Memory Acquisition

by Alexandre Borges

This article has as goal to explain how to acquire memory information to be analyzed by Volatility

Version 1.1 (FEB/23/2015)

Introduction

The IT security is the most complex area inside the digital world because we are exposed to a huge number of threats and dozens of malwares (virus, Trojans, spies and worms) come up every single day, including other hundreds of variants, and it is obvious that, for a while, crackers are winning the war. Malwares are becoming more sophisticated by adding rootkits techniques in their codes, by using anti-forensic techniques to hinder the analysis by experts, by abusing of encrypted codes and lots of other tricks.

The known static and dynamic analysis still is valid, but they are difficult to execute successfully. For example, if we try to execute a dynamic analysis by using a virtual machine, we find malwares that include in its code instructions such as "**sidt**" (Red Pill), "**str**" (used to load **task state segments – TSS**) and "**sldt**" (No Pill) to detect and stop their execution while inside in a virtualized environment.

Trying to attach a debugger can be hard because malwares are using calls such as "**QueryPeformanceCounter**", "**GetTickCoun**t" and SEH manipulation as anti-forensic methods to short their executions. Even usual tricks such as deploying calls such as "**FindWindowsA**" during a **TLS callback function** (called from a **TLS section**) to prevent a specific debugger (WinDbg, for example) to continue are commonly used. Of course, trying to analyze malware on the disk can be as difficult as trying to execute it (in a dynamic analysis) because usually they are encrypted and are using strange packers (not more the old and good UPX).

Therefore, there is only a place where we can fight against malwares with a reasonable chance to overcome it that is on the memory and here arises the best memory forensic of the world: **Volatility**. In my sincere opinio, the main fact that justify the choice to look for threats on memory is that most time the information resident on memory is not the same from disk. Furthermore, the information from memory is more complete.

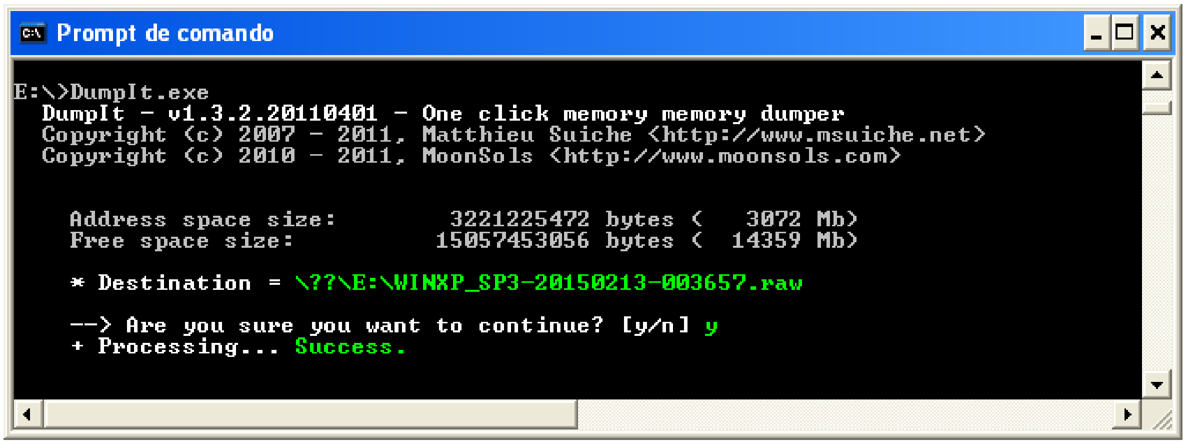
This article we will learn how to acquire the memory by using simple and efficient techniques on Windows and Linux. On next articles, we will study the Volatility framework.

