Memory Acquisition for Forensic Memory Analysis on Windows and Linux

Author: **Alexandre Borges**
Date: **JUN/11/2014**
revision: **A.1**

Introduction

Nowadays the digital world has been stroked byr a massive malware attack and every single day came up lots of variants of virus, spyware, Trojans and worms. It's undeniable and unavoidable: crackers are winning the war. Therefore, the Memory Forensics Analysis is one of more interesting fields in Digital Forensics and it is worth to take some time for studying. Unfortunately, some malware's attacks don't leave easily detectable traces on disk and our final hope is based on analyzing the system memory to find some clue about an existing malware. That's exactly this point where to perform a good memory acquisition for initiating a detailed analysis becomes so important.

There're some many ways and tool to dump the memory, but this simple article will show you a straight approach taking three tools for Windows system and another tool for Linux system. It's very important to highlight that all of these following procedures should be performed, when it's possible, from an external media (DVD, pen drive, etc.) and the memory dump should be saved to an external media too. Additionally, you should remember that any device inserted to computer will leave some information (Locard's exchange principle) and when handling Windows systems this procedure become even more relevant because of the Registry.

Memory acquisition on Windows System

There're two very good tools to perform the memory acquisition on Windows system: DumpIt from MoonSols (http://www.moonsols.com/downloads/7) and Memoryze from Mandiant/FireEye (https://www.mandiant.com/library/MemoryzeSetup3.0.msi). Both tools are free and they are available for downloading. Memoryze supports several

Windows versions including Windows 2012 (https://www.mandiant.com/resources/download/memoryze) as well DumpIt supports any 32-bits and 64-bits operating system.

The DumpIt is a nice tools and it's easy to handle it. You should copy the DumpIt to an external device such as a pendrive or hard disk (both should be NTFS formatted), extract it, open a Command Prompt and execute the following commands:

```
C:\> F:

F:\> cd DumpIt

F:\DumpIt>dir

 Volume in drive F is SAMSUNG
 Volume Serial Number is 3243-30C2

 Directory of F:\DumpIt

11/06/2014  13:38    <DIR>          .
11/06/2014  13:38    <DIR>          ..
03/05/2011  02:41           207.496 DumpIt.exe
18/07/2011  08:29               743 README.txt
               2 File(s)        208.239 bytes
               2 Dir(s)  571.126.833.152 bytes free

F:\DumpIt> DumpIt.exe

  DumpIt - v1.3.2.20110401 - One click memory memory dumper
  Copyright (c) 2007 - 2011, Matthieu Suiche
<http://www.msuiche.net>
  Copyright (c) 2010 - 2011, MoonSols <http://www.moonsols.com>


    Address space size:        17951621120 bytes (  17120 Mb)
    Free space size:          571126833152 bytes ( 544668 Mb)

    * Destination = \??\F:\DumpIt\EXADATA-20140611-164112.raw

    --> Are you sure you want to continue? [y/n] y
    + Processing... Success.

F:\DumpIt>dir
```

```
 Volume in drive F is SAMSUNG
 Volume Serial Number is 3243-30C2

 Directory of F:\DumpIt

11/06/2014  13:41    <DIR>          .
11/06/2014  13:41    <DIR>          ..
03/05/2011  02:41           207.496 DumpIt.exe
11/06/2014  13:44    17.951.621.120 EXADATA-20140611-164112.raw
18/07/2011  08:29               743 README.txt
               3 File(s) 17.951.829.359 bytes
               2 Dir(s)  535.191.465.984 bytes free

F:\DumpIt>
```

Amazing! DumpIt quickly has performed a raw acquisition from memory.

