Lab_14_1 (GRADED): CF2 – Volatile Memory

Otago Polytechnic, IN618 Security, Semmester 1 – 2019

1 Lab Introduction

In the exercises today, we will start learning more about Computer Forensics by performing a collection of volatile data from Random Access Memory (RAM). Then we will analyse the collected data and see what type of analysis techniques and tools are available to make sense of the data collected from RAM. We will start by performing a collection of the RAM on our Windows 10 machine, to understand how to collect, then analyse RAM data. Then we will perform an analysis on a previously collected RAM dump in an attempt to determine events that have occurred on an investigation target.

2 Lab Preparation

We will start the lab by downloading the required files. Copy the `resources_14_1` folder from the `I:` drive to the Desktop of your Windows 10 machine:

- `I:\COURSES\ITP\BITY2\IN618Security\week14\resources_14_1`

Open the `resources_14_1` folder. Inside are a variety of resources:

- `Imager_Lite_3.1.1`: The *FTK Imager* tool used for dumping RAM contents
- `volatility-2.6.1`: The *volatility* tool used for RAM analysis
- `distorm3-3.3.4.win-amd64.exe`: The `distorm3` package, a *volatility* dependency
- `system_troubles.mem`: A RAM dump for a hacking scenario. Note: this memory dump has been copied into the `volatility-2.6.1` folder
- `strings64.exe`: A Windows version of the popular `strings` command found in Linux operating systems

3 Collecting Volatile Memory (aka RAM Dump)

We will start the lab by collecting the contents of the volatile memory (RAM) of our Windows 10 system. This will provide us with experience in collecting digital evidence in the scenario of live incidence response. Before we start, we should generate some interesting content on our system. These actions will allow us to find more information when later analysing our memory dump. Perform the following tasks, opening various applications. For each application you open, **leave the application window open.**

- Open Firefox and browse to `http://google.com`, leave the window open
- Open Google Chrome and browse to `http://op.ac.nz`, and leave the window open
- Open a couple of other applications, choose from the following list
- Notepad++
- Snipping tool
- Calculator
- Microsoft Excel
- PuTTY
- And anything else you can find

We are now going to perform a RAM dump. Open the `resources_14_1` folder. Inside is another folder called `Imager_Lite_3.1.1`. Open the folder and review the contents. This folder contains all the files needed to run a computer forensics tool called *FTK Imager* – developed by a company called by Access Data. The tool is portable, meaning it does not need to be installed. The tool can perform forensic data collection of different types of media -– in today's lab we will use it to collect the RAM of our Windows 10 system.

Inside the `Imager_Lite_3.1.1` folder, double click the `FTK Imager.exe` file to start the program. You will be prompted to allow administrator access. Have a look around the application, making sure to look at the options available in each of the menus. If you want some help, check the *User Guide* under the *Help* menu.

In *FTK Imager*:

- Open the *File* menu
- Select *Capture Memory*

Check the image below for a visual guide:

