Pemanfaatan *Volatility* dalam Analisis Forensik Memori Digital (Studi Kasus: *Reminiscent Hackthebox*)

Yulian Sani
y.sani@aol.com

I. Pengantar

Forensik Memori (*Memory Forensics*) merupakan teknik analisis yang bertujuan untuk mengekstrak dan menginterpretasikan struktur data internal sistem operasi dari *memory dump*, sehingga memungkinkan identifikasi berbagai artefak sistem, seperti proses dan *thread* (Schuster, 2006). Forensik memori merupakan cabang dari Forensik Digital (*Digital Forensic*) yang fokus pada analisis isi RAM (*Random Access Memory*) untuk menemukan bukti aktivitas sistem yang sedang berjalan atau baru saja terjadi. Forensik memori merupakan teknik yang dilakukan dengan mengakuisisi dan menganalisis data volatil yang tersimpan di memori (RAM) yang akan hilang ketika sistem dimatikan, sehingga memiliki peran krusial dalam investigasi keamanan siber karena mampu mengungkap artefak penting yang tidak tercatat pada media penyimpanan permanen seperti *hard disk* atau media penyimpanan lainnya. Pendekatan forensik memori memiliki relevansi dalam investigasi keamanan siber, misalnya pada kasus serangan *malware*, proses *incident response*, investigasi ancaman dan kejadian dari dalam organisasi (*insider threat*), serta *Advanced Persistent Threat* (APT).

Saat ini, banyak teknik serangan modern dirancang untuk meminimalkan jejak pada media penyimpanan permanen. Akibatnya, aktivitas penyerang sering kali hanya dapat diidentifikasi melalui analisis memori, seperti pemeriksaan proses yang sedang berjalan, koneksi jaringan aktif, serta keberadaan kredensial sensitif yang tersimpan sementara di dalam RAM. Berdasarkan hal ini, forensik memori mampu memberikan visibilitas yang lebih komprehensif terhadap kondisi aktual sistem serta menjadi komponen penting dalam mendukung proses investigasi forensik digital.

Adapun jenis informasi atau artefak yang dapat diidentifikasi dan diekstraksi dari memori RAM selama proses forensik memori, misalnya:

1. **Proses yang Sedang Berjalan (*Running Processes*)**
   Forensik memori memungkinkan identifikasi seluruh proses yang sedang berjalan pada sistem, termasuk proses yang tidak terlihat oleh aplikasi standar (*hidden processes*). Hal ini penting untuk mendeteksi teknik penyembunyian, misalnya *malware*, *rootkit* atau proses yang di-*unlink* dari struktur sistem operasi. Informasi yang bisa diperoleh meliputi:
   - PID (*Process ID*)
   - Nama proses
   - Waktu eksekusi

   Informasi mengenai proses yang sedang berjalan dapat dimanfaatkan untuk mengidentifikasi indikasi *malware* yang menyamarkan sebagai proses yang sah (*legitimate process*). Contoh pada aktifitas ini misalnya *malware* yang menggunakan nama proses sistem yang sah seperti *svchost.exe* untuk menyamarkan aktivitasnya dan menghindari deteksi.

2. **Koneksi Jaringan (*Network Connections*)**
   Dari RAM, dapat dilihat koneksi jaringan yang aktif maupun yang baru saja terjadi. Artefak ini sangat penting dalam mengidentifikasi komunikasi dengan *source* attacker (*Command & Control* / C2). Informasi yang bisa didapat:
   - Alamat IP lokal dan *remote*
   - *Port* yang digunakan
   - Status koneksi (*ESTABLISHED*, *LISTENING*)
   - Proses yang membuka koneksi

   Informasi yang diperoleh dari koneksi jaringan dapat dimanfaatkan untuk mengidentifikasi indikasi eksfiltrasi data serta aktivitas komunikasi backdoor dengan server milik penyerang.

3. **DLL & Modul (*Loaded* DLLs & *Modules*)**
   Analisis terhadap library atau modul (DLL/*module*) yang dimuat oleh suatu proses dapat digunakan untuk mengidentifikasi indikasi injeksi kode berbahaya. Melalui analisis *file* tersebut, investigator dapat melihat:
   - Nama dan lokasi DLL
   - Validitas atau legitimasi DLL
   - Anomali, seperti DLL yang dijalankan dari path yang tidak wajar

   Analisis terhadap modul tersebut memungkinkan identifikasi teknik injeksi kode berbahaya, seperti *DLL Injection* maupun *Reflective DLL Injection*.

4. **Kredensial Pengguna (*User Credentials*)**
   Salah satu artefak sensitif yang dapat ditemukan melalui analisis RAM adalah kredensial pengguna (misalnya *username* dan *password*), yang tersimpan sementara di dalam memori selama sistem atau aplikasi sedang berjalan. *User credentials* dapat dimanfaatkan oleh penyerang dalam melakukan teknik *credential dumping* guna mendukung pergerakan lateral (*lateral movement*) dalam sistem.