Another excellent tool is the Memoryze that can help us to dump the memory for a future analysis. Although the Memoryze installation package is an .msi file, we should install it on another computer (my case, Windows 7) and to copy its installation directory (C:\Program Files (x86)\Mandiant\Memoryze) to an external drive (F:\). From there, execute the following steps:

```
F:\>cd Memoryze

F:\Memoryze>dir

 Volume in drive F is SAMSUNG
 Volume Serial Number is 3243-30C2

 Directory of F:\Memoryze

11/06/2014  13:40    <DIR>          .
11/06/2014  13:40    <DIR>          ..
10/07/2013  18:55             1.598 AcquireDriver.Batch.xml
10/07/2013  18:55             1.425 AcquireMemory.Batch.xml
10/07/2013  18:55             2.043
AcquireProcessMemory.Batch.xml
10/07/2013  18:55             1.844
DriverAuditModuleList.Batch.xml
10/07/2013  18:55             3.437
DriverAuditSignature.Batch.xml
```

```
10/07/2013  18:55             2.951 DriverDD.bat
10/07/2013  18:55             5.993 DriverSearch.bat
10/07/2013  18:55             2.631 DriverWalkList.bat
10/07/2013  18:55             2.544 HookAudit.Batch.xml
10/07/2013  18:55             4.577 HookDetection.bat
10/07/2013  18:55             2.995 MemoryDD.bat
10/07/2013  20:47        11.894.576 Memoryze.exe
10/07/2013  18:55           546.029 MemoryzeUserGuide.pdf
10/07/2013  18:55             9.681 Process.bat
10/07/2013  18:55             5.570 ProcessAuditMemory.Batch.xml
10/07/2013  18:55             3.837 ProcessDD.bat
              16 File(s)     12.491.731 bytes
               2 Dir(s)  553.175.212.032 bytes free

F:\Memoryze> MemoryDD.bat

Memoryze.exe by MANDIANT (c) 2011 -
http://www.mandiant.com/products/free_software/memoryze/
   Usage: MemoryDD.bat
     -offset   optional offset into physical memory. Exclude for
all.
     -size     optional size of physical memory to acquire.
Exclude for all.
     -output   directory to write the results. Default .\Audits
```

During the last command, Memoryze should has opened a new Command Prompt Window and initiated the memory dump. After the memory acquisition has finished, execute the following commands:

```
F:\Memoryze> cd Audits

F:\Memoryze\Audits> dir

 Volume in drive F is SAMSUNG
 Volume Serial Number is 3243-30C2

 Directory of F:\Memoryze\Audits

11/06/2014  13:51    <DIR>          .
11/06/2014  13:51    <DIR>          ..
11/06/2014  13:51    <DIR>          EXADATA
               0 File(s)              0 bytes
               3 Dir(s)  535.223.562.240 bytes free
```

```
F:\Memoryze\Audits> cd EXADATA

F:\Memoryze\Audits\EXADATA> dir

 Volume in drive F is SAMSUNG
 Volume Serial Number is 3243-30C2

 Directory of F:\Memoryze\Audits\EXADATA

11/06/2014  13:51    <DIR>          .
11/06/2014  13:51    <DIR>          ..
11/06/2014  13:51    <DIR>          20140611165146
               0 File(s)              0 bytes
               3 Dir(s)  535.223.562.240 bytes free

F:\Memoryze\Audits\EXADATA> cd 20140611165146

F:\Memoryze\Audits\EXADATA\20140611165146> dir
 Volume in drive F is SAMSUNG
 Volume Serial Number is 3243-30C2

 Directory of F:\Memoryze\Audits\EXADATA\20140611165146

11/06/2014  13:51    <DIR>          .
11/06/2014  13:51    <DIR>          ..
11/06/2014  13:51            20.056 BatchResults.xml
11/06/2014  13:51               283 Issues.BatchResults.xml
11/06/2014  13:55             2.172
issues.memory.4d021d38.img.xml
11/06/2014  13:55    17.951.621.120 memory.4d021d38.img
               4 File(s) 17.951.643.631 bytes
               2 Dir(s)  535.223.562.240 bytes free
```

Perfect! Memoryze has acquired the memory without presenting any problems. If some issues have happened during the dump, Memoryze saves xml files for post examination.

Mandiant offers a software named Redline (https://www.mandiant.com/library/Redline-1.12.msi) that can help us with an initial analysis. For use it, we should install it on a Windows system (Win7, for example) and afterwards run it:

