Hunting advanced volatile threats using memory forensics

Priya B Gadgil
p.gadgil@somaiya.edu
K. J. Somaiya College of Engineering, Mumbai, Maharashtra

Sangeeta Nagpure
sangeetanagpure@somaiya.edu
K. J. Somaiya College of Engineering, Mumbai, Maharashtra

ABSTRACT

Due to continuous growth in malware attacks, memory forensics has become very crucial as it contains many forensic artifacts that cyber forensic investigators cannot get through the traditional disk forensics. Forensic Analysis of a memory dump of victim’s machine provides a detailed analysis of malware, checking traces of malware that have been created while running in the machine. Moreover, recent malware techniques also use stealthy methods to go undetected in typical disk forensics. Such techniques always execute exclusively from the memory or hide in the legitimate process to avoid the typical signature-based antivirus detection. Many of the recent studies also show that the percentage of such attacks have increased drastically. It is also estimated that the same trend will continue in the future and advanced threat like fileless malware will become the major concern for the organizations as well as security researchers. This paper analyses memory forensics in the context of designing a forensic approach which will help to detect such advance malware threats. In this paper, we are analyzing a sample memory image infected by a malware. This paper proposes a generalized framework for doing step by step analysis of memory image for detecting fileless malware attacks.

Keywords — Memory forensics, Advanced volatile threat

1. INTRODUCTION

Computer forensic involves acquiring, processing, analyzing digital information and traces which can be used as evidence in administrative, civil or legal cases. Typically while working in the field of cyber forensics investigations one needs to unplug the suspect machine, make a forensically sound duplicate copy and work on the same copy. It is because the basic principle of digital forensics says ‘Never work on the Original Media’. Attackers carry out lifeless attacks that don’t drop malware on a victim’s system in order to work, and so easily evade detection. So to analyze these types of fileless attacks, one needs to carry out Memory Forensics.

In memory forensics, we need to capture the memory of the live suspect machine and analyses it to get the crucial information about the system. In this paper, one sample memory image of the infected system is analyzed using Memory Forensic tool ‘Volatility’. This is to demonstrate an analytical approach which can be helpful in detecting the advanced volatile threats. The main objective of our approach would be to find out the ‘Indicators of the Compromise’ (IOC’s) for the conformity of the malware detection.

2. RESOURCES NEEDED

(i) Volatility Memory Forensic tool installed.
(ii) Captured Memory of the suspected machine.

3. DEMONSTRATION

A sample memory image is chosen to demonstrate how memory forensics can be useful in digging the traces of the malware. In the example, we will be analyzing to get enough Indicators of Compromises (IOC’s). Using the tool ‘Volatility’ following analysis was performed on the disk image.

Methodology for memory analysis for advanced volatile search as follows using volatility:

1. Identify Rogue Processes
   a. Pslst
   b. Psscan
   c. Pstree
   d. Pstotal
2. Analyze process DLLs and handles
   a. Dlllist
   b. Cmdline
   c. Getsids
   d. Handles
   e. Filescan
   f. Mutantscan
   g. Svcscan
   h. Cmdscan
   i. Consoles

3. Review network artefacts
   a. Connections
   b. Connscan
   c. Sockets
   d. Sockscan
   e. Netscan

4. Look for evidence of code injection
   a. Malfind
   b. Ldrmodules

5. Checks for signs of a rootkit
   a. SSDT
   b. Psxview
   c. Modscan
   d. Apihooks
   e. Driverirp
   f. Idt

6. Dump suspicious processes and drivers
   a. Dlldump
   b. Moddump
   c. Procdump
   d. Memdump
   e. Dumpfiles
   f. filescan

Above is the general framework that we can use to dig deep into the search of malware especially the volatile threats. To detect such threats is very challenging sometimes as it evades itself from many antivirus solutions as well as network level protection solutions. For the demonstration, only relevant output snaps are shown to prove the maliciousness of the captured memory. We should also note that every plugin listed above in each section will not work for every memory capture as some of the plugins will be the ‘profile specific’.

3.1 Step 1. Know the profile of the machine
It is very important first to determine the Profile (Operating system) of the system from where the memory image was captured, as the memory structure is different for different Windows Operating System.

The image info plugin identifies the Windows operating system version, the service pack, and the system architecture by locating the KDBG (Kernel Debugging Data Block) within the memory image.

The image info plugin also shows the date and time when the memory sample was collected.
Multiple profiles are suggested because the identified operating systems share many features that are common.

Using the Suggested Profiles and the Image Type field in the output, we can summaries that the correct profile to use is WinXPSP2x86.