   Pada sistem seperti Windows, proses seperti *lsass.exe* biasanya menyimpan:
   - *Hash password* (NTLM, Kerberos)
   - Token autentikasi

5. **Artefak Malware (*Malware Artifacts*)**
   Teknologi forensik memori dapat berperan untuk mendeteksi *malware* yang beroperasi tanpa meninggalkan jejak pada media penyimpanan (*fileless*). Beberapa artefak yang dapat diinvestigasi melalui analisis memori meliputi:
   - *Shellcode*
   - *Injected code* dalam proses *legitimate*
   - *Reflective* DLL
   - *Suspicious memory region* (RWX *permission*)

Penggunaan tools seperti *Volatility* dapat membantu mengidentifikasi pola aktivitas mencurigakan yang mengindikasikan keberadaan malware.

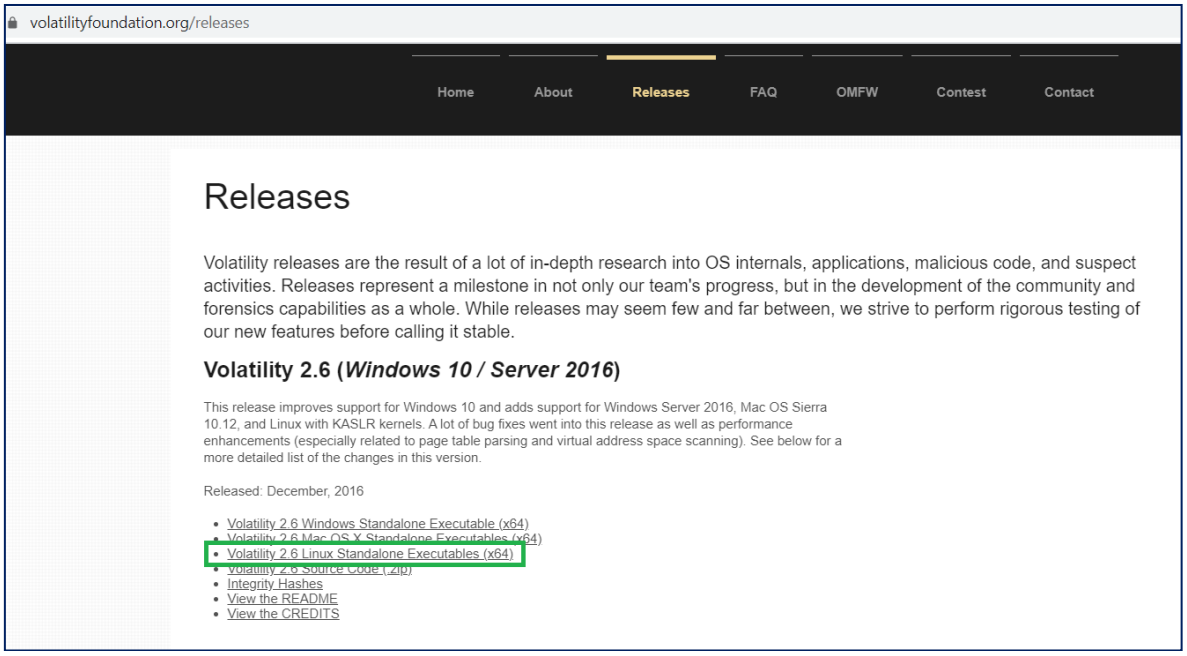
II. *Volatility*

Volatility pertama kali dikembangkan oleh Aaron Walters melalui penelitian akademis di bidang forensik memori. *Volatility* merupakan sebuah *framework open-source* yang digunakan dalam proses forensik memori untuk mendukung kegiatan *incident response* dan analisis *malware*, dibangun menggunakan bahasa pemrograman *Python*, dan saat ini telah mendukung berbagai sistem operasi, termasuk Microsoft Windows, macOS, dan Linux.

Volatility mampu mengekstraksi berbagai informasi yang tersimpan dalam memori sistem, seperti proses yang berjalan, koneksi dan *socket* jaringan, file *Dynamic Link Library* (DLL), serta *registry* pada sistem operasi. Selain itu, *Volatility* juga dapat digunakan untuk memperoleh informasi dari berkas *crash dump* maupun berkas *hibernation* sistem operasi. Dengan kemampuan tersebut, *Volatility* menjadi salah satu *tools* yang efektif dalam melakukan investigasi forensik berbasis memori.

III. Instalasi *Volatility*

Volatility yang digunakan pada praktik investigasi ini adalah *Volatility 2.6 Linux Standalone Executables (x64)* yang dapat di-*download* melalui https://www.volatilityfoundation.org/releases.



