |
|---|---|---|
| -404 | No result | No result (try again after 5 minutes) |
| -500 | Internal Server Error | System error |

- Example

Call Code Example

```
import requests
params = {'api_key': 'API KEY', 'hash':
'94EAC5559220793377C3F3B791AA81DB53DEEE34D21467D70799A32E8BD4BD51'}
response = requests.get('https://www.malwares.com/api/v2/file/behaviorinfo', params=params)
json_response = response.json()
```

Call Result Example


```
{
  "os_env": "Microsoft Windows XP Professional Service Pack 3 (KOR)",
  "installed_program": "Microsoft Office Professional 2010 (14.0.4763.1000) / Windows Internet Explorer (8.0.6001.18702)
/ Hangul 2010 (8.5.8.1232)/ Adobe Reader (9.0.0) / Adobe Flash Player 11 Active X (10.10) / Java(TM) (7.0.0)",
  "sha1": "0A34A5C547AADEA85CF48D0126FB68AC49DB4C42",
  "network": {
    "session": [],
    "flow": []
  },
  "result_msg": "Data exists",
  "hosts_file": [],
  "start": 1443771213,
  "version": "20150514",
  "behavior": [
    {
      "path": "<MWS_SAMPLE_DIR>₩mnzB06.exe",
      "pid": 1824,
      "ppid": 1352,
      "order": 1,
      "process_open_event": [],
      "file_information_query_event": [],
      "dll_load_event": [],
      "filehandle_create_event": [],
      "file_find_event": [],
      "file_write_event": [],
      "file_open_event": [],
      "reg_key_open_event": [],
      "file_read_event": [],
      "reg_key_create_event": [],
      "reg_key_query_event": [],
      "reg_value_query_event": []
    }
  ],
  "security_level": 3,
  "date": "2015-10-02 16:34:45",
  "sha256": "94EAC5559220793377C3F3B791AA81DB53DEEE34D21467D70799A32E8BD4BD51",
  "result_code": 1,
  "md5": "92E04BCF92CF58BF434393D0B3B6BCA2"
}
```

# Plugin Execution Result

**Command :** python vol.py –f memdump.img –-profile=Win7SP0x86 **malcom –p 2200**

```
C:₩Users₩DongHyun₩Desktop₩vol₩vol-m>python vol.py -f memdump.img --profile=Win7SP0x86 malcom -p 2200

Volatility Foundation Volatility Framework 2.5

[!] Process to dump in the current directory.

[+] To start a process dump.

Process(V) ImageBase  Name                 Result
---------- ---------- -------------------- ------
0x8270a198 0x00400000 malcom (2            OK: executable.2200.exe

[+] Copying Dump File ...
[-] Copy Dump File Path : C:₩Users₩DongHyun₩Desktop₩vol₩vol-m₩Volatility₩plugins₩executable.2200.exe

[+] Upload File & File Analysis ...
INFO    : requests.packages.urllib3.connectionpool: Starting new HTTPS connection (1): www.malwares.
com
INFO    : requests.packages.urllib3.connectionpool: Starting new HTTPS connection (1): www.malwares.
com

[!] Upload Information
[-] Upload Result : Data exists
[-] Upload Date : 2016-04-17 12:55:30

[!] File Information
[-] MD5 : C93B4FB12324303AEC8AE425B8BAA7DF
[-] SHA1 : B8BD49DDAB5FAB0E358214F26ADFAF1C7686CF63
[-] SHA256 : 45493FCBE3205D17EB93D6A7622363158A7A61DC0BA0ED874D6CC10CA5A5F407

[!] File Advanced Information
[-] View Count : 0
[-] First Seen : 2016-04-17 05:15:12
[-] Black & White List : White List
[-] File Type : exe_32bit
[-] File Size : 76800 Byte

[!] AV Scan Result : 1 / 57
```

