Experiment No.4
To Perform analysis of volatile memory to detect evidence of attack
Date of Performance:
Date of Submission:

Aim: To perform analysis of Volatile memory to detect evidence of attack

Objective: To make use of volatility tool and its plugin to extract the evidence from the dumped volatile memory image of Windows based computer system

Theory:

Volatility is an open-source memory forensics framework for incident response and malware analysis. This is a very powerful tool and we can complete lots of interactions with memory dump files, such as:

- List all processes that were running.
- List active and closed network connections.
- View internet history (IE).
- Identify files on the system and retrieve them from the memory dump.
- Read the contents of notepad documents.
- Retrieve commands entered into the Windows Command Prompt (CMD).
- Scan for the presence of malware using YARA rules.
- Retrieve screenshots and clipboard contents.
- Retrieve hashed passwords.
- Retrieve SSL keys and certificates.

The volatility plugins

The volatility tool supports various plugins in order to extract the information from dumped volatile memory images of Windows, Linux and Mac based computer system. Following is a list of few plugins for working with Windows based computer system dumped volatile memory.

Registry Analysis Plugins

1. hivelist : find and list available registry hives
2. hivedump: print all keys and subkeys in a hive
3. printkey: output a registry key , subkeys and values
4. dumpregistry: extract all available registry hives

5. userassist: find and parse userassist key values
6. hashdump: dump user NTLM and Lanman hashes
7. autoruns: shows autoruns key related information

Process related Plugins

1. pslist: High level view of running process
2. psscan: scan memory for EPROCESS blocks
3. pstree: Display parent-process relationship

Code injection related Plugins

1. malfind: Find injected code and dump sections
2. ldrmodules: Detect unlinked DLLs
3. hollowfind: Detect process hollowing techniques

Analyze process DLLs and Handles related Plugins

1. dlllist: List of loaded dlls by process
2. getsids: print process security identifiers
3. handles: list of open handles for each process

Process, Drivers, and Objects related Plugins

1. dlldump: Extract dlls from specific processes
2. moddump: Extract kernel drivers
3. procdump: Dump process to executable sample
4. memdump: Extract every memory sections into one file
5. filescan: scan memory for FILE_OBJECT handles
6. dumpfiles: Extract FILE_OBJECTS from memory
7. svcscan: scan for Windows service record structures
8. cmdscan: scan for command_history buffers
9. consoles: scan for CONSOLE_INFORMATION output

Rootkit related Plugins

1. psxview: find hidden process using cross-view
2. modscan: scan memory for loaded, unloaded and unlinked drivers
3. apihooks: find API/DLL function hooks
4. ssdt: Hooks in system service descriptor table
5. driverirp: identify I/O request packet (IRP) hooks
6. idt: display interrupt descriptor table

Process:

Step 1. Create a dump of the running computer system using dump tools such as dumpIt

Step 2. Install volatility on your computer system or copy the downloaded volatility-master directory onto a location of your computer

Step 3. Open the command terminal

Step 4. Go to the specific location where you copied volatility-master

Step 5. Enter the volatility-master through command prompt

Step 6. Use the command in the following format to work with the dumped memory image

```
>> vol.py command –f  dumped_file_name
```

Conclusion:

In a digital forensics investigation, extracting evidence from dumped volatile memory using the **Volatility** tool provides crucial insights into system processes, active network connections, running applications, and potential malicious activities. Information such as passwords, encryption keys, or running malware can be retrieved, aiding in reconstructing events or identifying suspects. By analyzing the volatile memory image, investigators can gather time-sensitive evidence that is critical to solving the case.