E-ISSN : 2722-0346

Implementasi Volatility dalam Mengalanalisa Malware pada Memory Dump

Gregorius Hendita Artha Kusuma
Teknik Informatika, Fakultas TeknikUniversitas Pancasila
gregorius@univpancasila.ac.id

***Abstract*— Malware infections on computer systems have become a significant threat to information security. In response to these challenges, memory analysis has proven to be an effective method for detecting and investigating malware activities. In this research, we utilize Volatility, a popular memory forensics tool, to analyze memory dumps from malware-infected systems.**

Our primary objective is to identify and uncover artifacts associated with malware infections within the memory dump. We leverage various widely-used Volatility plugins to extract critical information such as malicious processes, modified kernel modules, suspicious network traces, and other malicious entities.

Through a series of analysis steps, we successfully detect the presence of malware infections with a high level of accuracy. We also determine the types and variants of malware involved in the attack. Furthermore, we perform behavioral analysis of the malware, enabling us to understand the objectives, propagation methods, and impact of the infection.

The results of this research provide valuable insights for prevention and mitigation of malware attacks. By utilizing Volatility as a memory forensics analysis tool, researchers and security professionals can effectively identify and combat malware threats. We also outline recommendations for steps to strengthen system security and protect valuable data from future malware attacks.

***Kata Kunci* — volatility,malware,memory dump**

I. PENDAHULUAN

Di era digital saat ini, infeksi *malware* merupakan ancaman utama bagi keamanan informasi. *Malware*, seperti *virus*, *worm*, dan *ransomware*, dapat memasuki sistem komputer, membahayakan integritas data, dan mengganggu operasi penting. Untuk mengatasi ancaman ini, metode yang efektif untuk mendeteksi dan menyelidiki aktivitas malware sangat penting.

Investigasi memori telah menjadi teknik yang efektif untuk menganalisis keadaan sistem komputer pada saat terjadi kegagalan. Dengan memeriksa *core dumps* yang merekam konten memori sistem yang mudah menguap, penyelidik dapat menemukan bukti berharga tentang keberadaan, perilaku, dan dampak malware berbahaya. Analisis memori memungkinkan pemahaman yang lebih baik tentang taktik, teknik, dan proses yang digunakan oleh penyerang, memungkinkan mitigasi ancaman proaktif dan meningkatkan keamanan sistem secara keseluruhan.

Dalam beberapa tahun terakhir, *Volatility* telah mendapatkan ketenaran sebagai kerangka kerja forensik memori yang populer. Instabilitas menyediakan satu set lengkap *plugin* yang memudahkan untuk mengekstraksi dan menganalisis informasi berharga dari dump inti. Plugin ini membantu mengidentifikasi proses berbahaya, mendeteksi perubahan tingkat kernel, membangun kembali aktivitas jaringan, dan mengungkap berbagai artefak yang terkait dengan infeksi *malware*.

Log ini dimaksudkan untuk mengeksplorasi analisis dump inti yang diserang malware menggunakan Ketidakstabilan sebagai alat utama. Kami akan menjelaskan langkah-langkah yang terlibat dalam analisis memori, plugin *Volatility* yang digunakan, dan metode deteksi dan investigasi infeksi *malware*. Melalui penelitian ini diharapkan dapat membantu untuk lebih memahami teknik investigasi memori dalam memerangi ancaman malware, serta memberikan informasi praktis dan rekomendasi untuk peningkatan keamanan sistem komputer.

II. DASAR TEORI

A Malware

Merupakan singkatan dari "malicious software" yang mengacu pada perangkat lunak berbahaya yang dirancang untuk merusak, mengganggu, atau mencuri informasi dari sistem komputer yang terinfeksi. Malware dapat berupa virus, worm, trojan, ransomware, dan sebagainya.

Jenis-jenis *malware:*

a) *Virus*
Virus adalah program yang menempel pada file atau program lain dan bereplikasi saat file atau program tersebut dijalankan. Virus dapat merusak data atau merusak sistem.

b) *Worm*
Worm adalah jenis malware yang dapat menyebar sendiri melalui jaringan komputer tanpa membutuhkan interaksi pengguna. Worm biasanya menggunakan kelemahan sistem atau

celah keamanan untuk menginfeksi komputer lain dan mengirim salinan dirinya sendiri ke komputer-komputer tersebut.

c) *Trojan*
Trojan horse atau trojan adalah jenis malware yang bersembunyi di program atau file yang tampaknya aman atau berguna. Ketika program atau file tersebut dijalankan, Trojan dapat membuka pintu belakang ke dalam sistem, memungkinkan penyerang untuk mengakses dan mengontrol komputer korban.

d) *Ransomware*
Ransomware adalah jenis malware yang mengenkripsi data di komputer korban kemudian meminta uang tebusan (ransom) agar data tersebut dapat didekripsi. Ransomware sering kali menyerang dengan mengeksploitasi lubang keamanan atau melalui tautan berbahaya atau lampiran dalam email.

e) *Spyware*
Spyware adalah jenis malware yang dirancang untuk mengumpulkan informasi tentang pengguna tanpa izin atau sepengetahuan mereka. Spyware dapat merekam penekanan tombol, memantau aktivitas internet, mencuri informasi pribadi, dan mengirimkannya ke pihak yang tidak bertanggung jawab.

f) *Adware*
Adware adalah jenis malware yang menampilkan iklan yang tidak diinginkan di komputer pengguna. Adware sering dipasang bersamaan dengan program lain yang diunduh oleh pengguna dan dapat menghasilkan pendapatan bagi pembuatnya dengan menampilkan iklan.

g) *Botnet*
Botnet adalah jaringan komputer yang terinfeksi malware dan dikendalikan oleh penyerang jarak jauh. Komputer yang terinfeksi (disebut sebagai "bot" atau "zombie") dapat digunakan untuk meluncurkan serangan DDoS, mendistribusikan spam, atau melakukan aktivitas berbahaya lainnya.

h) *Rootkit*
Rootkit adalah jenis malware yang dirancang untuk menyembunyikan keberadaannya atau aktivitas berbahaya pada sistem yang terinfeksi. Rootkit sering mengubah konfigurasi sistem dan mengontrol akses ke sistem operasi, membuatnya sulit untuk dideteksi dan dihapus.