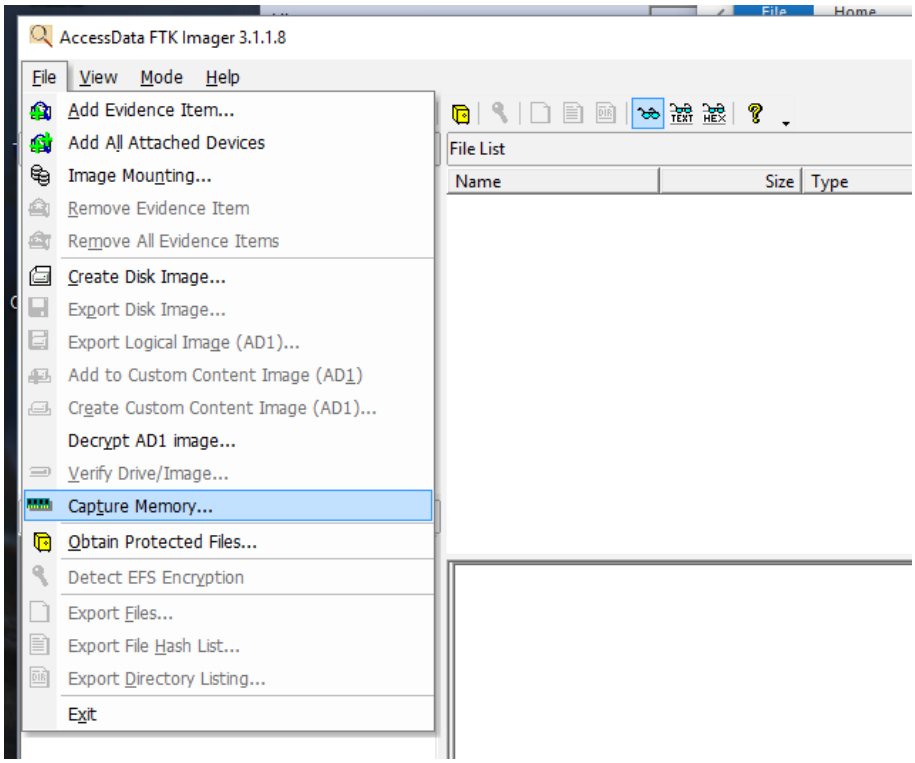

Figure 1: FTK Imager - *Capturing Volatile Memory*

A dialog box should appear, asking us for more information. Firstly, select an appropriate path to save the memory dump to... I would recommend saving the memory dump to:

```
C:\<username>\Desktop\resources_14_1\volatility-2.6.1\
```

Now, select a suitable file name, for example: `Win10RAMDump.mem`. Then click *Capture Memory* to start saving the RAM contents.

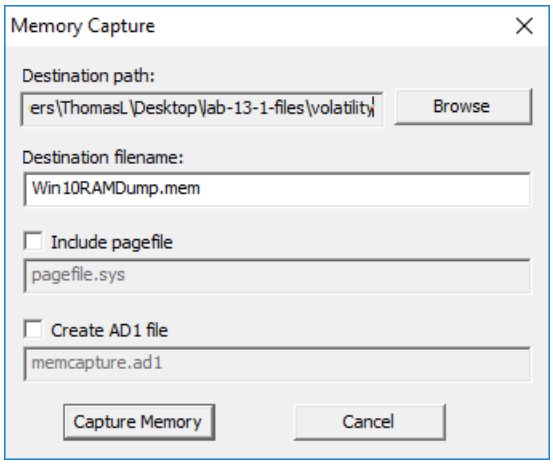

Figure 2: FTK Imager - *Capturing Volatile Memory*

It should take a little time to save the RAM. This is dependent on the capacity of the RAM on your machine. Once finished, close the dialog box.

We are going to open our RAM dump in *FTK Imager*. The tool does have the ability to open all types of files, but **it may not be advanced enough to determine the structure of the RAM dump** and provide us useful evidence. Perform the following steps to load the RAM dump in *FTK Imager*:

- Select *File* and then *Add Evidence Item*
- From the options available, select *Image File* and click *Next*
- Click *Browse* to find the memory dump file
- Finally, click *Finish* to load the file

We should see a window like that below. The different panes within the application has been labeled.