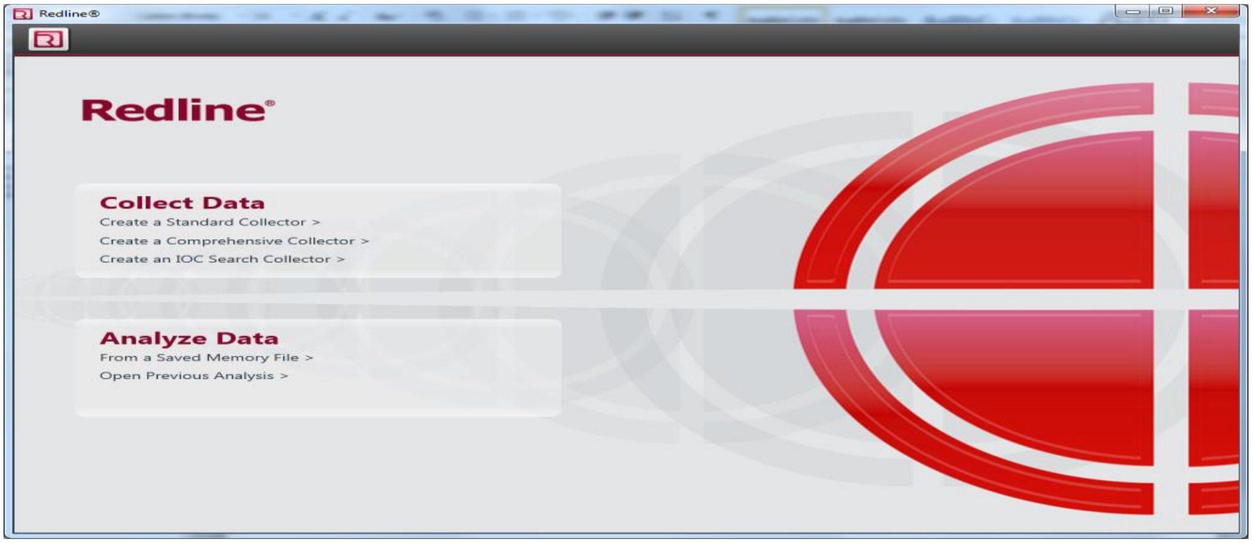

Redline®
Redline®
Collect Data
Create a Standard Collector >
Create a Comprehensive Collector >
Create an IOC Search Collector >
Analyze Data
From a Saved Memory File >
Open Previous Analysis >

Figure 1

Choose **Analyze data → From a Saved Memory File** menu:

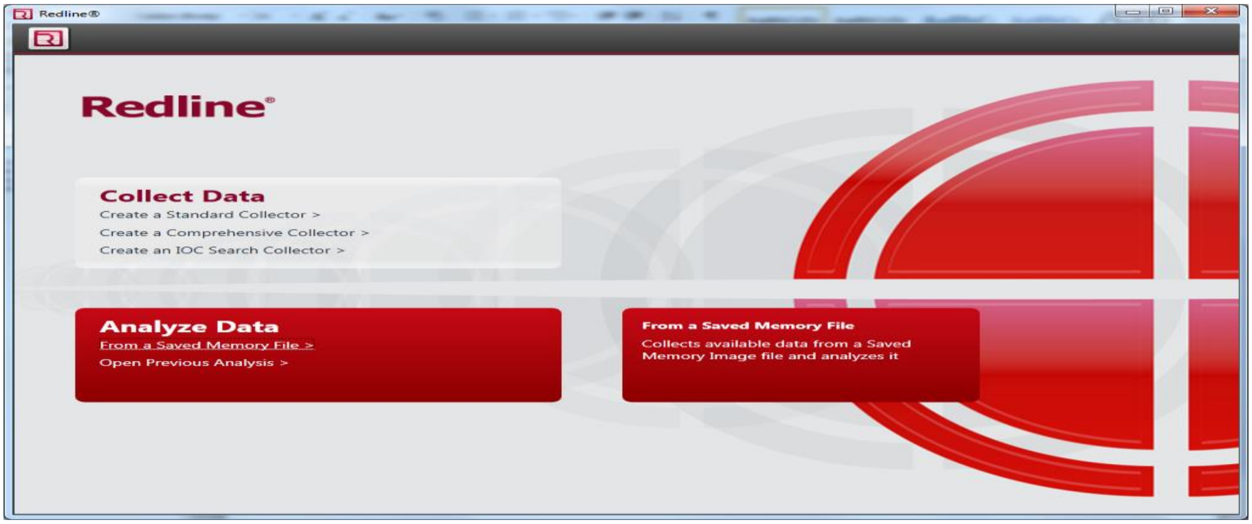
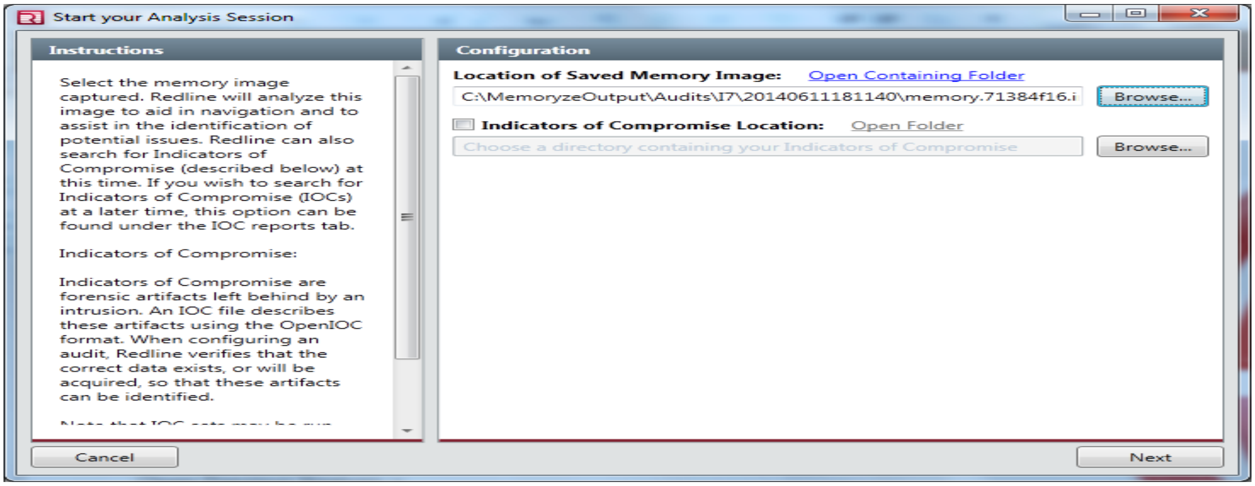