B. Memory dump

Memory dump (atau juga disebut sebagai core dump) adalah proses pengambilan dan penyimpanan isi memori sistem komputer pada suatu waktu tertentu. Memory dump biasanya dilakukan ketika terjadi kesalahan (crash) atau kegagalan sistem yang parah.

Saat terjadi kesalahan atau kegagalan sistem yang serius, seperti blue screen of death (BSOD) pada sistem operasi Windows, memory dump dapat direkam untuk menyimpan informasi tentang status memori sistem pada saat itu. Isi memori ini termasuk kode program, data, register prosesor, dan informasi lainnya yang mungkin membantu dalam menganalisis penyebab masalah.

Memory dump berguna dalam proses debugging dan analisis kesalahan sistem. Dengan menganalisis memory dump, pengembang perangkat lunak atau teknisi sistem dapat mencari tahu apa yang terjadi sebelum kegagalan terjadi, melacak bug atau masalah yang mungkin ada dalam program atau sistem, dan mengidentifikasi penyebab akar masalah.

Memory dump biasanya menghasilkan file besar yang berisi dump lengkap dari memori fisik atau sebagian tertentu dari memori yang relevan. File memory dump ini dapat dianalisis dengan menggunakan perangkat lunak khusus seperti *debugger* untuk mencari tahu penyebab kesalahan dan mengembangkan solusi yang sesuai.

Penting untuk diingat bahwa akses dan analisis memory dump biasanya memerlukan pengetahuan teknis yang cukup dan dapat menjadi tugas yang kompleks. Oleh karena itu, biasanya dilakukan oleh pengembang perangkat lunak, teknisi sistem, atau ahli keamanan yang berpengalaman.

C. Volatility

Volatility adalah sebuah kerangka kerja (framework) sumber terbuka (open source) yang digunakan untuk melakukan analisis forensik pada memori komputer yang mengalami memory dump. Kerangka kerja ini dapat membantu para profesional keamanan dan analis forensik dalam memahami dan menganalisis data yang terdapat dalam memori fisik komputer.

Volatility dirancang khusus untuk bekerja dengan file memory dump yang dihasilkan oleh sistem operasi seperti Windows, Linux, Mac OS, dan beberapa sistem operasi lainnya. Dengan menggunakan Volatility, analis dapat mengekstraksi informasi berharga dari memori sistem yang dapat digunakan untuk mendeteksi serangan jaringan, menganalisis malware, memulihkan data yang hilang, dan menyelidiki aktivitas mencurigakan pada sistem.

Kerangka kerja ini menyediakan sejumlah plugin yang dapat digunakan untuk menganalisis berbagai aspek memori, termasuk proses yang berjalan, daftar pemuatan modul (DLL), koneksi jaringan, register, tumpukan (stack), dan banyak lagi. Volatility memanfaatkan struktur internal memori yang diketahui untuk menginterpretasikan data dalam file memory dump, sehingga memungkinkan pengguna untuk memperoleh informasi yang signifikan tentang keadaan sistem pada saat memory dump dibuat.

Volatility juga mendukung analisis memori dalam lingkungan virtual, seperti file image virtual machine (VM) atau snapshot, yang memungkinkan pengguna untuk menganalisis memori dalam VM yang sedang berjalan.

Dalam analisis forensik, Volatility merupakan alat yang berguna untuk mendapatkan pemahaman yang lebih dalam tentang keadaan sistem pada saat memory dump diambil. Namun, penggunaan Volatility memerlukan pengetahuan dan pemahaman yang cukup tentang struktur internal sistem operasi dan mekanisme analisis forensik.

Fitur-fitur *Volatility:*

a. Analisis Proses: Volatility dapat digunakan untuk menganalisis dan mendapatkan informasi tentang proses yang

berjalan pada sistem pada saat memory dump diambil. Fitur ini memungkinkan pengguna untuk melihat daftar proses, ID proses, alamat memori, file eksekusi yang terkait, dan informasi penting lainnya.

b. Analisis Modul: Volatility dapat mengekstraksi informasi tentang modul atau library (DLL) yang dimuat pada sistem pada saat memory dump diambil. Ini termasuk daftar modul yang dimuat, alamat memori, informasi versi, dan tautan dengan proses tertentu.

c. Analisis Jaringan: Volatility mendukung analisis jaringan pada memori. Pengguna dapat melihat daftar koneksi jaringan, informasi IP, port, status koneksi, dan lain-lain. Fitur ini berguna untuk mendeteksi aktivitas jaringan yang mencurigakan atau untuk menelusuri jejak serangan jaringan.

d. Analisis Register: Volatility memungkinkan analisis register sistem pada saat memory dump diambil. Pengguna dapat melihat nilai register, informasi penting seperti sistem operasi, pengguna yang sedang login, dan konfigurasi register lainnya.

e. Analisis Tumpukan (Stack): Volatility dapat menganalisis tumpukan (stack) memori untuk mendapatkan informasi tentang pemanggilan fungsi, alamat memori, dan jejak eksekusi program. Hal ini dapat membantu dalam pemahaman alur eksekusi dan mendeteksi jejak aktivitas mencurigakan.

f. Analisis File Cache: Volatility mendukung analisis file cache dalam memori. Pengguna dapat melihat daftar file yang ada dalam cache, alamat memori terkait, ukuran file, waktu akses, dan informasi lainnya. Fitur ini berguna untuk mendapatkan wawasan tentang file-file yang telah diakses pada saat memory dump diambil.

g. Analisis Malware: Volatility memiliki fitur khusus yang mendukung analisis malware pada memori. Ini mencakup pendeteksian malware, analisis pola perilaku, pengekstrakan kode berbahaya, dan identifikasi komunikasi jaringan yang mencurigakan.