Setelah download *Volatility* selesai, lakukan *unzip* dengan perintah berikut ini:

```
unzip volatility_2.6_lin64_standalone.zip
```

```
root@kali:~/ProgramFiles# unzip volatility_2.6_lin64_standalone.zip
Archive:  volatility_2.6_lin64_standalone.zip
   creating: volatility_2.6_lin64_standalone/
  inflating: volatility_2.6_lin64_standalone/AUTHORS.txt
  inflating: volatility_2.6_lin64_standalone/CREDITS.txt
  inflating: volatility_2.6_lin64_standalone/LEGAL.txt
  inflating: volatility_2.6_lin64_standalone/LICENSE.txt
  inflating: volatility_2.6_lin64_standalone/README.txt
  inflating: volatility_2.6_lin64_standalone/volatility_2.6_lin64_standalone
```

Selanjutnya ubah nama folder *volatility* berada, dalam hal ini penulis mengubah yang sebelumnya *volatility_2.6_lin64_standalone* menjadi *volatility_2.6*.

```
mv volatility_2.6_lin64_standalone volatility_2.6
```

```
root@kali:~/ProgramFiles# mv volatility_2.6_lin64_standalone volatility_2.6
```

Selain directory, agar mudah dieksekusi penulis juga mengubah nama *file executable volatility* yang sebelumnya *volatility_2.6_lin64_standalone* menjadi *volatility*.

```
cd volatility_2.6/
mv volatility_2.6_lin64_standalone volatility
```

```
root@kali:~/ProgramFiles# mv volatility_2.6_lin64_standalone volatility_2.6
root@kali:~/ProgramFiles# cd volatility_2.6/
root@kali:~/ProgramFiles/volatility_2.6# ls
AUTHORS.txt  CREDITS.txt  LEGAL.txt  LICENSE.txt  README.txt  volatility_2.6_lin64_standalone
root@kali:~/ProgramFiles/volatility_2.6# mv volatility_2.6_lin64_standalone volatility

root@kali:~/ProgramFiles/volatility_2.6# ls
AUTHORS.txt  CREDITS.txt  LEGAL.txt  LICENSE.txt  README.txt  volatility
```

```
file volatility
```

```
root@kali:~/ProgramFiles/volatility_2.6# file volatility
volatility: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=dcb7928ff5bfd3f1
de0f2f774950cd22212132c1, stripped
```

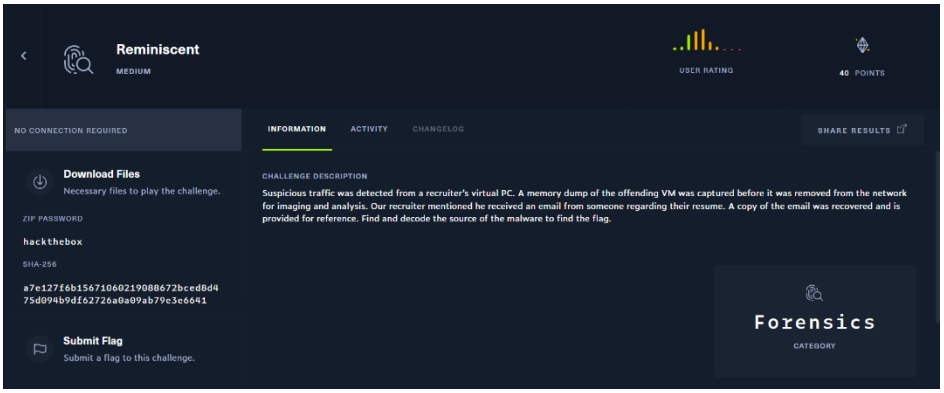
```
./volatility -h
```

```
root@kali:~/ProgramFiles/volatility_2.6# ./volatility -h
Volatility Foundation Volatility Framework 2.6
Usage: Volatility - A memory forensics analysis platform.

Options:
  -h, --help            list all available options and their default values.
                        Default values may be set in the configuration file
                        (/etc/volatilityrc)
  --conf-file=/root/.volatilityrc
                        User based configuration file
  -d, --debug           Debug volatility
  --plugins=PLUGINS     Additional plugin directories to use (colon separated)
  --info                Print information about all registered objects
  --cache-directory=/root/.cache/volatility
                        Directory where cache files are stored
  --cache               Use caching
  --tz=TZ               Sets the (Olson) timezone for displaying timestamps
                        using pytz (if installed) or tzset
  -f FILENAME, --filename=FILENAME
                        Filename to use when opening an image
  --profile=WinXPSP2x86
                        Name of the profile to load (use --info to see a list
                        of supported profiles)
```