Redline®
Redline®
Collect Data
Create a Standard Collector >
Create a Comprehensive Collector >
Create an IOC Search Collector >
Analyze Data
From a Saved Memory File >
Open Previous Analysis >
From a Saved Memory File
Collects available data from a Saved Memory Image file and analyzes it

Figure 2

Browse and choose the saved memory image by Memoryze:



Start your Analysis Session

Instructions

Select the memory image captured. Redline will analyze this image to aid in navigation and to assist in the identification of potential issues. Redline can also search for Indicators of Compromise (described below) at this time. If you wish to search for Indicators of Compromise (IOCs) at a later time, this option can be found under the IOC reports tab.

Indicators of Compromise:

Indicators of Compromise are forensic artifacts left behind by an intrusion. An IOC file describes these artifacts using the OpenIOC format. When configuring an audit, Redline verifies that the correct data exists, or will be acquired, so that these artifacts can be identified.

Note that IOC sets may be run

Configuration

Location of Saved Memory Image: Open Containing Folder

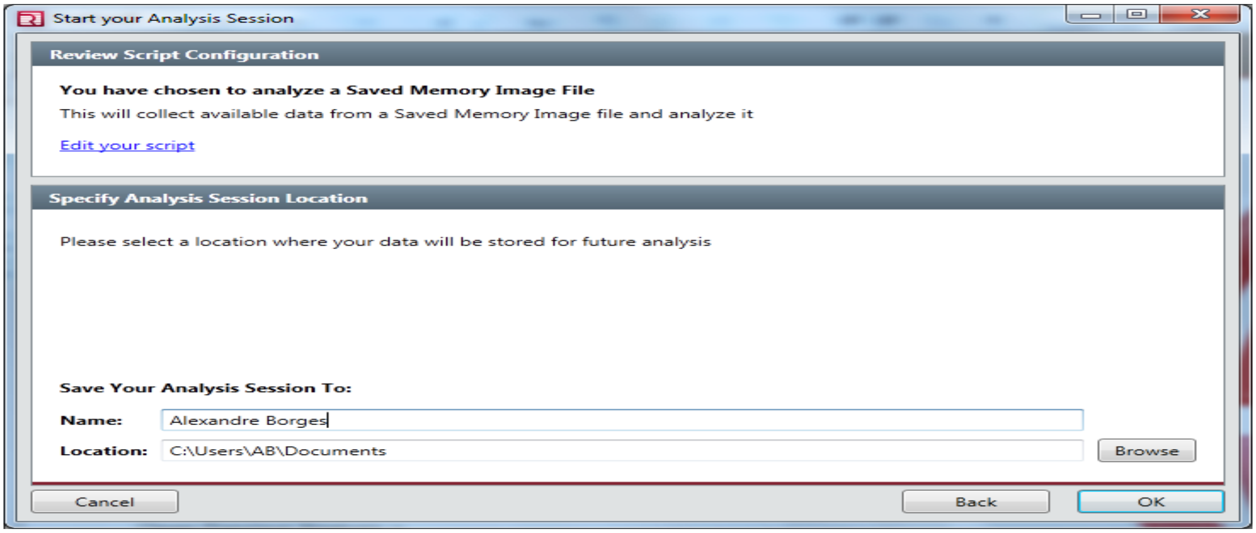
C:\MemoryzeOutput\Audits\I7\20140611181140\memory.71384f16.i