# Case Study #2

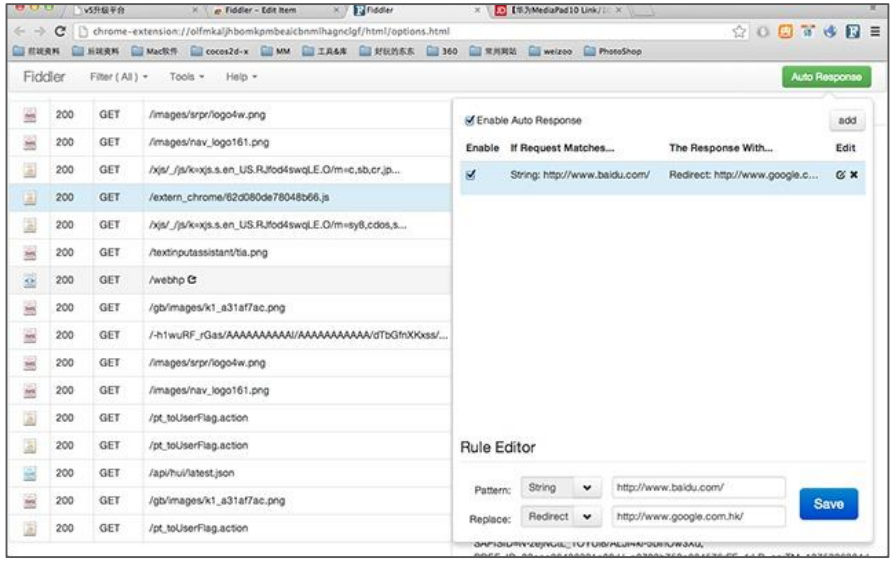
Facebook Account / Password Leak Plugin – MAJ3STY

Chrome

Internet Explorer

Firefox

Facebook



```
153A98B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
153A98C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
153A98D0  00 00 00 00 48 0C 84 20 B7 23 3E 11 98 06 00 80  ....H.„ ·#>.˜..€
153A98E0  6C 73 64 3D 41 56 71 66 50 33 5F 48 26 65 6D 61  lsd=AVqfP3_H&ema
153A98F0  69 6C 3D 64 69 67 69 74 61 6C 69 73 78 39 39 40  il=digitalisx99@
153A9900  67 6D 61 69 6C 2E 63 6F 6D 26 70 61 73 73 3D 64  gmail.com&pass=d
153A9910  6E 66 6C 73 6B 66 6B 31 32 33 34 26 70 65 72 73  [redacted]&pers
153A9920  69 73 74 65 6E 74 3D 26 64 65 66 61 75 6C 74 5F  istent=&default_
153A9930  70 65 72 73 69 73 74 65 6E 74 3D 31 26 74 69 6D  persistent=1&tim
153A9940  65 7A 6F 6E 65 3D 2D 35 34 30 26 6C 67 6E 64 69  ezone=-540&lgndi
153A9950  6D 3D 65 79 4A 33 49 6A 6F 78 4F 54 49 77 4C 43  m=eyJ3IjoxOTIwLC
```

**&email=digitalisx99@gmail.com&pass=diskmemory**

**&email=** **&pass=**

```
import volatility.utils as utils
import volatility.commands as commands
import volatility.win32.tasks as tasks
from time import time
from urllib2 import unquote
```

Import Volatility API

Set Plugin help & option

-p = PID, -s = Site Name

```
site_list = ['facebook', 'google', 'daum']
class userInfo(commands.Command):
    """The plugin user ID and password in the memory image acquisition plugin."""
    def __init__(self, config, *args, **kwargs):
        commands.Command.__init__(self, config, *args, **kwargs)
        config.add_option('PID', short_option = 'p', default = None, help='Brows
        config.add_option('SITE', short_option = 's', default = None, help='You
```

```python
def render_text(self, outfd, data):
    startTime = time()
    outfd.write('[!] Searching UserInfo in Memory Image... Wait!!\n')
    for proc in data:
        if not self._config.PID == None and str(proc.UniqueProcessId) not in list(self._config.PID.split(',')):
            continue
        outfd.write("\n[+] Found Browser Process(PID) : {0}({1})\n".format(proc.ImageFileName, proc.UniqueProcessId))
        for vad, process_space in proc.get_vads():
            start = vad.Start
            offset = vad.Length
            processData = process_space.zread(start, offset)
            if processData == None:
                if self._config.verbose:
                    outfd.write('[!] Memory Vad Range {0} ~ {1} Not Accessible\n'.format(start, start+offset))
            else:
                self.Text_table(outfd, processData, start, offset, 'facebook')
                self.Text_table(outfd, processData, start, offset, 'google')

    endTime = time()
    outfd.write("[!] Time : {0}\n\n".format(endTime-startTime))
```