IV. Simulasi Analisis *Reminiscent*

4.1. Ekstraksi File

1. File *reminiscent.zip* sebagai objek yang akan diinvestigasi dapat ditemukan pada menu ***Challenges ➔ Forensics*** melalui situs https://hackthebox.org atau dengan mengakses URL: https://app.hackthebox.com/challenges/Reminiscent setelah *user* melakukan *login*.




```
root@kali:~/Documents/Hackthebox/Reminiscent# md5sum reminiscent.zip && sha1sum reminiscent.zip
9ac38307490fb7614adaac524fc5a896  reminiscent.zip
9d9ce46beeb6bc3d2dda57b8d1b4c7810ffce63e  reminiscent.zip
```

2. Dari file *reminiscent.zip* tersebut, jika *extract* akan menghasilkan 3 *file* yaitu:

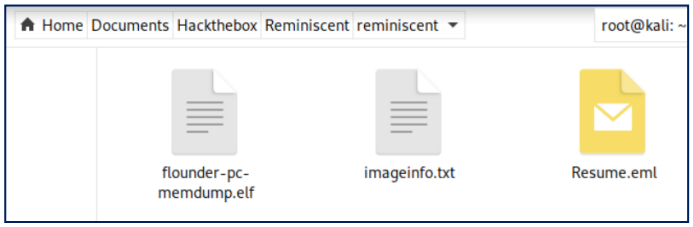
   a. *Resume.eml*

   File ini berisi pesan dari pemilik sistem yang meminta pengguna untuk melakukan peninjauan terhadap file "*resume*" yang tersimpan pada *capture file*.

```
MIME-Version: 1.0
Content-Type: multipart/alternative;
 boundary="=_a8ebc8b42c157d88c1096632aeae0559"
Date: Mon, 02 Oct 2017 22:30:24 -0400
From: Brian Loodworm <bloodworm@madlab.lcl>
To: flounder@madlab.lcl
Subject: Resume
Organization: HackTheBox
Message-ID: <add77ed2ac38c3ab639246956c25b2c2@madlab.lcl>
X-Sender: bloodworm@madlab.lcl
Received: from mail.madlab.lcl (HELO mail.madlab.lcl) (127.0.0.1)
 by mail.madlab.lcl (qpsmtpd/0.96) with ESMTPSA (ECDHE-RSA-AES256-GCM-SHA384 encrypted); Mon, 02 Oct 2017 22:30:24 -0400

--=_a8ebc8b42c157d88c1096632aeae0559
Content-Transfer-Encoding: 7bit
Content-Type: text/plain; charset=US-ASCII

Hi Frank, someone told me you would be great to review my resume..
Could you have a look?

resume.zip [1]

Links:
------
[1] http://10.10.99.55:8080/resume.zip
--=_a8ebc8b42c157d88c1096632aeae0559
Content-Transfer-Encoding: quoted-printable
Content-Type: text/html; charset=UTF-8
```

   b. *flounder-pc-memdump.elf*

   Merupakan berkas utama dari berkas *digital imaging* utama yang selanjutnya akan dianalisis untuk mencari *evidence* yang *valid*. Dalam skenario ini, evidence yang *valid* berupa *flag* atau kode yang harus di-*submit* pada *platform Hack The Box*.



   c. *imageinfo.txt*

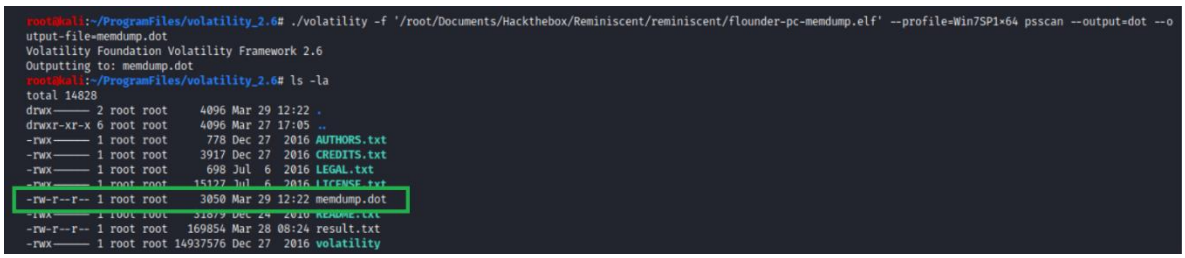
   Berisi informasi mengenai profil atau *properties* dari file *flounder-pc-memdump.elf*