□ **Indicators of Compromise Location:** Open Folder

Choose a directory containing your Indicators of Compromise

Figure 3

Assign a name to the session and click on **OK**:



Start your Analysis Session

Review Script Configuration

You have chosen to analyze a Saved Memory Image File

This will collect available data from a Saved Memory Image file and analyze it

Edit your script

Specify Analysis Session Location

Please select a location where your data will be stored for future analysis

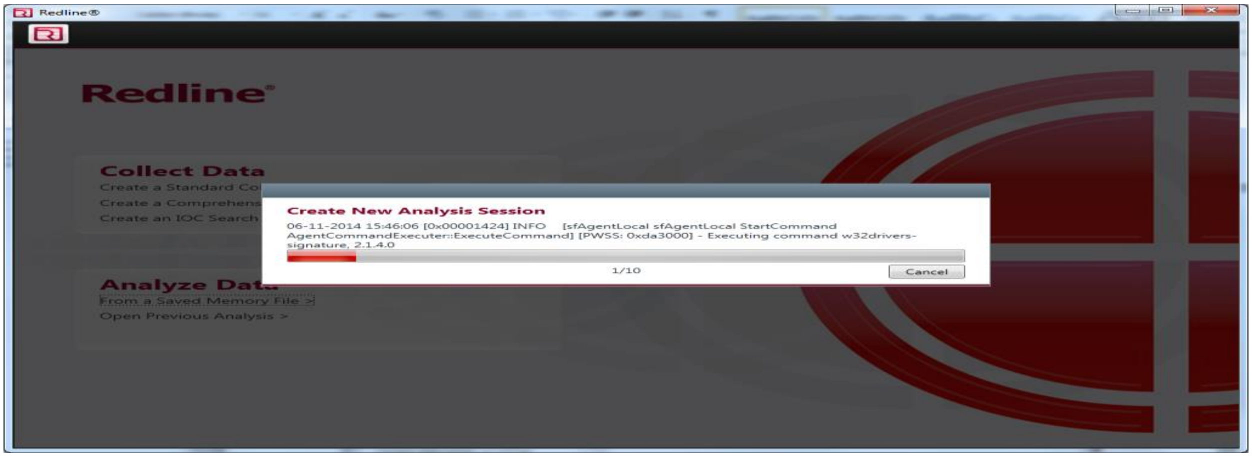
Save Your Analysis Session To:

Name: Alexandre Borges

Location: C:\Users\AB\Documents

Figure 4

Wait for few minutes (in my case, it has taken more than 20 minutes….):