**Check the Parameter inserted user**

**Get VAD Info**

```python
def Text_table(self, outfd, procData, vad_start, vad_length, site):
    for userId, userPw in self.Userinfo(procData, site):
        outfd.write(" [-] Vad Address Range : {0} ~ {1}\n".format(vad_start, vad_start+vad_length))
        outfd.write(" [-] {0} User Email : {1}\n".format(site, userId))
        outfd.write(" [-] {0} User Pass  : {1}\n\n".format(site, userPw))
```

**Print Data**

```python
def parse_data(self, procData, email, pw, parse_end):
    userInfo = procData[procData.find(email):procData.find(parse_end)]
    if userInfo == '':
        pass
    else:
        userId = userInfo[userInfo.find(email)+len(email):userInfo.find(pw)]
        userPw = userInfo[userInfo.find(pw)+len(pw):]
        yield userId, userPw
```

**Parse Data**

```python
def Userinfo(self, procData, site):
    if site == 'facebook':
        for userId, userPw in self.parse_data(procData, '&email=', '&pass=', '&default_persistent='):
            yield userId, userPw

    elif site == 'google':
        for userId, userPw in self.parse_data(procData, '&Email=', '&Passwd=', '&signIn='):
            yield userId, unquote(userPw)
```

**Searching Data in VAD**

```
.................
.................
....H.„ ·#>.˜..€
lsd=AVqfP3_H&ema
il=digitalisx99@
gmail.com&pass=d
[redacted]&pers
istent=&default_
persistent=1&tim
ezone=-540&lgndi
m=eyJ3IjoxOTIwLC
```

**Get Info**

# Plugin Execution Result

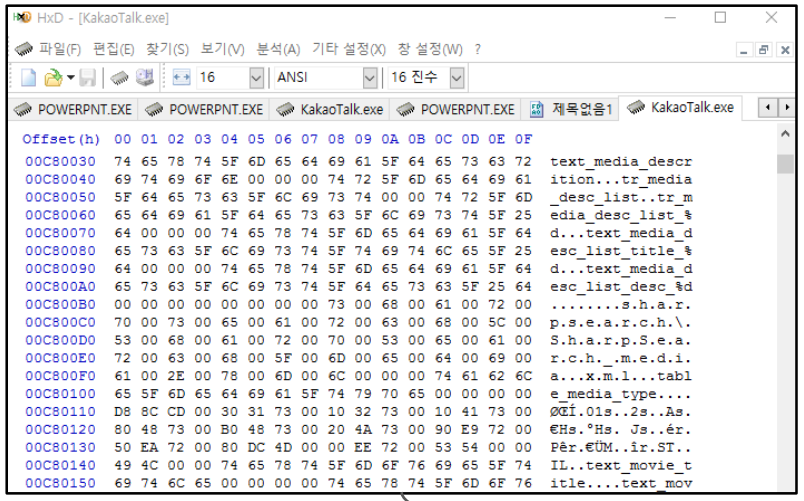
**Command :** python vol.py –f memdump.img -–profile=Win7SP0x86 **userinfo –p 3632**

```
root@kali:/usr/share/volatility/volatility/plugins# vol --profile=Win7SP0x86 -f ~/Desktop/pluginTest.vmem userinfo -p 3632
Volatile Systems Volatility Framework 2.1
[!] Searching UserInfo in Memory Image... Wait!!

[+] Found Browser Process(PID) : iexplore.exe(3632)
 [-] Vad Address Range : 50790400 ~ 51838976
 [-] facebook User Email : facebookUserinfo@naver.com
 [-] facebook User Pass  : MaJ3stY

 [-] Vad Address Range : 89128960 ~ 91226112
 [-] google User Email : googleUserinfo@naver.com
 [-] google User Pass  : MaJ3stY!!@@

 [-] Vad Address Range : 98893824 ~ 103088128
 [-] facebook User Email : facebookUserinfo@naver.com
 [-] facebook User Pass  : MaJ3stY

 [-] Vad Address Range : 98893824 ~ 103088128
 [-] google User Email : googleUserinfo@naver.com
 [-] google User Pass  : MaJ3stY!!@@

[!] Time : 1.18268704414
```