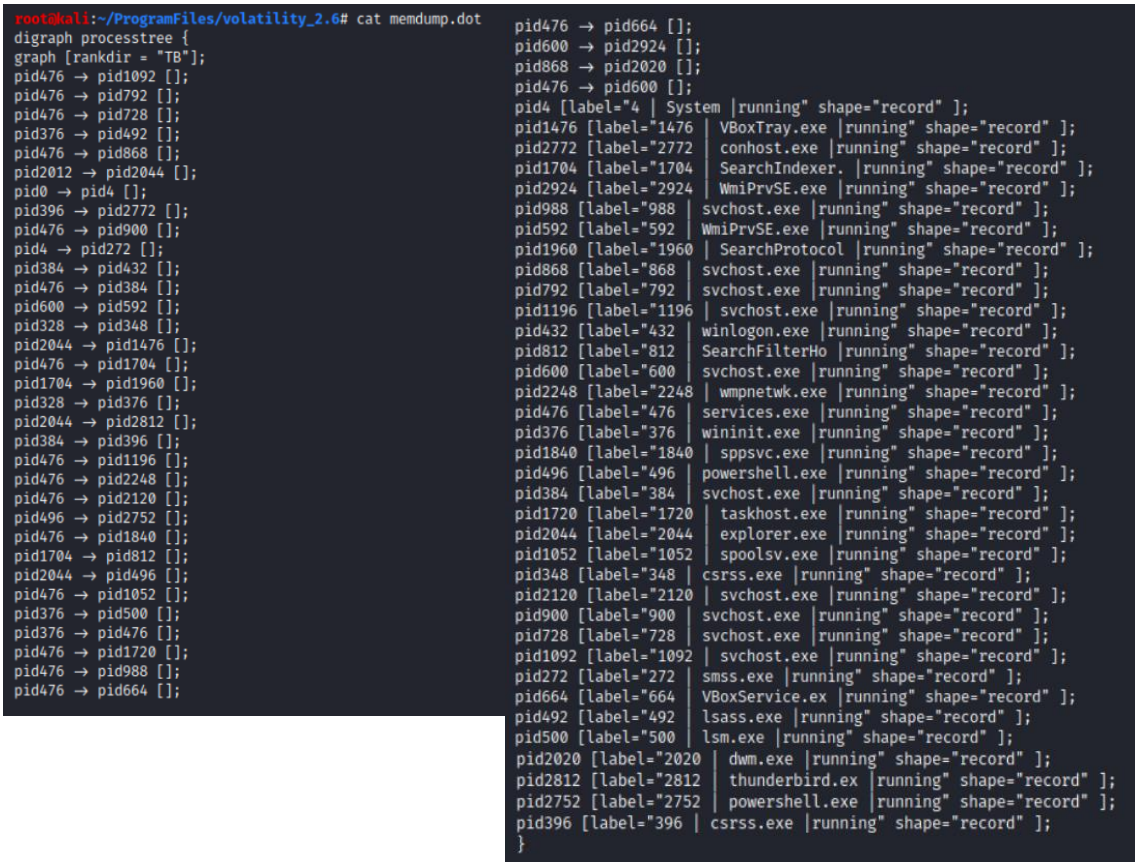
```
File  Edit  View  Search  Terminal  Help
          Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP1x64_23418
                     AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
                     AS Layer2 : VirtualBoxCoreDumpElf64 (Unnamed AS)
                     AS Layer3 : FileAddressSpace (/home/infosec/dumps/mem_dumps/01/flounder-pc-memdump.elf)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf800027fe0a0L
          Number of Processors : 2
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff800027ffd00L
                KPCR for CPU 1 : 0xfffff880009eb000L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2017-10-04 18:07:30 UTC+0000
     Image local date and time : 2017-10-04 11:07:30 -0700
```

4.2. Analisis File

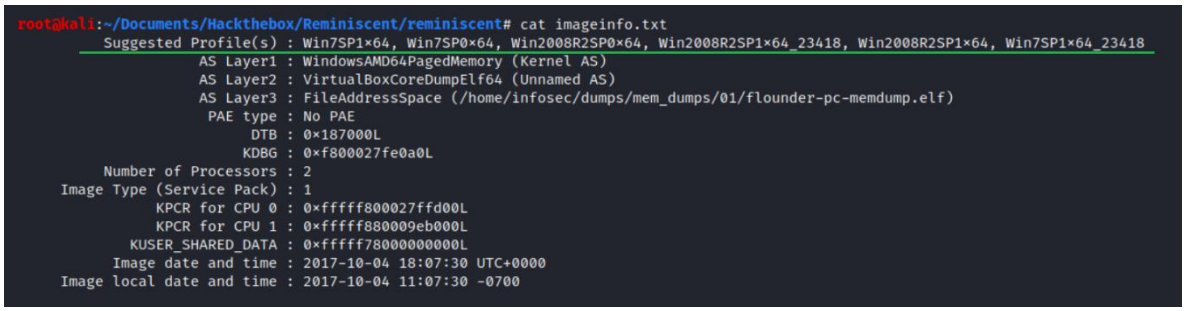
1. Berdasarkan *flounder-pc-memdump.elf* yang berisi daftar proses *memory*, dilakukan ekstraksi untuk menghasilkan *dump file* dengan nama file *memdump.dot*. Perintah yang digunakan adalah sebagai berikut:

```
./volatility -f
'/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-
memdump.elf' --profile=Win7SP1x64 psscan --output=dot --output-
file=memdump.dot
```