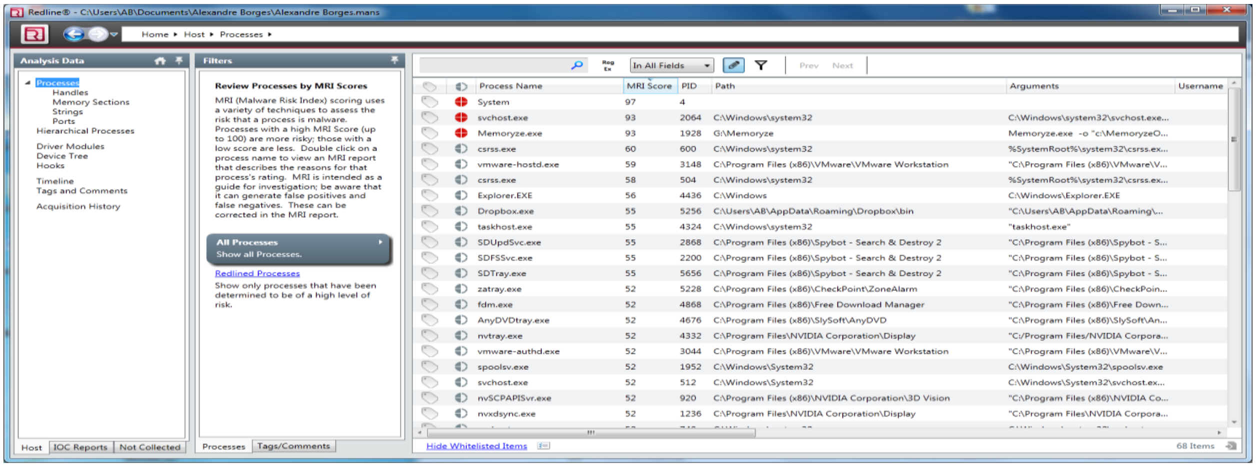


Create New Analysis Session

```
06-11-2014 15:46:06 [0x00001424] INFO    [sfAgentLocal sfAgentLocal StartCommand
AgentCommandExecuter::ExecuteCommand] [PWSS: 0xda3000] - Executing command w32drivers-
signature, 2.1.4.0
1/10
```

Figure 5

We've finished it! Now Redline offers many possibilities. We're able to exam Processes, Hooks, Driver Modules, Timeline events, Memory Sections, Handles:



Review Processes by MRI Scores

MRI (Malware Risk Index) scoring uses a variety of techniques to assess the risk that a process is malware. Processes with a high MRI Score (up to 100) are more risky; those with a low score are less. Double click on a process name to view an MRI report that describes the reasons for that process's rating. MRI is intended as a guide for investigation; be aware that it can generate false positives and false negatives. These can be corrected in the MRI report.

Show only processes that have been determined to be of a high level of risk.

		Process Name	MRI Score	PID	Path	Arguments	Username
	●	System	97	4			
	●	svchost.exe	93	2064	C:\Windows\system32	C:\Windows\system32\svchost.exe...	
	●	Memoryze.exe	93	1928	G:\Memoryze	Memoryze.exe -o "c:\MemoryzeO...	
	◐	csrss.exe	60	600	C:\Windows\system32	%SystemRoot%\system32\csrss.ex...	
	◐	vmware-hostd.exe	59	3148	C:\Program Files (x86)\VMware\VMware Workstation	"C:\Program Files (x86)\VMware\V...	
	◐	csrss.exe	58	504	C:\Windows\system32	%SystemRoot%\system32\csrss.ex...	
	◐	Explorer.EXE	56	4436	C:\Windows	C:\Windows\Explorer.EXE	
	◐	Dropbox.exe	55	5256	C:\Users\AB\AppData\Roaming\Dropbox\bin	"C:\Users\AB\AppData\Roaming\...	
	◐	taskhost.exe	55	4324	C:\Windows\system32	"taskhost.exe"	
	◐	SDUpdSvc.exe	55	2868	C:\Program Files (x86)\Spybot - Search & Destroy 2	"C:\Program Files (x86)\Spybot - S...	
	◐	SDFSSvc.exe	55	2200	C:\Program Files (x86)\Spybot - Search & Destroy 2	"C:\Program Files (x86)\Spybot - S...	
	◐	SDTray.exe	55	5656	C:\Program Files (x86)\Spybot - Search & Destroy 2	"C:\Program Files (x86)\Spybot - S...	
	◐	zatray.exe	52	5228	C:\Program Files (x86)\CheckPoint\ZoneAlarm	"C:\Program Files (x86)\CheckPoin...	
	◐	fdm.exe	52	4868	C:\Program Files (x86)\Free Download Manager	"C:\Program Files (x86)\Free Down...	
	◐	AnyDVDtray.exe	52	4676	C:\Program Files (x86)\SlySoft\AnyDVD	"C:\Program Files (x86)\SlySoft\An...	
	◐	nvtray.exe	52	4332	C:\Program Files\NVIDIA Corporation\Display	"C:/Program Files/NVIDIA Corpora...	
	◐	vmware-authd.exe	52	3044	C:\Program Files (x86)\VMware\VMware Workstation	"C:\Program Files (x86)\VMware\V...	
	◐	spoolsv.exe	52	1952	C:\Windows\System32	C:\Windows\System32\spoolsv.exe	
	◐	svchost.exe	52	512	C:\Windows\System32	C:\Windows\System32\svchost.ex...	
	◐	nvSCPAPISvr.exe	52	920	C:\Program Files (x86)\NVIDIA Corporation\3D Vision	"C:\Program Files (x86)\NVIDIA Co...	
	◐	nvxdsync.exe	52	1236	C:\Program Files\NVIDIA Corporation\Display	"C:\Program Files\NVIDIA Corpora...	