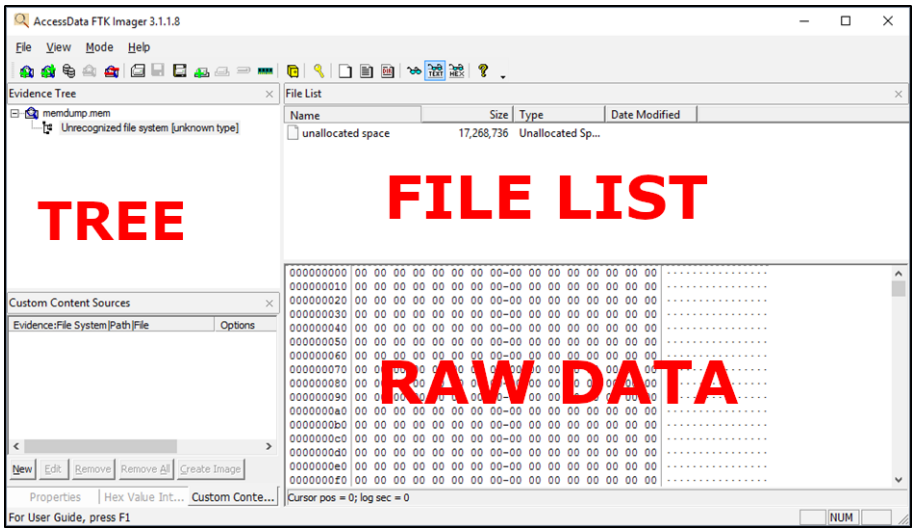

Figure 3: FTK Imager - *The User Interface*

The *Tree* pane would usually should file system information, such as folders and files. However, **RAM dumps are unstructured data** that *FTK Imager* cannot understand! Therefore, no files are shown in the *File List* pane. The only pane with useful information is the *Raw Data* pane. This pane shows the contents of the RAM dump in hexadecimal and text. We do not need the hexadecimal view. Right-click anywhere in the *Raw Data* section and select *Show Text Only*.

Scroll through the data in the *Raw Data* column. Try look for some interesting information, like URLs! You should soon realise that **this is not a good way to analyse data**. The problem is the RAM dump is not structured, and we cannot view interesting information. We are now going to try the most basic forensic analysis method: **keyword searching**.

- Right-click the *Raw Data* pane
- Select *Find...*

- This tool is exactly like any other find tool. We can search for strings!
- In the find dialog box, enter `http://google.com`

I hope that you will find some appearance of the keyword in your RAM dump. We could perform other keyword searches, but this method is not very efficient, as *FTK Imager* was not designed to perform memory analysis, rather, it was designed to perform file system analysis.

Q1. Why can FTK Imager not correct display the contents (e.g., file content) of the RAM dump?

HINT: Structure.

4 Installing Volatility

Our last analysis method for our volatile memory was not very effective! It was manual, and very time consuming. We should try another approach using a tool called *volatility*! Unfortunately, *volatility* is not easy to install and configure – this is common for advanced security and forensic tools, where more development time is spent on **functionality**, rather than **usability**! The *volatility* tool is written in the Python programming language, and we need to install some additional *libraries* so that we can use *volatility* effectively.

Start by opening Windows PowerShell as an Administrator.

- Click the *Start Menu*
- Start typing and search for `PowerShell`
- Right-click and select *Run as Administrator*

When you have PowerShell up and running, execute the following steps:

```
# Change to the Python directory
cd C:\Python27\

# Download the get-pip script, so that we can install the Python package manager
curl.exe https://bootstrap.pypa.io/get-pip.py -o get-pip.py

# Run the get-pip script, to install pip
C:\Python27\python.exe .\get-pip.py

# Finally, install the pycryptodome package
C:\Python27\Scripts\pip.exe install pycryptodome
```

The entire point on the previous steps was to install the `pycryptodome` package, so that we have access to various cryptographic methods that *volatility* uses. However, we still need one more package. Execute the following steps to configure another package named `distorm3` – another *volatility* requirement.

- Navigate to your `resources_14_1` folder
- Double click the `distorm3-3.3.4.win-amd64.exe` file

When installing the `distorm3` package... there is **one key step!** When asked what Python version to install for, make sure you **select Python Version 2.7 (found in Registry)**. Have a look at the screenshot below to confirm you are selecting the correct version...

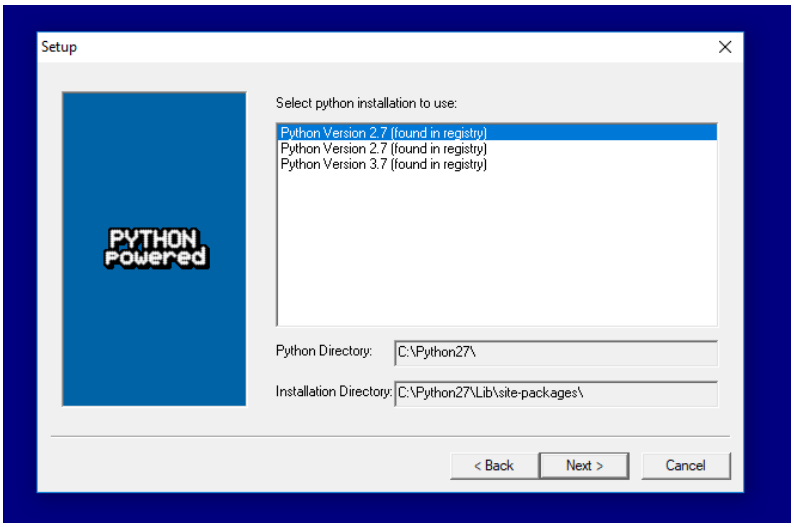

Figure 4: FTK Imager - *The User Interface*

We should now be ready to analyse volatile RAM data! To make sure you have setup your environment correctly perform the following steps:

- Return to you `PowerShell` window
- Change to the *volatility* directory
- `cd C:\Users\<username>\Desktop\resources_14_1\volatility-2.6.1`
- Use the `ls` command to make sure you are in the correct folder
- If you can see the `vol.py` file, you are in the correct location!
- Finally, run the *volatility* program using:
- `C:\Python27\python.exe vol.py --info`

When running the last command you should see the *volatility* help menu without any errors! If you are unsure that you have the correct configuration, feel free to ask the tutor before continuing...

5 Analysing Volatile Memory using Volatility

We are ready to perform some actual memory analysis! Try the following command to view the program help menu:

```
C:\Python27\python.exe vol.py --help
```

There are a large amount of options and configurations in *volatility*. Find the section in the help menu called: *Supported Plugin Commands*. Have a look over the list of available plugins.

Q2. Document three plugins that sound interesting. Include a summary of what they do. Use your own words; please do not just copy/paste the help menu.

unloadedmodules Prints a list of unloaded modules

cachedump dump cached domain hashes from memory

Now we have an idea of what plugins we could run to extract information. However, before running a plugin we need to find a profile to match our target system. RAM contents and structure vary dramatically between each Windows version so we need to find the same system in the profile list as our system. To help you determine specific information about your Windows operating system try the following command in `PowerShell`:

```
systeminfo.exe
```

Take note of the following entries:

- OS Name
- OS Version
- System type

Q3. What system did we collect the RAM dump from? Include the Windows version and architecture (e.g., 32 or 64-bit)

Microsoft Windows 10 Enterprise - 10.0.17134 N/A Build 17134 - x64-based PC

Now run the following command, locate the Profiles section and find a suitable profile that matches your answer from Question 3:

```
C:\Python27\python.exe vol.py --info
```

Q4. What profile are you going to use?

Win10x64_17134

It might be useful to check with other students or your tutor before continuing. Selecting the correct profile is very important! And there are various profiles available for Windows 10...

OK, let us extract some interesting information from the RAM dump. Try the following command to list all the processes that were running on the system when the RAM was collected. Note: You may need to replace the profile in the command below with the correct profile:

```
C:\Python27\python.exe vol.py --profile <profilename> -f Win10RAMDump.mem pslist
```

We should break the command down to understand the purpose:

- `python.exe`: the Python executable
- `vol.py`: the *volatility* executable
- `-profile <profilename>`: The selected profile. Make sure to replace `<profilename>` with the actual name of a profile
- `-f Win10RAMDump.mem`: the memory dump file
- `pslist`: the plugin we want to run, in this case the list processes plugin

You should get some output that is very similar to the information displayed in Windows Task Manager when we view running processes. If we were performing a forensic investigation, we should include all the output in our documentation. However, for today's exercise, select some interesting processes and document them below. Make sure to include the Process Name, Process ID (PID) and Start Time.

Q5. Document some interesting processes below. Can you identify the processes that you opened previously?

chrome.exe
putty.exe

Let us try one more example: scanning for network connections in our RAM dump. Try the following command:

```
C:\Python27\python.exe vol.py --profile <profilename> -f Win10RAMDump.mem netscan
```

Again, review the output. Look for interesting content. You could try to perform some reverse DNS lookups on the listed IP addresses to find where the connections are going. Get used to the volatility tool by running a couple of other plug-ins. Review the output and do some Google searchers to find out what the contents mean. Try talking to fellow students or the lecturer about the plug-in you are using.

Some interesting plugins are:

- `pslist`
- `pstree`
- `cmdscan`
- `cmdline`
- `netscan`

Once you feel comfortable using volatility, proceed to the next section of the lab.