D. Mendeteksi dan menyelidiki malware

Mendeteksi dan menyelidiki infeksi malware adalah tujuan utama dari analisis core dump menggunakan Volatility. Selama fase ini, berbagai teknik dan langkah diambil untuk mengidentifikasi keberadaan malware, menentukan jenis malware yang terlibat, serta memahami perilaku dan dampak infeksi. Beberapa aspek penting dari proses ini meliputi:

a. Identifikasi proses berbahaya

Dengan bantuan Volatility dan plugin yang sesuai, analisis dilakukan saat dump memori diambil untuk mengidentifikasi proses berbahaya atau mencurigakan pada sistem. Proses ini dapat menyebabkan malware bersembunyi di memori dan mencoba menghindari deteksi oleh alat keamanan tradisional.

b. Jejak jaringan yang mencurigakan

Analisis memori dapat digunakan untuk mengidentifikasi jejak jaringan yang mencurigakan, seperti koneksi abnormal, komunikasi dengan alamat IP atau domain yang mencurigakan, atau aktivitas protokol yang tidak biasa. Informasi ini dapat memberikan informasi tentang metode distribusi malware dan komunikasi dengan server command and control (C&C) atau entitas berbahaya lainnya.

c. Modifikasi kernel yang mencurigakan

Modifikasi kernel adalah tanda yang sering digunakan oleh malware untuk menghindari deteksi dan memperoleh hak akses yang lebih tinggi dalam sistem. Dengan Volatility, analisis dilakukan untuk mendeteksi modifikasi kernel yang mencurigakan yang mungkin dilakukan oleh malware, seperti hooking atau rootkit.

d. Analisis perilaku malware

Dalam upaya untuk memahami tujuan dan metode penyebaran malware, analisis perilaku dilakukan pada elemen yang terkait dengan malware, seperti kode berbahaya, modifikasi sistem, manipulasi berkas, enkripsi data, dan aktivitas lainnya yang tercatat dalam memory dump. Analisis ini membantu mengungkap alur kerja malware dan potensi kerusakan yang dapat disebabkannya.

e. Dampak dan upaya mitigasi

Selain mengidentifikasi dan memahami infeksi malware, analisis memory dump juga dapat memberikan wawasan tentang dampak yang ditimbulkan oleh malware pada sistem. Informasi ini dapat digunakan untuk mengembangkan strategi mitigasi yang efektif, termasuk langkah-langkah perbaikan keamanan dan tindakan pencegahan untuk mencegah infeksi serupa di masa depan.

III. METODE PENGUJIAN

Peneliti menguji pada memori dump yang sudah terinfeksi oleh cridex. Cridex adalah salah satu jenis malware yang termasuk dalam keluarga Trojan. Trojan Cridex, juga dikenal sebagai Cridex Banking Trojan atau Feodo, pertama kali muncul pada tahun 2011 dan telah menjadi ancaman yang signifikan di dunia maya.

Cridex dirancang untuk mencuri informasi sensitif dari komputer yang terinfeksi, terutama informasi keuangan seperti login dan password perbankan, data kartu kredit, dan informasi pribadi lainnya. Biasanya, Cridex menyebar melalui email spam yang mengandung lampiran berbahaya atau tautan yang mengarah ke situs web yang terinfeksi. Jika pengguna mengklik lampiran atau tautan tersebut, Cridex akan diunduh dan menginfeksi komputer korban.

Setelah berhasil menginfeksi komputer, Cridex dapat mencuri informasi dari program email, browser, dan aplikasi perbankan yang ada di komputer. Ini dapat mengakibatkan pencurian identitas, kehilangan dana dari rekening bank, atau penyalahgunaan informasi pribadi korban.

Dan peneliti akan menganalisis memori dump untuk mendapatkan informasi yang penting untuk mengatasi/mencegah malware tersebut berkembang.

Tahapan analisis yang akan dilakukan peneliti:

 E-ISSN : 2722-0346

· **Preliminary Analysis**:

- ✓ Identifikasi informasi dasar: Periksa metadata dan informasi dasar tentang memory dump, seperti arsitektur sistem target, versi sistem operasi, dan waktu pengambilan memory dump.
- ✓ Verifikasi integritas: Pastikan keaslian dan integritas memory dump yang akan digunakan dalam analisis.

· **Process Analysis**:

- ✓ Ekstraksi daftar proses: Gunakan plugin seperti pslist atau psscan untuk mendapatkan daftar proses yang berjalan saat memory dump diambil.
- ✓ Analisis hubungan proses: Identifikasi proses yang mencurigakan, periksa hubungan parent-child antar proses, dan identifikasi kemungkinan hubungan yang tidak biasa atau tidak wajar.

· **Network Analysis**:

- ✓ Ekstraksi informasi jaringan: Gunakan plugin seperti netscan atau connscan untuk mendapatkan informasi tentang koneksi jaringan yang aktif saat memory dump diambil.
- ✓ Analisis jejak jaringan: Identifikasi aktivitas jaringan yang mencurigakan, seperti koneksi ke alamat IP atau domain yang tidak biasa, atau port yang tidak lazim digunakan.

