Texas Tech University

A Course on Digital forensics

Memory Forensics

Module 3 – The Volatility Framework

Akbar S. Namin, Spring 2018

ASSIGNMENT-2

Please perform each step of the assignment. Get screenshot of each step you do, write your explanation for the step-4, and upload your file "YourSurname_Assignment2"under assignment link.

1. Install Volatility and the dependency libraries (unless you're working with the standalone version).

2. Run the "vol.py --info" command. What profiles does your version support?

3. Run the "vol.py --help" command with and without a plugin name. How does the output differ?

The correct answer is: You'll see the plugin-specific options

4. Run the kdbgscan plugin against a Windows memory sample.
   A) What profile does it suggest?
   B) What is the virtual address of the kernel debugger data structure?
   C) Were any inaccurate profiles suggested? Why or why not?

The virtual address will be shown on the line with "Offset(V)"
In some cases, you'll see slightly inaccurate profiles (for example Win7 SP1 versus Win7 SP0) because the OS data structures look similar. In these cases, look at the Service Pack value in the kdbgscan output to determine which suggestion is correct.

5. Using the profile you determined in step 4, list processes in your memory dump. Then run the same plugin again, but redirect output to a text file so you can save it for later analysis.

6. Perform the following steps:
   A) Copy exampleplugin.py into volatility/plugins
   B) Edit exampleplugin.py and change the name from ExamplePlugin to a name of your choice
   C) Edit the description of the plugin
   D) Edit the plugin to print the process ID (UniqueProcessId) in addition to the process name

E) Run "vol.py --info" and see if your new plugin is registered
F) Run your new plugin and observe the output
G) Add a new method named render_csv (comma separated values) to the plugin and configure it to output data in CSV format
H) Run the plugin with --output=csv and observe the output

The correct answer is: (see below for an example)

```
import volatility.utils as utils
import volatility.commands as commands
import volatility.win32.tasks as tasks

class MyNewPlugin(commands.Command):
    """This is an example plugin that I modified"""

    def calculate(self):
    """This method performs the work"""

        addr_space = utils.load_as(self._config)
        for proc in tasks.pslist(addr_space):
            yield proc

    def render_text(self, outfd, data):
        """This method formats output to the terminal.
        :param outfd | <file>
        data | <generator>
        """

        for proc in data:
            outfd.write("Process:          {0}          PID          {1}\n".format(proc.ImageFileName,
proc.UniqueProcessId))

    def render_csv(self, outfd, data):
        for proc in data:
            outfd.write("{0},{1}\n".format(proc.ImageFileName, proc.UniqueProcessId))
```