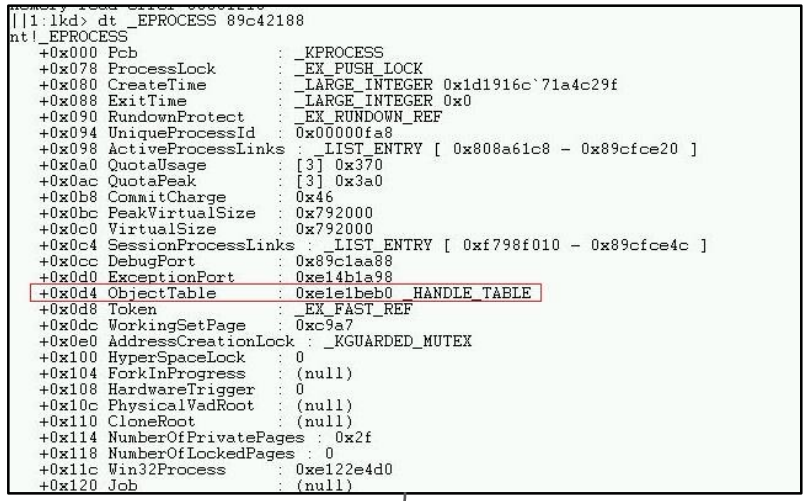
# Development Tip

Personal experience

# Based on String

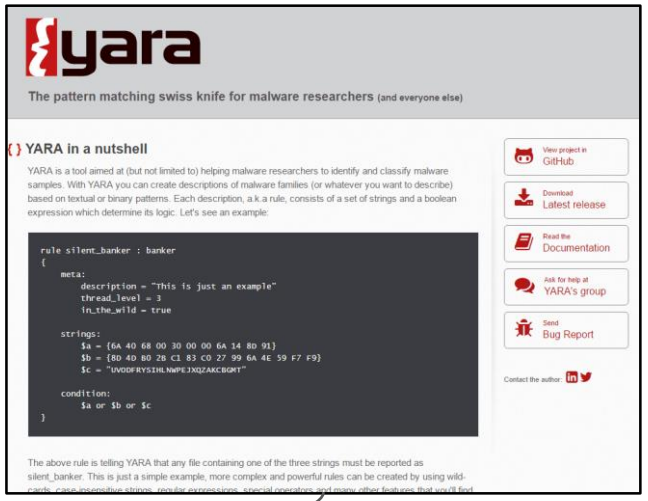



# Memory Struct



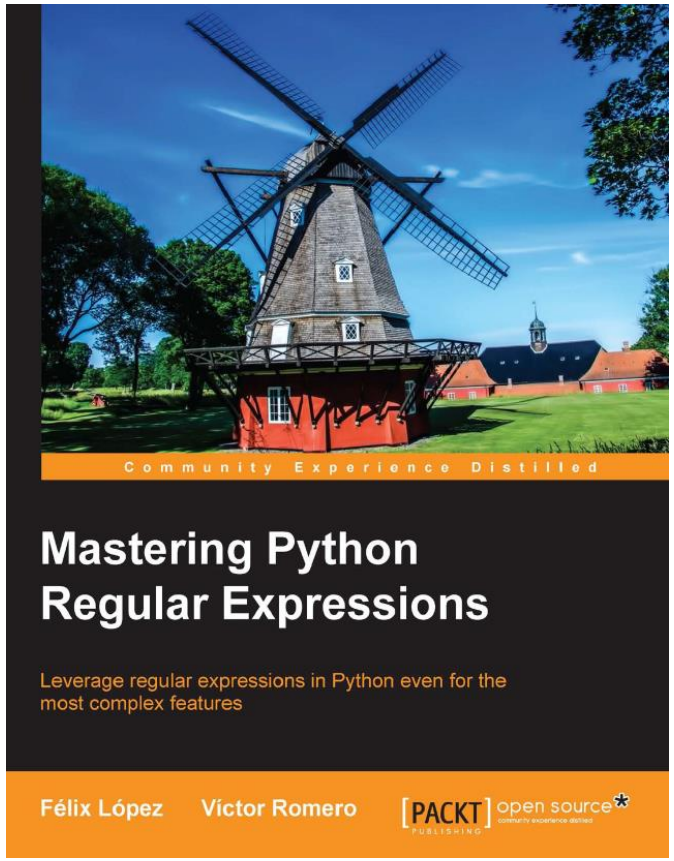
```
[|1:lkd> dt _EPROCESS 89c42188
nt!_EPROCESS
   +0x000 Pcb              : _KPROCESS
   +0x078 ProcessLock      : _EX_PUSH_LOCK
   +0x080 CreateTime       : _LARGE_INTEGER 0x1d1916c`71a4c29f
   +0x088 ExitTime         : _LARGE_INTEGER 0x0
   +0x090 RundownProtect   : _EX_RUNDOWN_REF
   +0x094 UniqueProcessId  : 0x00000fa8
   +0x098 ActiveProcessLinks : _LIST_ENTRY [ 0x808a61c8 - 0x89cfce20 ]
   +0x0a0 QuotaUsage       : [3] 0x370
   +0x0ac QuotaPeak        : [3] 0x3a0
   +0x0b8 CommitCharge     : 0x46
   +0x0bc PeakVirtualSize  : 0x792000
   +0x0c0 VirtualSize      : 0x792000
   +0x0c4 SessionProcessLinks : _LIST_ENTRY [ 0xf798f010 - 0x89cfce4c ]
   +0x0cc DebugPort        : 0x89c1aa88
   +0x0d0 ExceptionPort    : 0xe14b1a98
   +0x0d4 ObjectTable      : 0xe1e1beb0 _HANDLE_TABLE
   +0x0d8 Token            : _EX_FAST_REF
   +0x0dc WorkingSetPage   : 0xc9a7
   +0x0e0 AddressCreationLock : _KGUARDED_MUTEX
   +0x100 HyperSpaceLock   : 0
   +0x104 ForkInProgress   : (null)
   +0x108 HardwareTrigger  : 0
   +0x10c PhysicalVadRoot  : (null)
   +0x110 CloneRoot        : (null)
   +0x114 NumberOfPrivatePages : 0x2f
   +0x118 NumberOfLockedPages : 0
   +0x11c Win32Process     : 0xe122e4d0
   +0x120 Job              : (null)
```