Figure 6

I'm sure that Readline is a very complete tool for an initial analysis of a memory dump acquired by Memoryze.

For completeness, the FTK Imager from Access Data (the FTK imager 3.1.4 is available from http://www.accessdata.com/support/product-downloads) is a nice tool for postmortem forensic analysis, but it's also able to make a perfect dump from memory. The only disadvantage is that it must be installed in advanced in the machine where the memory will be captured. If you want to run the procedure from a pen drive, you need to purchase the Live Response product (http://www.accessdata.com/products/digital-forensics/live-response).

Continuing with our explanation, execute the FTK:

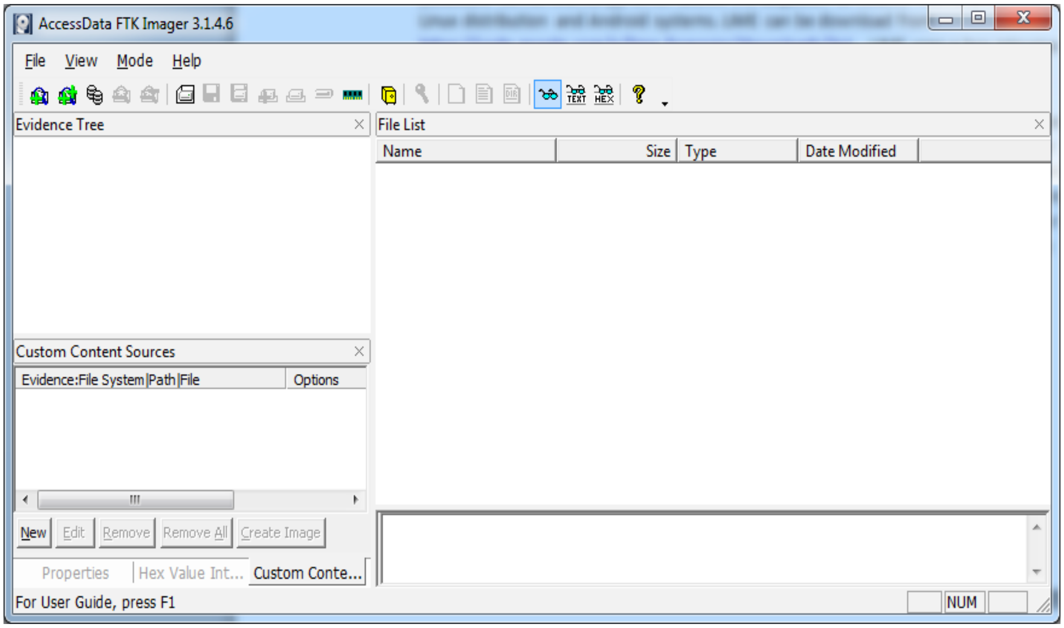
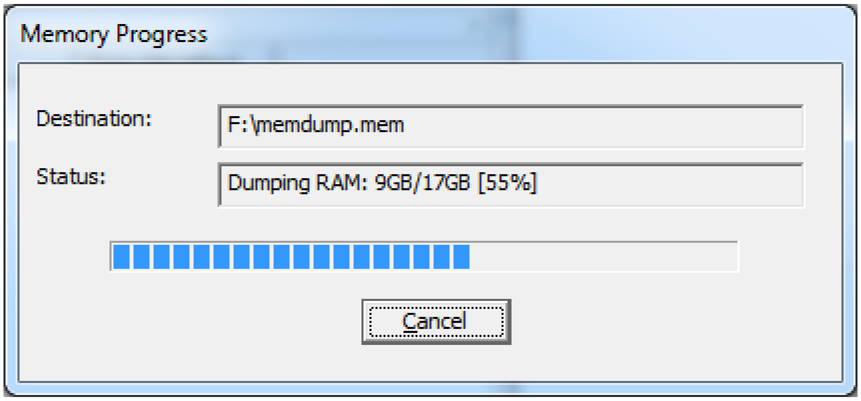


Figure 7

Click on **File → Capture Memory** menu and choose an external drive to save the memory dump:



Destination: F:\memdump.mem

Status: Dumping RAM: 9GB/17GB [55%]

Figure 8

Nice! The memory dump was done!

Memory acquisition on Linux System

I've been researching many good tools for Linux memory acquisition, but I've chosen to show the LiME (Linux Memory Extraction) that's a LKM (Loadable Kernel Memory), which executes the dump from a volatile memory of any modern Linux distribution and Android systems. LiME is available from https://code.google.com/p/lime-forensics/downloads/list. LiME uses a less intrusive approach making its memory acquisition more accurate.