· **Malware Analysis**:

- ✓ Identifikasi malware artifacts: Gunakan plugin seperti malfind, ssdt, atau apihooks untuk mencari tanda-tanda adanya malware dalam memory dump, seperti kode berbahaya, modifikasi kernel, atau manipulasi sistem lainnya.
- ✓ Analisis perilaku malware: Rekonstruksi perilaku malware berdasarkan artefak yang ditemukan, termasuk aktivitas yang mencurigakan, modifikasi berkas, atau enkripsi data.

· **Memory Analysis**:

- ✓ Analisis alokasi memori: Gunakan plugin seperti vadinfo atau vadtree untuk menganalisis alokasi memori yang mencurigakan, seperti alokasi memori tersembunyi atau modifikasi kode yang mencurigakan.
- ✓ Analisis heap: Periksa heap memory untuk mencari data yang telah dihapus atau manipulasi yang mencurigakan oleh malware.

· **Evidence Collection**:

- ✓ Ekstraksi file dan registry: Gunakan plugin seperti dumpfiles atau hivelist untuk mengekstraksi file dan entri registri yang relevan dari memory dump untuk analisis lebih lanjut.

IV. HASIL DAN PEMBAHASAN

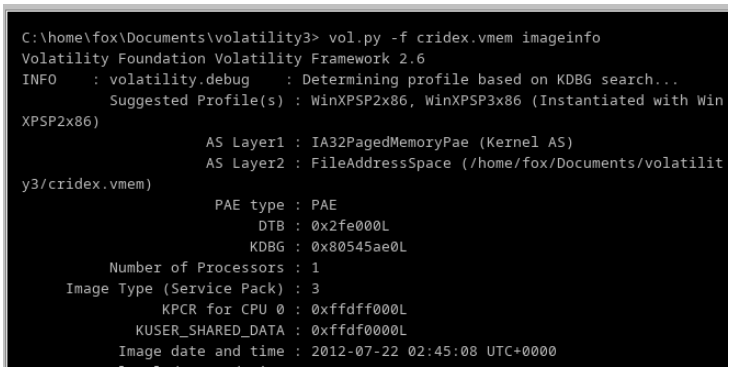
Peneliti memakai file sample cridex yang diambil dari website https://github.com/volatilityfoundation/volatility/wiki/Memory-Samples dan disimpan dengan file bernama cridex.vmem

1. Mencari detail file

Command yang digunakan untuk mencari detail dari suatu dump memori adalah -f cridex.vmem imageinfo, fungsi dari imageinfo adalah untuk mengidentifikasi dan memberikan informasi tentang berbagai atribut dan properti gambar memori, seperti arsitektur sistem, versi sistem operasi, dan profil profil. Berikut adalah beberapa informasi yang biasanya diberikan oleh perintah imageinfo:

- **Profile**: Menampilkan profil atau plugin yang direkomendasikan untuk digunakan dalam analisis berdasarkan informasi gambar memori yang diberikan.
- **Sistem Operasi**: Menampilkan informasi tentang sistem operasi yang terpasang pada gambar memori, termasuk versi dan arsitektur sistem.
- **Tanggal dan Waktu**: Menampilkan tanggal dan waktu ketika gambar memori diambil.
- **Build Information**: Menampilkan informasi tambahan tentang build sistem, seperti build number atau timestamp.
- **Bitness**: Menampilkan informasi apakah gambar memori 32-bit atau 64-bit.
- **Profil Kernel**: Menampilkan profil kernel yang direkomendasikan untuk digunakan dalam analisis.

jika dijalankan untuk file cridex.vmem maka output yang keluar adalah



Gambar 1. Informasi File Memori

bisa dilihat dari gambar diatas bahwa terdapat informasi dari file memori tersebut diantaranya adalah sistem operasi yang dipakai, jumlah prosesor dan tanggal waktu file memori tersebut dibuat.

2. *Process List*

setelah mengetahui profile yang ingin dianalisis maka peneliti perlu melihat *process list* dari komputer tersebut dengan command vol.py -f cridex.vmem --profile=WinXPSP2x86 pslist, pslist berfungsi untuk menampilkan daftar proses yang sedang berjalan dalam gambar memori (memory image) yang sedang dianalisis. Fungsi utama dari pslist adalah untuk mengidentifikasi proses-proses yang aktif pada saat gambar memori diambil dan memberikan informasi terperinci tentang setiap proses.

Berikut adalah beberapa informasi yang biasanya diberikan oleh perintah pslist:

- **PID**: Menampilkan Process ID (PID) dari setiap proses.
- **Nama Proses**: Menampilkan nama proses yang sedang berjalan.
- **Parent PID**: Menampilkan PID dari proses induk (parent) dari setiap proses.
- **Session ID**: Menampilkan ID sesi dari setiap proses.
- **Permintaan CPU**: Menampilkan jumlah permintaan CPU (CPU request) yang dilakukan oleh setiap proses.
- **Waktu Pembuatan**: Menampilkan waktu pembuatan (creation time) dari setiap proses.
- **Alamat Base**: Menampilkan alamat base (base address) dari ruang alamat (address space) proses.