# Other Source




# Volatility Plugins

# Use Regular Expressions

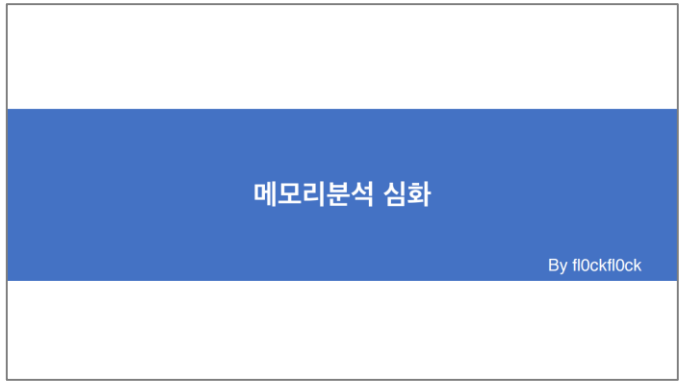



**Download** : https://www.facebook.com/download/preview/172047083216705

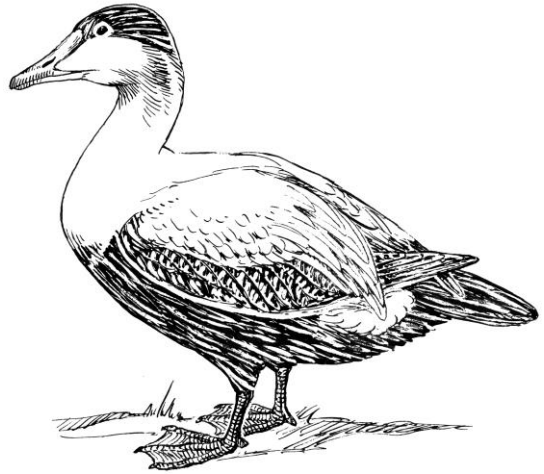
- **Parsing and Carving :** https://www.facebook.com/download/preview/1219613874756729
- **Regexr (Regular Expressions Practice) :** http://regexr.com/
- **CyberChef (Encode & Decode) :** https://gchq.github.io/CyberChef/
- **HxD (Hex Editor) :** https://mh-nexus.de/en/hxd/

# Use Kyle Choi`s Presentation (Volshell)

- **Malware Analysis :** https://www.facebook.com/download/preview/1075446245906624
- **Deepening Memory Analysis :** https://www.facebook.com/download/preview/1590791651220823
- **TRIAGE :** https://www.facebook.com/download/preview/1188700827828227