For using LiME, we have to extract it our forensic workstation and compile it. I suggest that you compile this code on a system using the same version and type of Linux. For example, my system is running Kali Linux 1.0.6 64-bits (current version is 1.0.7 and it can be downloaded from http://www.kali.org/downloads/) and I'm going to acquire memory content from a system running the same version of Kali Linux.This assures that we're able to load the module on system's memory without any problem. Therefore, execute the following commands:

```
root@hacker:~# mkdir LiMe

root@hacker:~# cp lime-forensics-1.1-r17.tar.gz LiMe/
root@hacker:~# cd LiMe/

root@hacker:~/LiMe# ls

doc  lime-forensics-1.1-r17.tar.gz  src

root@hacker:~/LiMe# tar zxvf lime-forensics-1.1-r17.tar.gz

doc/
doc/LiME_Documentation_1.1.pdf
src/
src/disk.c
src/lime.h
src/main.c
src/Makefile
src/Makefile.sample
src/tcp.c

root@hacker:~/LiMe# ls
doc  lime-forensics-1.1-r17.tar.gz  src

root@hacker:~/LiMe# cd src

root@hacker:~/LiMe/src# ls
disk.c  lime-3.7-trunk-amd64.ko  lime.h  main.c  Makefile
Makefile.sample  tcp.c

root@hacker:~/LiMe/src# make clean

make tidy
make[1]: Entering directory `/root/LiMe/src'
rm -f *.o *.mod.c Module.symvers Module.markers modules.order
\.*.o.cmd \.*.ko.cmd \.*.o.d
rm -rf \.tmp_versions
make[1]: Leaving directory `/root/LiMe/src'
rm -f *.ko


root@hacker:~/LiMe/src# ls
disk.c  lime.h  main.c  Makefile  Makefile.sample  tcp.c
```

```
root@hacker:~/LiMe/src# make

make -C /lib/modules/3.7-trunk-amd64/build M=/root/LiMe/src
modules
make[1]: Entering directory `/usr/src/linux-headers-3.7-trunk-
amd64'
  CC [M]  /root/LiMe/src/tcp.o
  CC [M]  /root/LiMe/src/disk.o
  CC [M]  /root/LiMe/src/main.o
/root/LiMe/src/main.c: In function ‘__check_dio’:
/root/LiMe/src/main.c:56:1: warning: return from incompatible
pointer type [enabled by default]
  LD [M]  /root/LiMe/src/lime.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /root/LiMe/src/lime.mod.o
  LD [M]  /root/LiMe/src/lime.ko
make[1]: Leaving directory `/usr/src/linux-headers-3.7-trunk-
amd64'
strip --strip-unneeded lime.ko
mv lime.ko lime-3.7-trunk-amd64.ko
make tidy
make[1]: Entering directory `/root/LiMe/src'
rm -f *.o *.mod.c Module.symvers Module.markers modules.order
\.*.o.cmd \.*.ko.cmd \.*.o.d
rm -rf \.tmp_versions
make[1]: Leaving directory `/root/LiMe/src'

root@hacker:~/LiMe/src#
```

We've done it! For next step, it's appropriate to have an external hard disk to save the memory dump. Then we should copy the LiME module (this case, lime-3.7-trunk-amd64.ko) to a pen drive, insert it in the target system (machine under analysis) and run the following commands:

```
root@hacker:/media/pendrive# insmod /media/pendrive/lime-3.7-
trunk-amd64.ko "path=/media/external_drive/kali_memory_dump.bin
format=lime"

root@hacker:/media/pendrive# cd /media/external_drive

root@hacker:/media/external_drive# ls -lh kali_memory_dump.bin
```

```
-r--r--r-- 1 root root 18G Jun 11 01:55 kali_memory_dump.bin
```

Fantastic! We got a memory dump from Kali Linux. There're other format options for the image such as **raw** and **padded**, but the suggested format for tools such as Volatility is **lime**.

If it's necessary to perform a new memory dump, remove the lime module from memory before repeating the process:

```
root@hacker:/media/pendrive# rmmod lime
```

That's everything fine. However, what could we do with these memory dumps ? Obvious: we'll use the best forensic tool in the world to analyze it and try to find out any potential malwares: Volatility. That's our next article. See you. ☺

Alexandre Borges.