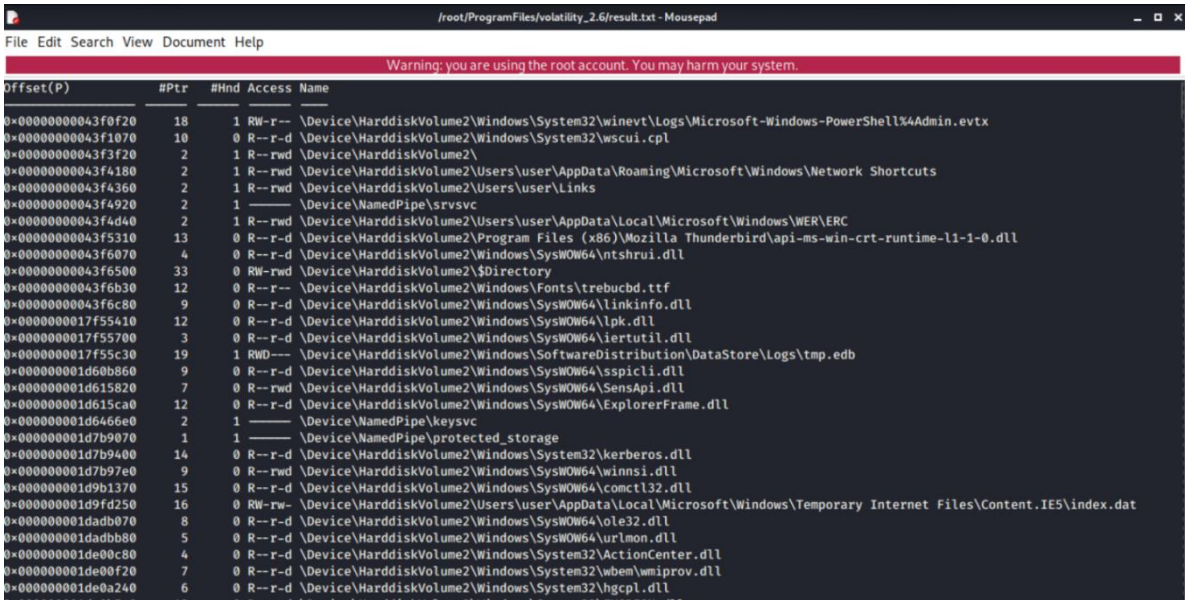


2. Selain itu berdasarkan informasi dari file *imageinfo.txt*, diperoleh informasi mengenai kemungkinan sistem operasi yang digunakan, yaitu: **Win7SP1x64**, **Win7SP0x64**, **Win2008R2SP0x64**, **Win2008R2SP1x64_23418**, **Win2008R2SP1x64**, atau **Win7SP1x64_23418**.



3. Selanjutnya dilakukan *filescan* terhadap *memdump.elf* untuk mendapatkan nilai *offset* dan aplikasi yang berjalan di dalam *background memory* sistem operasi tersebut.

```
./volatility -f
'/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-
memdump.elf' --profile=Win7SP1x64 filescan > result.txt
```



4. Berdasarkan *filescan* tersebut, diperoleh informasi yang memuat aplikasi atau berkas sistem yang sebelumnya sudah dijalankan. Dari aplikasi atau *file* sistem tersebut, dilakukan identifikasi untuk mendapatkan kemungkinan *file* yang mengandung anomali. Berdasarkan analisis ditemukan *file* yang kemungkinan mengandung pesan tersembunyi, yaitu: *resume.pdf.lnk.*

```
./volatility -f
'/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-
memdump.elf' --profile=Win7SP1x64 filescan | grep resume
```

```
root@kali:~/ProgramFiles/volatility_2.6# ./volatility -f '/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-memdump.elf' --profile=Win7SP1x64
filescan | grep resume
Volatility Foundation Volatility Framework 2.6
0x000000001e1f6200      1      0 R--r-- \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
0x000000001e8feb70      1      1 R--rw- \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
```