3.2 Step 2: Collect all the processes list for the machine
Collecting the processes those were running on the machine is the first step in the memory forensics to start the analysis as it will give us the information about all the running tasks in the memory.

```
\$ pslist -ppid winxp2sp2x86
```

3.3 Step 3: Dig dip into the analysis to find out the indicators of compromise
We will start to search the Indicators by identifying the parent and the child process. Even with the help of a process list, it is very difficult to know the parent process for the specific child process.

```
\$ pstree
```

Processes are arranged in the logical manner i.e. Parent process is on the Top of the list. And corresponding child processes are listed below. With this output we can check from the process list parameters are appropriate.

Now let’s check whether there are any terminated processes are present in the memory with the help of `psxview`.

```
\$ psxview
```

From the above output, we can say that none of the processes was terminated as all the processes are turning up in the `pslist` as well as `psxscan` output. The terminated processes would have flagged as false in the process list. That means such processes were terminated before capturing the memory of the machine.
The Next step is to check whether any process is trying to connect the remote IP's. Command connscan will help us to find such a process.

From the above results the important points that should be noted are:

- Process with PID 888 is svchost.exe with Parent ID 656 i.e. services.exe

The next step is to check memory for suspicious dll's to check the probable hooking available in the victim machine inserted by the malicious process. Command dlllist displays all the available dll in the memory. Manual inspection is needed to check whether any suspicious dll is present or not.

The output of the dlllist is quite big. In this example, we get suspicious dll entry as dll.dll in the path C:\WINDOWS\system32\Since we have got suspicious dll entry which needs to be checked in detail later investigation with some more dll related commands will help us to find the traces of the infected part of the memory and the corresponding processes. The command ldrmodules detects the unlinked dll in the memory. Although the output of this command is very large the quick inspection can lead us to get the right traces.

The command ldrmodules gives the two suspicious readings in this case. One suspicious dll name in a system32 path and one temporary file that might have created when malware was running in the memory of the machine.

4. ANALYSIS AND RESULTS

From the rough Analysis, we have collected the enough IOC’s to point the malware infection. Further memory analysis will also lead us to trace the complete path of the memory and exact location in the memory. The following are the Indicators of malware Infections in the victim machine:
- Process with PID 888 is svchost.exe with Parent ID 656 i.e. services.exe
- Local IP 192.168.30.128 is connecting to itself through process ID 1980.
- Presence of suspicious dll name “gaopdtxmsflaavppfmgkbskshvxtlvrjypj.dll” in ldrmodules command confirms the detached dll which might have run on the system. The dll is again associated with PID 888(svchost.exe).

Now, to confirm the results from the experimental demonstration let’s check the details of the remote IP addresses as well as malicious dll’s. It is a possibility that remote IP’s can be any legitimate application server IP.

Two engines from the virus total detected this is the malicious IP. If we see the registration details of the IP from the whois database. When remote connection IP address checked on the whois database we got the following results:

From the results, it is clear that IP is registered in the remote country and further analysis with threat intelligent solution leads us to the conclusion that IP address is related to malicious activities. We can get the similar conclusion after analyzing dumped suspicious dll and temp file on the online solution which supports the malware infection to the machine.

We can define the detection flow chart for such memory analysis for stealthy volatile attacks.

---

**Fig. 1: Volatile threat detection flow**

Capture the Memory

Get all the Process List

Are all critical processes Are appropriate?

Yes

Find out the dll list and tracing for hooks. And also examine the open connection, remote connection.

No

Run process related command to find suspicious process also get the network anomalies, suspicious process.

Review for rootkits or code injection techniques. Run hook finding commands like ophook, hhookshow,

Get the Final Result indicating IOC’s

Continue Traces of Malware using malware specific search like malfind, yarn ACM
5. CONCLUSION

Memory Forensics can be extremely useful in conducting the stealthy volatile attacks which many a time reside only in memory or exclusively run from the machine memory. It can also modify the registry values which sometimes makes it be persistent even after the reboot of the machine. The example demonstrated shows us that examining the available process list is the very first thing investigation can be start off with. Apart from the process list, there are some other aspects who plays a very important role in analyzing such volatile malware threats. E.g. analyzing the open ports, examining remote connections, detailed examination of dll for hooking techniques etc. Indicators at these basic steps will guide investigators about the places those needs to be actually searched for the malware traces.

6. FUTURE SCOPE

Although we have presented a general brief flow and approach with which we can proceed with the analysis for finding out traces of advanced volatile threats residing in the memory or registry the use of the command line tool ‘volatility’ requires the knowledge of command line tool as well as knowledge and methodology of static and dynamic malware analysis. Some work can be done in order to help the cyber investigators in detecting and analyzing malware from the RAM dump of the machine. Also, analysis using volatility needs all the commands to be remembered with the appropriate command format. Automating tasks up to some extent will help cyber investigators a lot in the detecting advanced volatile threats.

7. REFERENCES