dan *output*nya adalah seperti gambar dibawah ini,

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start
0x823c89c8	System	4	0	53	240	------	0	
0x822f1020	smss.exe	368	4	3	19	------	0	2012-07-22 02:42:31 UTC+00
0x822a0598	csrss.exe	584	368	9	326	0	0	2012-07-22 02:42:32 UTC+00
0x82298700	winlogon.exe	608	368	23	519	0	0	2012-07-22 02:42:32 UTC+00
0x81e2ab28	services.exe	652	608	16	243	0	0	2012-07-22 02:42:32 UTC+00
0x81e2a3b8	lsass.exe	664	608	24	330	0	0	2012-07-22 02:42:32 UTC+00
0x82311360	svchost.exe	824	652	20	194	0	0	2012-07-22 02:42:33 UTC+00
0x81e29ab8	svchost.exe	908	652	9	226	0	0	2012-07-22 02:42:33 UTC+00
0x823001d0	svchost.exe	1004	652	64	1118	0	0	2012-07-22 02:42:33 UTC+00
0x821dfda0	svchost.exe	1056	652	5	60	0	0	2012-07-22 02:42:33 UTC+00
0x82295650	svchost.exe	1220	652	15	197	0	0	2012-07-22 02:42:35 UTC+00
0x821dea70	explorer.exe	1484	1464	17	415	0	0	2012-07-22 02:42:36 UTC+00
0x81eb17b8	spoolsv.exe	1512	652	14	113	0	0	2012-07-22 02:42:36 UTC+00
0x81e7bda0	reader_sl.exe	1640	1484	5	39	0	0	2012-07-22 02:42:36 UTC+00
0x820e8da0	alg.exe	788	652	7	104	0	0	2012-07-22 02:43:01 UTC+00
0x821fcda0	wuauclt.exe	1136	1004	8	173	0	0	2012-07-22 02:43:46 UTC+00
0x8205bda0	wuauclt.exe	1588	1004	5	132	0	0	2012-07-22 02:44:01 UTC+00

Gambar 2. Informasi Memori

Dari gambar diatas terlihat beberapa *process* yang terjadi di dalam memori seperti exploler.exe tidak ada yang mencurigakan tetapi *reader_sl.exe* sangat asing dilihat dan peneliti menaruh perhatian pada program tersebut.

3. *Hiding Processes*

Tahap ini peneliti ingin mencari tahu apa ada proses yang disembunyikan di dalam memori tersebut dengan command *psxview* dan outputnya akan menjadi seperti gambar dibawah ini

```
Volatility Foundation Volatility Framework 2.6
Offset(P)  Name                    PID pslist psscan thrdproc pspcid csrss session deskthrd
---------- -------------------- ------ ------ ------ -------- ------ ----- ------- --------
0x02498700 winlogon.exe            608 True   True   True     True   True  True    True
0x02511360 svchost.exe             824 True   True   True     True   True  True    True
0x022e8da0 alg.exe                 788 True   True   True     True   True  True    True
0x020b17b8 spoolsv.exe            1512 True   True   True     True   True  True    True
0x0202ab28 services.exe            652 True   True   True     True   True  True    True
0x02495650 svchost.exe            1220 True   True   True     True   True  True    True
0x0207bda0 reader_sl.exe          1640 True   True   True     True   True  True    True
0x025001d0 svchost.exe            1004 True   True   True     True   True  True    True
0x02029ab8 svchost.exe             908 True   True   True     True   True  True    True
0x023fcda0 wuauclt.exe            1136 True   True   True     True   True  True    True
0x0225bda0 wuauclt.exe            1588 True   True   True     True   True  True    True
0x0202a3b8 lsass.exe               664 True   True   True     True   True  True    True
0x023dea70 explorer.exe           1484 True   True   True     True   True  True    True
0x023dfda0 svchost.exe            1056 True   True   True     True   True  True    True
0x024f1020 smss.exe                368 True   True   True     True   False False   False
0x025c89c8 System                    4 True   True   True     True   False False   False
0x024a0598 csrss.exe               584 True   True   True     True   False True    True
```

Gambar 3. Informasi *Hiding Process Memori*

jika ingin mengetahui apa ada *hiding processes* kita bisa melihat 2 kolom pertama yaitu *pslist* dan *psscan* jika tertulis false maka proses tersebut termasuk *hiding processes*, namun dalam kasus ini sesuai dengan gambar tidak ada false di kolom *pslist* dan *psscan*.

4. *Running Sockets* dan *open connections*

peneliti menggunakan command connscan untuk menganalisis dan menampilkan informasi tentang koneksi jaringan yang aktif pada saat gambar memori (memory image) diambil. Fungsi utama dari connscan adalah untuk mengidentifikasi koneksi jaringan yang berjalan dalam sistem yang sedang dianalisis, termasuk detail seperti alamat IP, port, status koneksi, dan proses yang terlibat.

```
C:\home\fox\Documents\volatility3> vol.py -f cridex.vmem --profile=WinXPSP2x86 connscan
Volatility Foundation Volatility Framework 2.6
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------ ---
0x02087620 172.16.112.128:1038      41.168.5.140:8080         1484
0x023a8008 172.16.112.128:1037      125.19.103.198:8080       1484
```

Gambar 4. Informasi koneksi berjalan

dari gambar diatas terlihat bahwa ada 2 koneksi yang berjalan pada memori dump tersebut menggunakan pid 1484. Setelah itu peneliti ingin mencari sockets yang berjalan pada memori dengan command sockets