5. Selanjutnya investigator melakukan ekstraksi terhadap kedua *file* tersebut. Pada tahap awal investigator melakukan identifikasi dan analisis dari *file* pertama yang memiliki nilai offset 0x000000001e1f6200, namun dari hasil analisis pada *file* ini tidak ditemukan barang bukti yang *valid*.

```
./volatility -f
'/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-
memdump.elf' --profile=Win7SP1x64 dumpfiles -Q
0x000000001e1f6200 -D .
```

```
root@kali:~/ProgramFiles/volatility_2.6# ./volatility -f '/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-memdump.elf' --pr
dumpfiles -Q 0x000000001e1f6200 -D .
Volatility Foundation Volatility Framework 2.6
DataSectionObject 0x1e1f6200   None   \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
SharedCacheMap 0x1e1f6200   None   \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
```

6. Selanjutnya dilakukan identifikasi dan analisis terhadap berkas *resume.pdf.lnk* pada *offset* 0x000000001e8feb70.

```
./volatility -f
'/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-
memdump.elf' --profile=Win7SP1x64 dumpfiles -Q
0x000000001e8feb70 -D .
```

```
root@kali:~/ProgramFiles/volatility_2.6# ./volatility -f '/root/Documents/Hackthebox/Reminiscent/reminiscent/flounder-pc-memdump.elf' --profile=Win7SP1x64
dumpfiles -Q 0x000000001e8feb70 -D .
Volatility Foundation Volatility Framework 2.6
DataSectionObject 0x1e8feb70   None   \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
SharedCacheMap 0x1e8feb70   None   \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
```

Dari hasil ekstraksi *file* menggunakan perintah diatas akan membentuk 2 *file* baru yaitu:

a) file.None.0xfffffa80017dcc60.vacb
b) file.None.0xfffffa80022ac740.dat

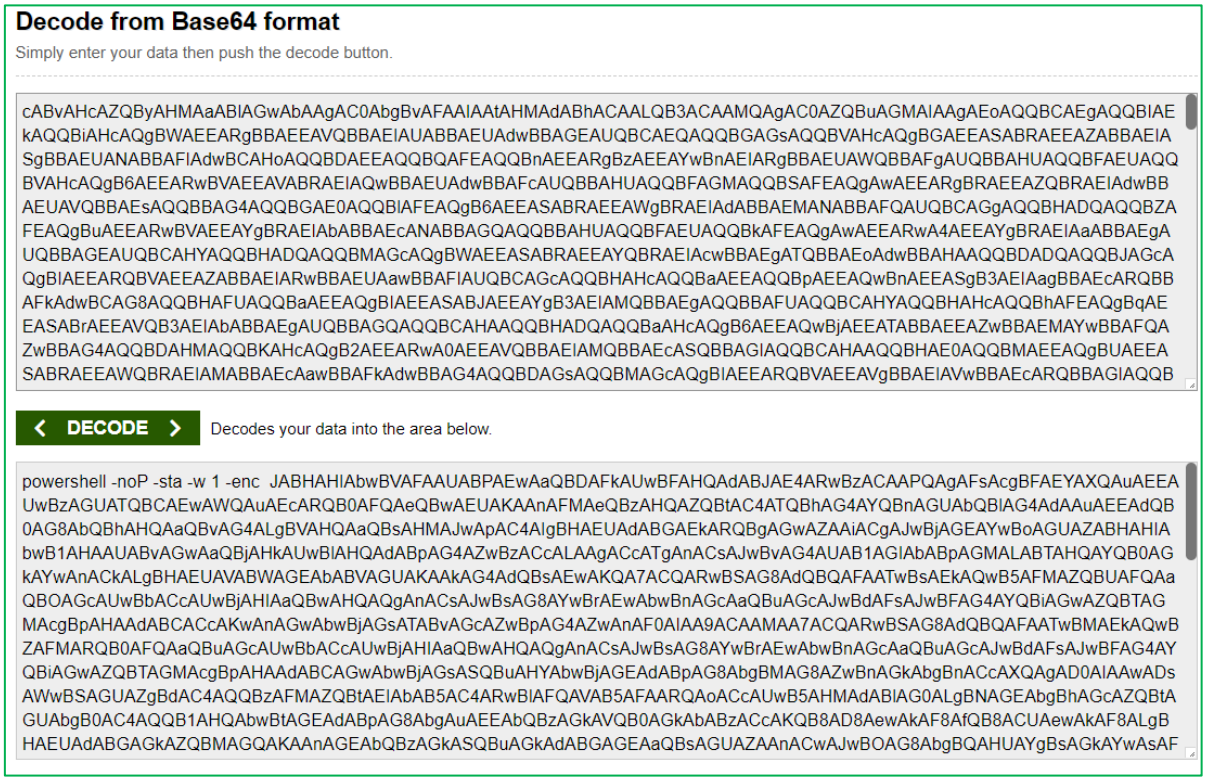
```
root@kali:~/ProgramFiles/volatility_2.6# ls -la
total 15100
drwx------ 2 root root     4096 Mar 29 12:25 .
drwxr-xr-x 6 root root     4096 Mar 27 17:05 ..
-rwx------ 1 root root      778 Dec 27  2016 AUTHORS.txt
-rw-r--r-- 1 root root   262144 Mar 29 12:25 file.None.0xfffffa80017dcc60.vacb
-rw-r--r-- 1 root root    16384 Mar 29 12:25 file.None.0xfffffa80022ac740.dat
-rwx------ 1 root root    15127 Jul  6  2016 LICENSE.txt
```