6 Solving a Case using Volatility

Inside the volatility folder, there is a previously collected memory dump named:

- `system_troubles.mem`

Riveting Backstory: An undisclosed company has contacted you to perform a forensic investigation. One of the employees of the company received an email from a co-worker that contained a PDF. After opening the PDF, the employee has been experiencing problems with their bank account. A RAM dump from the employees computer has been collected after the suspected infection. You are tasked with determining what has happened! To help you out, the system the RAM dump is taken from is: **Windows XP SP2 x86**.

Q6. What volatility profile are you going to use to analyse `system_troubles.mem` file?

WinXPSP2x86

Q7. Document all running processes from the employee's system:

```
Offset(V)  Name                PID   PPID   Thds     Hnds   Sess  Wow64 Start
Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------
------------------------------
0x823c8830 System                  4      0     58      573 ------      0
0x81f04228 smss.exe              548      4      3       21 ------      0 2010-02-26
03:34:02 UTC+0000
0x822eeda0 csrss.exe             612    548     12      423      0      0 2010-02-26
```

Q8. Document the full command you used to determine the answer to question 7.

C:\Python27\python.exe vol.py --profile WinXPSP2x86 -f system_troubles.mem psl

Remember, the employee said unusual activities started happening after opening a PDF document attached to an email. Examine the list of processes – try to find something related to PDFs.

Q9. Document the process that is relevant to PDFs. Include the name, PID and Start time

HINT: Try opening Adobe Acrobat PDF reader on your Windows 10 system and look for the name of the process.

Each process has a parent process, or PPID. Find the PPID of the process you documented in answer to Question 9.

Q10. Document the parent process. Include the name, PID and Start time.

Can we determine anything about how the malicious PDF entered the employee's system? We know it was emailed to the employee. But how was the email opened by the employee...

Q11. How did the employee access the email?

Firefox

OK, now we have determined where the email has originated from. We now want to further analyse how the PDF infected the system. It is most likely that the PDF file had some sort of program that downloaded malware to the employee's system. To download, the program would need to contact an IP address or domain name...

GRADED Q12. What is the IP address that the malicious PDF contacted?

HINT: You need to run another plug-in concerned with network connections. PIDs are you friend! And in this case, PPIDs are not your friend!

We only performed a basic analysis. We could perform some much more advanced analysis on our evidence. One example, we could extract the data only associated with the specific PID we are interested in. Then we could run a variety of other tools against the data. For example, we could recover files, search for URLs etc. For your information, You can extract the data only associated with the interesting PID using volatility.

```
C:\Python27\python.exe vol.py --profile <profilename> -f system_troubles.mem
-p 1752 memdump --dump-dir .
```

In the above command, the `-p 1752` argument means to dump the data only for the process ID of 1752. We should also copy the memory dump from Firefox as well.

GRADED Q13. Document the full command for dump all the data from the Firefox process.

In the `resources_14_1` folder there is one more tool we have not yet used. The tool, named `strings64.exe`, is a Microsoft version of the popular `strings` application available in Linux. Although not designed specifically for computer forensics, it can be exceptionally useful. The tool takes any type of data input, and extracts strings in ASCII or Unicode. Therefore, we can run it against unstructured data and find strings, such as URLs, email address, or document content! **You must use PowerShell for this exercise!**

You can dump all strings from the `1752.dmp` file using the following command:

```
.\strings64.exe volatility-2.6.1\1752.dmp
```

If you ran the command above, use Ctrl + C to cancel it. We could search for unique strings using the same command, this time by adding a pipe (`|`) to redirect the output from the first command into a keyword search. A full example is provided below which searches for the keyword `cats`:

```
.\strings64.exe volatility-2.6.1\1752.dmp | Select-String "cats"
```

Use the power of the strings utility to try to find a URL that can identify the bank...

GRADED Q14. What bank does the employee use?

HINT: Try a variety of keywords! Keep them relevant to the scenario.

We could even extract the malicious PDF document using file carving. How about we save that until next week... Data recovery! Final part of the graded lab. We should determine an approximate time that the employee accessed the email.

GRADED Q15. Provide an approximate time of when the employee accessed the malicious email via the web browser.

HINT: You need a volatility plugin! You could examine processes, or another digital artifact (Some information on this page will be useful: `http://www.4n6k.com/2013/05/userassist-forensics-timelines.html` and try to find a volatility plugin that can look for these artifacts).