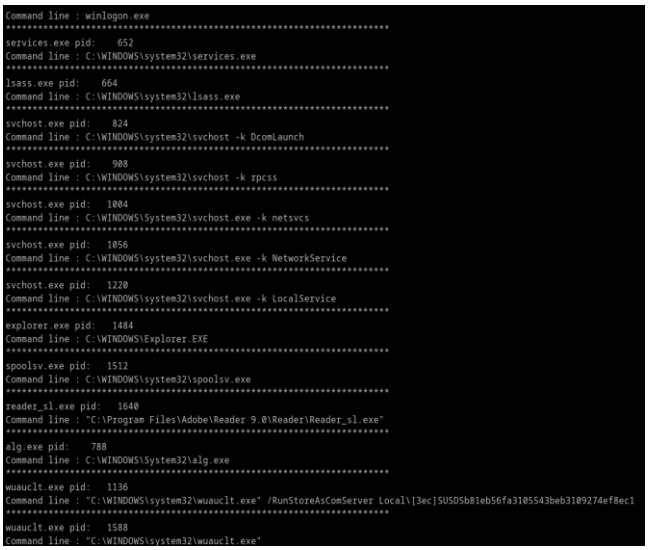
```
Volatility Foundation Volatility Framework 2.6
Offset(V)       PID   Port  Proto Protocol        Address         Create Time
---------- -------- ------ ------ --------------- --------------- -----------
0x81ddb780      664    500     17 UDP             0.0.0.0         2012-07-22 02:42:53 UTC+0000
0x82240d08     1484   1038      6 TCP             0.0.0.0         2012-07-22 02:44:45 UTC+0000
0x81dd7618     1220   1900     17 UDP             172.16.112.128  2012-07-22 02:43:01 UTC+0000
0x82125610      788   1028      6 TCP             127.0.0.1       2012-07-22 02:43:01 UTC+0000
0x8219cc08        4    445      6 TCP             0.0.0.0         2012-07-22 02:42:31 UTC+0000
0x81ec23b0      908    135      6 TCP             0.0.0.0         2012-07-22 02:42:33 UTC+0000
0x82276878        4    139      6 TCP             172.16.112.128  2012-07-22 02:42:38 UTC+0000
0x82277460        4    137     17 UDP             172.16.112.128  2012-07-22 02:42:38 UTC+0000
0x81e76620     1004    123     17 UDP             127.0.0.1       2012-07-22 02:43:01 UTC+0000
0x82172808      664      0    255 Reserved        0.0.0.0         2012-07-22 02:42:53 UTC+0000
0x81e3f460        4    138     17 UDP             172.16.112.128  2012-07-22 02:42:38 UTC+0000
0x821f0630     1004    123     17 UDP             172.16.112.128  2012-07-22 02:43:01 UTC+0000
0x822cd2b0     1220   1900     17 UDP             127.0.0.1       2012-07-22 02:43:01 UTC+0000
0x82172c50      664   4500     17 UDP             0.0.0.0         2012-07-22 02:42:53 UTC+0000
0x821f0d00        4    445     17 UDP             0.0.0.0         2012-07-22 02:42:31 UTC+0000
```

Gambar 5. Informasi Socket Memori

terlihat *socket* yang berjalan pada memori dump dan *process id* yang terlibat dan *process id* yang menjadi perhatian adalah 1484 yang menjalankan port 1038 menggunakan TCP.

5. *Command Line*

Pada tahap ini peneliti ingin mencari tahu *command line* apa saja yang dijalankan pada komputer tersebut, dan untuk mencari tahu maka peneliti menggunakan command volatility yaitu cmdline



Gambar 6. Informasi Command Line

Gambar diatas merupakan *command line* yang berjalan pada komputer dan kita dapat melihat *proccess id* yang peneliti beri perhatian yaitu 1484, *process id* tersebut secara langsung berhubungan dengan *process id* 1640 yang menjalankan Reader_sl.exe dan sekarang kita dapat mengedintifikasi hubungan antara exploler.exe dengan Reader_sl.exe.

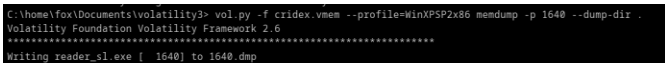
6. *Memory dump*

Setelah mengetahui bahwa *pid* 1484 terhubung dengan *pid* 1640 maka peneliti ingin mengetahui apa yang terjadi *process* apa yang ada didalam *pid* 1640 dengan menggunakan command, sebelumnya peneliti membuat dump dari file exe dari *pid* 1640 dengan command

vol.py -f cridex.vmem --profile=WinXPSP2x86 procdump -p 1640 --dump-dir .

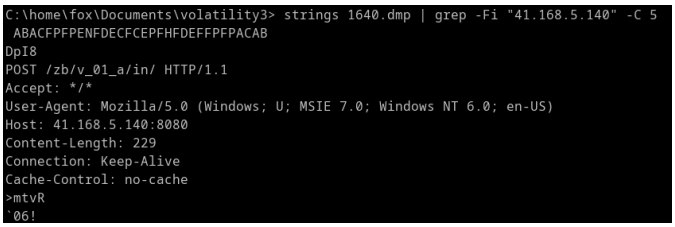
Gambar 7. Informasi *Excutable*

maka akan muncul file baru, itu adalah file dump dari exe yang ada di *pid* 1640.Setelah itu peneliti perlu melihat proses apa yang terjadi dibelakang *pid 1640* dengan command
vol.py -f cridex.vmem --profile=WinXPSP2x86 memdump -p 1640 --dump-dir .



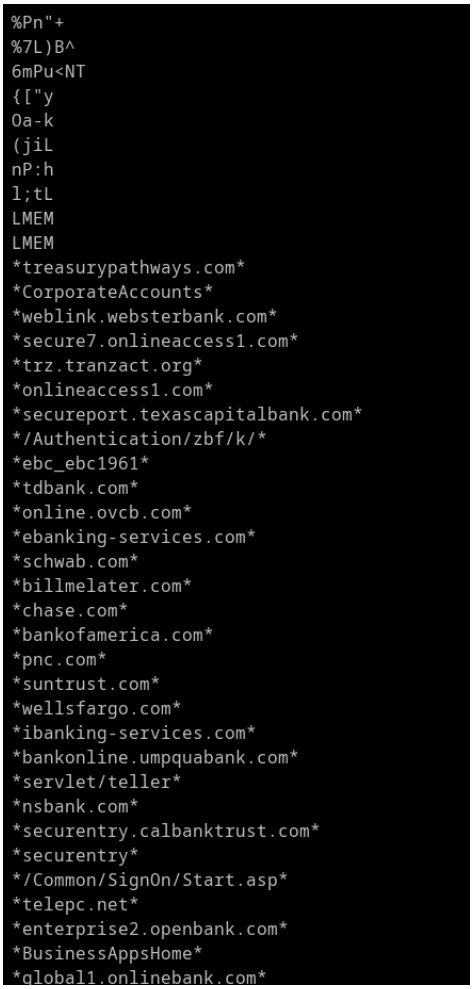
Gambar 8. Informasi Prosses di Command Line