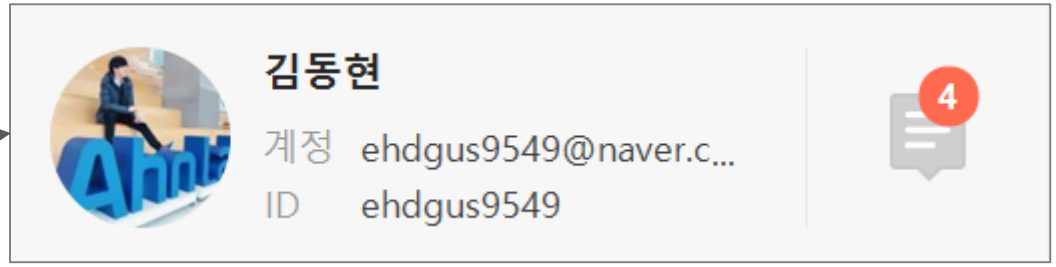






# Volakao - My Profile

```
0BE9B5F0  48 54 54 50 2F 31 2E 31 20 32 30 30 20 4F 4B 0D  HTTP/1.1 200 OK.
0BE9B600  0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61  .Content-Type: a
0BE9B610  70 70 6C 69 63 61 74 69 6F 6E 2F 6A 73 6F 6E 3B  pplication/json;
0BE9B620  20 63 68 61 72 73 65 74 3D 75 74 66 2D 38 0D 0A   charset=utf-8..
0BE9B630  54 72 61 6E 73 66 65 72 2D 45 6E 63 6F 64 69 6E  Transfer-Encodin
0BE9B640  67 3A 20 63 68 75 6E 6B 65 64 0D 0A 43 6F 6E 6E  g: chunked..Conn
0BE9B650  65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 4B  ection: close..K
0BE9B660  61 6B 61 6F 3A 20 54 61 6C 6B 0D 0A 43 61 63 68  akao: Talk..Cach
0BE9B670  65 2D 43 6F 6E 74 72 6F 6C 3A 20 6E 6F 2D 63 61  e-Control: no-ca
0BE9B680  63 68 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D  che..Pragma: no-
0BE9B690  63 61 63 68 65 0D 0A 0D 0A 61 30 39 0D 0A 7B 22  cache....a09..{"
0BE9B6A0  65 6D 61 69 6C 53 74 61 74 75 73 22 3A 31 2C 22  emailStatus":1,"
0BE9B6B0  65 6D 61 69 6C 41 64 64 72 65 73 73 22 3A 22 65  emailAddress":"e
0BE9B6C0  68 64 67 75 73 39 35 34 39 40 6E 61 76 65 72 2E  hdgus9549@naver.
0BE9B6D0  63 6F 6D 22 2C 22 65 6D 61 69 6C 56 65 72 69 66  com","emailVerif
```




```
0BE9BC00  46 4B 2F 66 39 63 64 6E 39 5F 31 31 30 78 31 31  FK/f9cdn9_110x11
0BE9BC10  30 5F 63 2E 6A 70 67 22 2C 22 66 75 6C 6C 50 72  0_c.jpg","fullPr
0BE9BC20  6F 66 69 6C 65 49 6D 61 67 65 55 72 6C 22 3A 22  ofileImageUrl":"
0BE9BC30  68 74 74 70 3A 2F 2F 74 68 2D 70 2E 74 61 6C 6B  http://th-p.talk
0BE9BC40  2E 6B 61 6B 61 6F 2E 63 6F 2E 6B 72 2F 74 68 2F  .kakao.co.kr/th/
0BE9BC50  74 61 6C 6B 70 2F 77 6B 76 4F 51 6A 72 73 66 45  talkp/wkvOQjrsfE
0BE9BC60  2F 4D 6E 4B 36 68 65 6D 68 6D 70 50 68 6C 77 59  /MnK6hemhmpPhlwY
0BE9BC70  6E 4B 61 38 4F 46 4B 2F 66 39 63 64 6E 39 5F 36  nKa8OFK/f9cdn9_6
0BE9BC80  34 30 78 36 34 30 5F 73 2E 6A 70 67 22 2C 22 6F  40x640_s.jpg","o
0BE9BC90  72 69 67 69 6E 61 6C 50 72 6F 66 69 6C 65 49 6D  riginalProfileIm
0BE9BCA0  61 67 65 55 72 6C 22 3A 22 68 74 74 70 3A 2F 2F  ageUrl":"http://
0BE9BCB0  70 2E 74 61 6C 6B 2E 6B 61 6B 61 6F 2E 63 6F 2E  p.talk.kakao.co.
0BE9BCC0  6B 72 2F 74 61 6C 6B 70 2F 77 6B 76 4F 51 6A 72  kr/talkp/wkvOQjr
0BE9BCD0  73 66 45 2F 4D 6E 4B 36 68 65 6D 68 6D 70 50 68  sfE/MnK6hemhmpPh
0BE9BCE0  6C 77 59 6E 4B 61 38 4F 46 4B 2F 66 39 63 64 6E  lwYnKa8OFK/f9cdn
```

