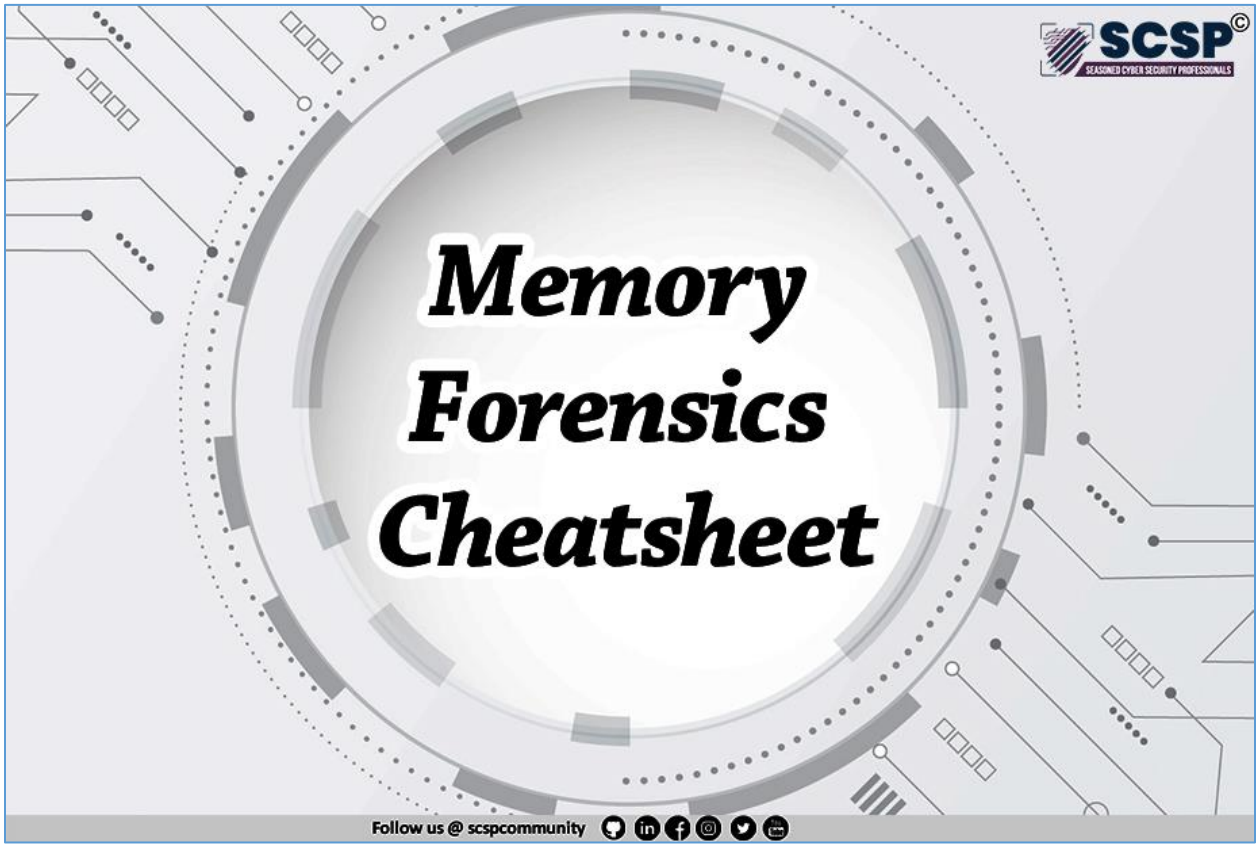
SCSP©
SEASONED CYBER SECURITY PROFESSIONALS
Memory Forensics Cheatsheet
Follow us @ scspcommunity

▶ Analyzing Windows Memory dumps to identify any malicious artifacts

Identify Indicators of Compromise (IoCs)

- Look out for any Rogue Processes
- Examine Process Objects
- Find any Code injection indicators
- Dump suspicious processes
- Monitor any unusual network activity
- Check registry entries

▶ Dumping Windows OS Memory for analysis

Use DumpIt tool to create memory dump

Extract information about memory images like profile etc.

$ vol.py -f <memory_dump_file> imageinfo

To list all processes and identify any hidden processes

$ vol.py -f <memory_dump_file> --profile=
<Identified_Profile> pslist/psscan

▸ Extracting Artifacts

To list all network connection states and verifying which network connections are made by valid processes

```
$ vol.py -f <memory_dump_file> --profile=<Identified_Profile>
      netscan/connscan
```

Printing memory addresses and paths of registry hives

```
$ vol.py -f <memory_dump_file> --profile=<Identified_Profile> hivelist
```

Printing values of a specific registry key

```
$ vol.py -f <memory_dump_file> --profile=<Identified_Profile>
      printkey -K "<path\to\key\value>"
```

▶ Identifying malicious processes

Checking for any potential code or DLL injections using built-in Volatility module

$ vol.py -f <memory_dump_file> --profile=<Idenitified_Profile> malfind

Dumping suspicous looking processes

$ vol.py -f <memory_dump_file> --profile=<Identified_Profile> procdump -D "<path\to\dump\dir>" -p <PID>

Comparing YARA rules against suspicious processes

$ vol.py -f <memory_dump_file> --profile=<Identified_Profile> yarascan -p <PID> -y "<rules.yara>"

Checking dumped executable against an Anti-Virus like ClamAV. (You may also upload it to Virustotal)

$ clamscan </path/to/file>