setelah menjalankan command tersebut peneliti ingin mencari detail apa yang terjadi dibelakang *process* tersebut dengan menggunakan command string yaitu strings 1640.dmp | grep -Fi "41.168.5.140" -C 5



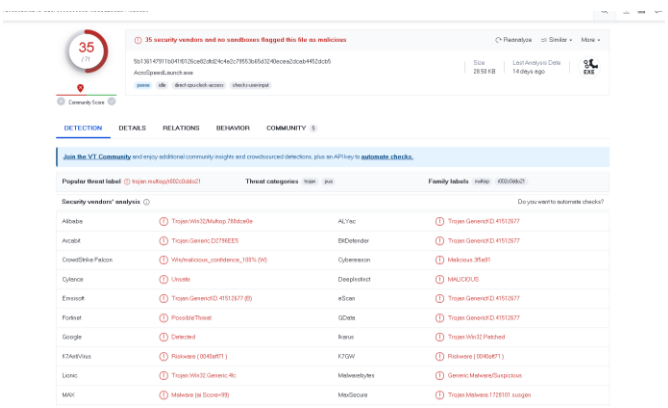
Gambar 9. Informasi POST Request

Gambar diatas adalah bahwa process tersebut menjalakankan POST request yang berpotensi mengirim data dari komputer korban dan dari memori dump yang telah kita buat menyimpan banyak informasi di dalamnya dan peneliti melihat secara lengkap dengan command strings yaitu strings 1640.dmp | less



Gambar 10. Informasi Hasil Akhir

Terdapat aktifitas mencurigakan pada proses tersebut. tetapi peneliti perlu melakukan konfirmasi bahwa file tersebut adalah sebuah malware dan untuk melakukan itu peneliti menggunakan website virustotal yaitu https://www.virustotal.com untuk file dump dari *pid* 1640 dan di dapati seperti gambar berikut ini



Gambar 11. Informasi Hasil File Dump

35 dari 71 *scanners* mendeteksi file tersebut adalah file berbahaya maka dengan ini kita dapat menyimpulkan bahwa *reader_sl.exe* adalah malware, dan tahap selanjutnya adalah peneliti ingin mengetahui bagaimana file tersebut berjalan dan muncul, maka peneliti akan menggali pada registry untuk mengetahuinya.

7. *Registry Hive*

Jika pembaca pernah mengalami terinfeksi malware dan mengetahui bahwa file tersebut adalah file berbahaya dan langsung menghapusnya maka langkah tersebut kurang benar karena jika pembaca hanya menghapus file tersebut tanpa menghapus file inti dari malware tersebut maka malware tersebut akan muncul lagi dengan nama yang berbeda. Peneliti ingin mengetahui kunci dari malware agar komputer bisa terbebas dari malware tersebut, maka peneliti menjalankan command hivelist yang berfungsi untuk menampilkan daftar entri registry hive yang terbuka dalam gambar memori (memory image) yang sedang dianalisis. Fungsi utama dari hivelist adalah untuk mengidentifikasi dan memberikan informasi tentang hive registry yang terbuka, termasuk lokasi alamat fisik dan ukuran setiap hive.

```
C:\home\fox\Documents\volatility3> vol.py -f cridex.vmem --profile=WinXPSP2x86 hivelist
Volatility Foundation Volatility Framework 2.6
Virtual    Physical   Name
---------- ---------- ----
0xe18e5b60 0x093f8b60 \Device\HarddiskVolume1\Documents and Settings\Robert\Local Settings\Application Data\Microsoft\Windows\UsrClas
0xe1a19b60 0x0a5a9b60 \Device\HarddiskVolume1\Documents and Settings\Robert\NTUSER.DAT
0xe18398d0 0x08a838d0 \Device\HarddiskVolume1\Documents and Settings\LocalService\Local Settings\Application Data\Microsoft\Windows\U
0xe18614d0 0x08e624d0 \Device\HarddiskVolume1\Documents and Settings\LocalService\NTUSER.DAT
0xe183bb60 0x08e2db60 \Device\HarddiskVolume1\Documents and Settings\NetworkService\Local Settings\Application Data\Microsoft\Windows
0xe17f2b60 0x08519b60 \Device\HarddiskVolume1\Documents and Settings\NetworkService\NTUSER.DAT
0xe1570510 0x07669510 \Device\HarddiskVolume1\WINDOWS\system32\config\software
0xe1571008 0x0777f008 \Device\HarddiskVolume1\WINDOWS\system32\config\default
0xe15709b8 0x076699b8 \Device\HarddiskVolume1\WINDOWS\system32\config\SECURITY
0xe15719e8 0x0777f9e8 \Device\HarddiskVolume1\WINDOWS\system32\config\SAM
0xe13ba008 0x02e4b008 [no name]
0xe1035b60 0x02ac3b60 \Device\HarddiskVolume1\WINDOWS\system32\config\system
0xe102e008 0x02a7d008 [no name]
```