7. Ekstrak teks yang dapat dibaca dari berkas *dump* memori tersebut guna mengidentifikasi informasi penting seperti artefak, URL, atau indikasi aktifitas mencurigakan.

```
strings file.None.0xfffffa80017dcc60.vacb
```

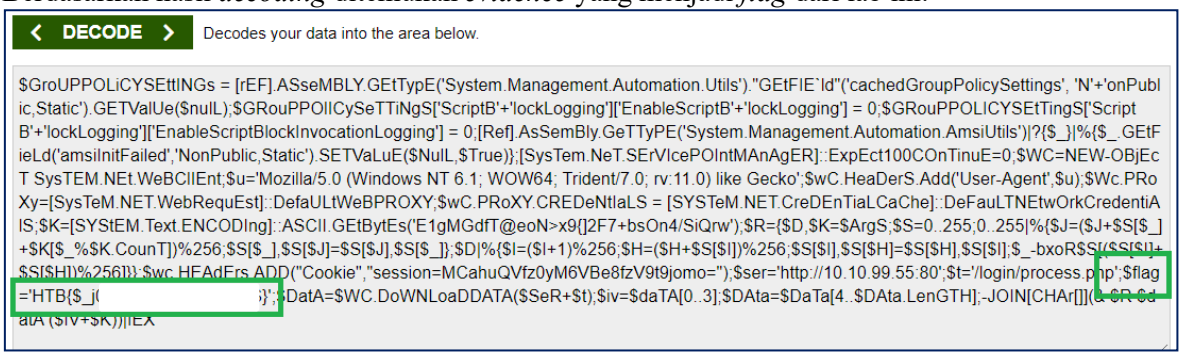
8. Dari hasil pembacaan menggunakan perintah *strings* diatas, akan ditemukan sejumlah informasi dan karakter yang terenkripsi. Investigator menganalisis pada berkas tersebut dan menyimpulkan bahwa karakter tersebut adalah Base64.

   Selanjutnya investigator mencoba melakukan *decoding* terhadap *cipher* tersebut, serta menganalisis keluarannya dan mendapatkan temuan atau *flag* akhir pada skenario ini.



Berdasarkan hasil *decoding* diatas, menunjukkan bahwa input berupa string terenkripsi dikonversi menjadi perintah asli berupa kode *PowerShell*.

Berdasarkan hasil *decoding* ditemukan *evidence* yang menjadi *flag* dari lab ini.



Daftar Pustaka

1. Schuster, A. (2006). Searching for Processes and Threads in Microsoft Windows Memory Dumps. Digital Investigation, 3(Supplement), 10–16. DOI: 10.1016/j.diin.2006.06.010.
2. Ligh, M. H., Case, A., Levy, J., & Walters, A. (2014). The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory. Indianapolis: Wiley Publishing.
3. Walters, A., Petroni, N., Fraser, T., & Arbaugh, W. (2006). FATKit: A Framework for the Extraction and Analysis of Digital Forensic Data from Volatile System Memory. Digital Investigation Workshop (DFRWS).
4. The Volatility Foundation. (2026). Volatility Framework. Retrieved from https://volatilityfoundation.org/
5. The Volatility Foundation. (2026). The Volatility Framework: Memory Forensics Platform. Retrieved from https://volatilityfoundation.org/the-volatility-framework/
6. Hack The Box. (2026). Reminiscent Challenge – Forensics Category. Retrieved from https://app.hackthebox.com/challenges/Reminiscent

Biografi Penulis

Yulian Sani. Seorang *enthusiast* di bidang keamanan siber dan teknologi informasi dengan pengalaman di industri perbankan, manufaktur, dan konsultansi. Saat ini, ia berperan dalam fungsi *Internal Audit* dan *Data Analytics*, dengan keahlian dalam keamanan siber, *penetration testing*, serta *software engineering*.

Meraih gelar Magister Administrasi Bisnis dari Institut Teknologi Telkom (Universitas Telkom). Sebelumnya, ia juga menyelesaikan Magister Teknik Informatika dengan fokus Keamanan Informasi dari Universitas Langlangbuana, serta gelar Sarjana Teknik Informatika dari Universitas Siliwangi. Saat ini memiliki beberapa sertifikasi profesional di bidang teknologi informasi, seperti CEH, CHFI, dan ECIH.