# Volakao - Friend Profile

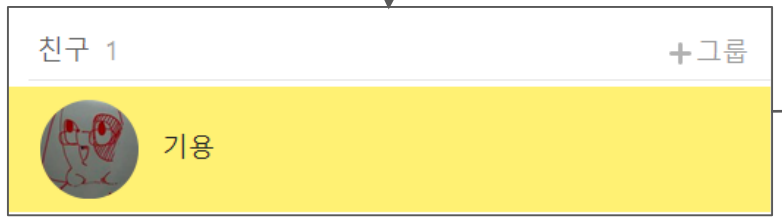
**UTF-8-decoded:**

기용

**UTF-8-encoded: ( permalink )**

\xEA\xB8\xB0\xEC\x9A\xA9

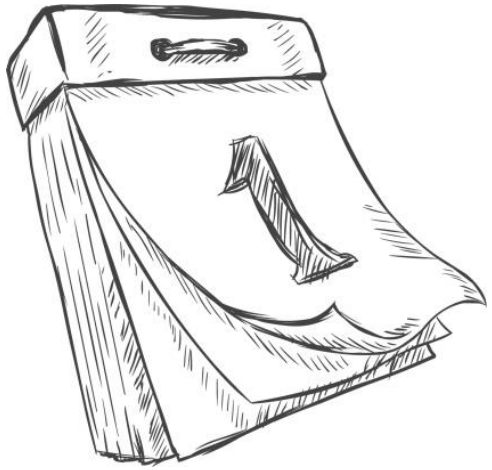
```
0BF0B640  08 0D 08 08 09 04 04 08 08 0C 03 93 AD 02 EA B8  ............".."ê¸
0BF0B650  B0 EC 9A A9 68 74 74 70 3A 2F 2F 74 68 2D 70 2E  °ìšªhttp://th-p.
0BF0B660  74 61 6C 6B 2E 6B 61 6B 61 6F 2E 63 6F 2E 6B 72  talk.kakao.co.kr
0BF0B670  2F 74 68 2F 74 61 6C 6B 70 2F 77 6B 76 4B 5A 41  /th/talkp/wkvKZA
0BF0B680  43 41 56 46 2F 30 74 65 57 6B 76 6A 4B 42 51 48  CAVF/0teWkvjKBQH
0BF0B690  6E 79 78 4C 62 4A 44 35 56 43 6B 2F 65 75 79 75  nyxLbJD5VCk/euyu
0BF0B6A0  7A 70 5F 31 31 30 78 31 31 30 5F 63 2E 6A 70 67  zp_110x110_c.jpg
0BF0B6B0  68 74 74 70 3A 2F 2F 74 68 2D 70 2E 74 61 6C 6B  http://th-p.talk
0BF0B6C0  2E 6B 61 6B 61 6F 2E 63 6F 2E 6B 72 2F 74 68 2F  .kakao.co.kr/th/
0BF0B6D0  74 61 6C 6B 70 2F 77 6B 76 4B 5A 41 43 41 56 46  talkp/wkvKZACAVF
0BF0B6E0  2F 30 74 65 57 6B 76 6A 4B 42 51 48 6E 79 78 4C  /0teWkvjKBQHnyxL
0BF0B6F0  62 4A 44 35 56 43 6B 2F 65 75 79 75 7A 70 5F 36  bJD5VCk/euyuzp_6
0BF0B700  34 30 78 36 34 30 5F 73 2E 6A 70 67 68 74 74 70  40x640_s.jpghttp
0BF0B710  3A 2F 2F 70 2E 74 61 6C 6B 2E 6B 61 6B 61 6F 2E  ://p.talk.kakao.
0BF0B720  63 6F 2E 6B 72 2F 74 68 2F 74 61 6C 6B 70 2F 77  co.kr/th/talkp/w
0BF0B730  6B 76 4B 5A 41 43 41 56 46 2F 30 74 65 57 6B 76  kvKZACAVF/0teWkv
0BF0B740  6A 4B 42 51 48 6E 79 78 4C 62 4A 44 35 56 43 6B  jKBQHnyxLbJD5VCk
0BF0B750  2F 65 75 79 75 7A 70 2E 6A 70 67 73 74 6F 72 79  /euyuzp.jpgstory
0BF0B760  67 6F 64 6C 6F 76 65 35 30 39 36 7B 7D 50 D9 3A  godlove5096{}PÙ:
0BF0B770  58 6C ED E0 58 6C ED E0 78 B3 F0 0B 94 B7 F0 0B  XlíàXlíàx³ð.”·ð.
```

# This is Secret!

# Plugin development Tip for homework (7 Days version)

- Try to imitate, it is very hard to modify existing code.
- Think Windows-based plug-in concept
- Ask a question to an expert (Maj3sty)
- **Try to repeat constantly.**



# Epilogue

Finishing the story


"I saw the angel in the marble and carved until
I set him free, The best artist has that thought
alone Which is contained within the marble
shell" - Michelangelo

*“I saw signs of hope in a lot of data.*

*The true Forensic analyst is the one who*

*will find its true value.”*

# Thank you

digitalisx99@gmail.com, Facebook Messenger