Gambar 12. Informasi Hasil Target

Peneliti ingin mengetahui apa exe yang menjalankan malware pada saat komputer baru dinyalakan atau *startup* dan untuk itu command yang dijalankan adalah printkey dan targetnya adalah registry dari "Software\Microsoft\Windows\CurrentVersion\Run”

```
C:\home\fox\Documents\volatility3> vol.py -f cridex.vmem --profile=WinXPSP2x86 printkey -K "Software\Microsoft\Windows\CurrentVersion\Run"
Volatility Foundation Volatility Framework 2.6
Legend: (S) = Stable   (V) = Volatile

----------------------------
Registry: \Device\HarddiskVolume1\Documents and Settings\Robert\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
Key name: Run (S)
Last updated: 2011-04-13 00:55:13 UTC+0000

Subkeys:

Values:
----------------------------
Registry: \Device\HarddiskVolume1\Documents and Settings\Robert\NTUSER.DAT
Key name: Run (S)
Last updated: 2012-07-22 02:31:51 UTC+0000

Subkeys:

Values:
REG_SZ       KB00207877.exe  : (S) "C:\Documents and Settings\Robert\Application Data\KB00207877.exe"
----------------------------
Registry: \Device\HarddiskVolume1\WINDOWS\system32\config\default
Key name: Run (S)
Last updated: 2011-04-12 20:31:49 UTC+0000

Subkeys:

Values:
----------------------------
Registry: \Device\HarddiskVolume1\Documents and Settings\LocalService\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
Key name: Run (S)
Last updated: 2011-04-13 00:55:13 UTC+0000

Subkeys:
```

Gambar 13. Informasi Proses pada saat Startup

Gambar diatas adalah proses yang berjalan pada saat komputer *startup*, jika menjalankan proses untuk sistem adalah normal tetapi jika ada proses yang menjalankan file exe itu adalah anomali atau tidak normal dan dapat dilihat terdapat proses yang menjalankan exe.

```
Values:
REG_SZ       KB00207877.exe  : (S) "C:\Documents and Settings\Robert\Application Data\KB00207877.exe"
----------------------------
Registry: \Device\HarddiskVolume1\WINDOWS\system32\config\default
Key name: Run (S)
Last updated: 2011-04-12 20:31:49 UTC+0000

Subkeys:
```

Gambar 14. Informasi Proses Registry dan Subkey

dilihat bahwa ada proses KB00207877.exe yang berjalan dan lokasi file tersebut ada di folder *Application Data* maka peneliti perlu mencari tahu apakah KB00207877.exe terhubung dengan *process id* 1640 dan untuk mengetahui itu maka peneliti menggunakan strings pada dump memori yang sudah dibuat.

```
C:\home\fox\Documents\volatility3> strings 1640.dmp | grep -Fi "KB00207877.exe"
KB00207877.exe
C:\Documents and Settings\Robert\Application Data\KB00207877.exe(,
KB00207877.EXEn
KB00207877.exe
KB00207877.exe
C:\Documents and Settings\Robert\Application Data\KB00207877.exe(,
C:\home\fox\Documents\volatility3>
```

Gambar 15. Informasi Hasil Malware

Terlihat pada gambar diatas bahwa KB00207877.exe terhubung dengan *pid* 1640 dan dapat disimpulkan bahwa korban dapat menghapus file tersebut dan dengan begitu malware tidak akan muncul lagi.

V. KESIMPULAN

Penggunaan Volatility sebagai alat analisis forensik memori terbukti sangat berguna dalam menganalisis memory dump yang terinfeksi malware. Dengan menggunakan berbagai perintah seperti imageinfo, pslist, pstree, connscan, dan hivelist, serta plugin-plugin lainnya, analis dapat mengidentifikasi proses-proses, hubungan hierarki, koneksi jaringan, dan entri registry yang terkait dengan infeksi malware. Melalui tahapan analisis yang terstruktur dan dokumentasi yang baik, Volatility memungkinkan pemahaman yang lebih mendalam tentang perilaku malware,

aktivitas jaringan yang mencurigakan, dan modifikasi sistem yang dilakukan oleh malware.

DAFTAR PUSTAKA

[1] Michael Solomon, Diane Barrett, Neil Broom. (2005), "Computer Forensics Jumpstart", Alameda, SYBEX Inc.

[2] Adestein. Frank, 2006, "Live Forensics – Diagnosing Your System Without Killing It First ", Communication of The ACM February 2006/Vol 49.

[3] B. Raharjo, "SEKILAS MENGENAI FORENSIK DIGITAL," Jurnal Sosioteknologi, 2013

[4] E. S. Wijaya and Y. P. , "Integrasi Metode Steganografi DCS pada Image dengan Kriptografi Blowfish sebagai Model Anti Forensik untuk Keamanan Ganda Konten Digital," 2015.

[5] R. U. Putri and J. E. Istiyanto, "Analisis Forensik Jaringan Studi Kasus Serangan SQL Injection pada Server Universitas Gadjah Mada," 2012.

[6] M. N. Al-Azhar, Digital Forensic : Panduan Praktis Investigasi Komputer, Jakarta: Salemba Infotek, 2012.

[7] K. E. Pramudita, "Brute Force Attack dan Penerapannya pada Password Cracking," p. 1, 2010

[8] J. A. Lapso, G. L. Peterson, and J. S. Okolica, "Whitelisting system state in windows forensic memory visualizations," Digital Investigation, vol. 20, 2017, doi: 10.1016/j.diin.2016.12.002.

[9] Tomer Teller, Adi Hayon, "Enhancing Automated Malware Analysis Machines with Memory Analysis" , Blackhat Arsenal , pp. 1-5, 2014.

[10] "Oracle, "http://www.oracle.com/technetwork/articles/java/index-137868.html, accessed May 30, 2018.