First technique: DumpIt (for Windows)

Eventually, the DumpIt tool (http://www.moonsols.com/wp-content/uploads/downloads/2011/07/DumpIt.zip) is the easiest tool to dump information from memory. Nevertheless, I had some problems to use this version of DumpIt tool (1.3.2) when working with machines with memory above of 4 GB RAM because either the dump fails or the dump is useless for using with Volatility. Therefore, it is recommended to use DumpIt for system with up to 4 GB RAM. Additionally, the Moonsols make available the version 1.4 for free download, but it doesn't support either dump from 64-bits systems or Windows 8, so it is not so useful for a regular basis analysis. The Enterprise 2.0 version brings all features, but it is a commercial and expensive product (http://www.moonsols.com/#pricing).

For showing the next step, we are going to dump the memory from a Windows XP SP3 x86 system with 4 GB RAM. Thus, save the DumpIt in a USB pen drive and, from there, execute it:

```
E:\>DumpIt.exe
```



```
E:\>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:         3221225472 bytes (   3072 Mb)
    Free space size:           15057453056 bytes (  14359 Mb)

    * Destination = \??\E:\WINXP_SP3-20150213-003657.raw

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

DumpIT output **Figure 1**

```
E:\> dir *.raw

12/02/2015  21:45     3.221.225.472 WINXP_SP3-20150213-003657.raw
                 1 arquivo(s)   3.221.225.472 bytes
                 0 pasta(s) 11.836.227.584 bytes disponíveis
```

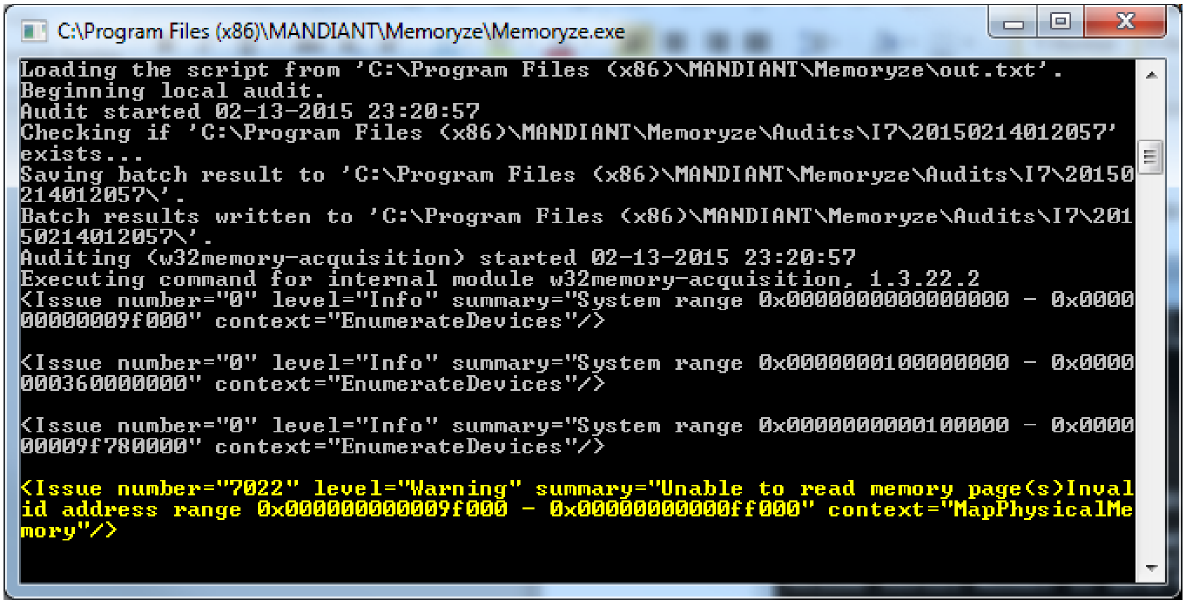
Second technique: Memorize (for Windows)

Memorize (https://dl.mandiant.com/EE/library/MemoryzeSetup3.0.msi) is a free tool distributed by FireEye (Mandiant) that makes the dump of whole memory without facing the same limitations found on systems with more than 4 GB RAM.

The following example shows how to use Memorize on a Windows 7 x64 system with 12 GB RAM. Therefore, execute the next commands:

```
C:\>cd "Program Files (x86)"

C:\Program Files (x86)>cd MANDIANT

C:\Program Files (x86)\MANDIANT>cd Memoryze

C:\Program Files (x86)\MANDIANT\Memoryze>dir


07/10/2013  06:55 PM             1,598 AcquireDriver.Batch.xml
07/10/2013  06:55 PM             1,425 AcquireMemory.Batch.xml
07/10/2013  06:55 PM             2,043 AcquireProcessMemory.Batch.xml
02/13/2015  11:20 PM    <DIR>          Audits
07/10/2013  06:55 PM             1,844 DriverAuditModuleList.Batch.xml
07/10/2013  06:55 PM             3,437 DriverAuditSignature.Batch.xml
07/10/2013  06:55 PM             2,951 DriverDD.bat
07/10/2013  06:55 PM             5,993 DriverSearch.bat
07/10/2013  06:55 PM             2,631 DriverWalkList.bat
07/10/2013  06:55 PM             2,544 HookAudit.Batch.xml
07/10/2013  06:55 PM             4,577 HookDetection.bat
07/10/2013  06:55 PM             2,995 MemoryDD.bat
07/10/2013  08:47 PM        11,894,576 Memoryze.exe

(truncated output)

              17 File(s)     12,492,190 bytes
               3 Dir(s)  281,286,672,384 bytes free

C:\Program Files (x86)\MANDIANT\Memoryze> MemoryDD.bat
```

When we execute (as Administrator) the **MemoryDD.bat** it opens a new window to trace the process as shown below:



```
Loading the script from 'C:\Program Files (x86)\MANDIANT\Memoryze\out.txt'.
Beginning local audit.
Audit started 02-13-2015 23:20:57
Checking if 'C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7\20150214012057'
exists...
Saving batch result to 'C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7\20150
214012057\'.
Batch results written to 'C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7\201
50214012057\'.
Auditing (w32memory-acquisition) started 02-13-2015 23:20:57
Executing command for internal module w32memory-acquisition, 1.3.22.2
<Issue number="0" level="Info" summary="System range 0x0000000000000000 - 0x0000
00000009f000" context="EnumerateDevices"/>

<Issue number="0" level="Info" summary="System range 0x0000000100000000 - 0x0000
000360000000" context="EnumerateDevices"/>

<Issue number="0" level="Info" summary="System range 0x0000000000100000 - 0x0000
00009f780000" context="EnumerateDevices"/>

<Issue number="7022" level="Warning" summary="Unable to read memory page(s)Inval
id address range 0x000000000009f000 - 0x00000000000ff000" context="MapPhysicalMe
mory"/>
```

Memoryze Dump **Figure 2**

After performing the dump, we can see the memory image by executing the following commands:

```
C:\Program Files (x86)\MANDIANT\Memoryze>cd Audits

C:\Program Files (x86)\MANDIANT\Memoryze\Audits>dir
 Volume in drive C has no label.
 Volume Serial Number is EA78-9906

 Directory of C:\Program Files (x86)\MANDIANT\Memoryze\Audits

02/13/2015  11:20 PM    <DIR>          .
02/13/2015  11:20 PM    <DIR>          ..
02/13/2015  11:20 PM    <DIR>          I7
               0 File(s)              0 bytes
               3 Dir(s)  281,285,427,200 bytes free

C:\Program Files (x86)\MANDIANT\Memoryze\Audits>cd I7

C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7>dir
 Volume in drive C has no label.
 Volume Serial Number is EA78-9906

 Directory of C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7

02/13/2015  11:20 PM    <DIR>          .
02/13/2015  11:20 PM    <DIR>          ..
02/13/2015  11:20 PM    <DIR>          20150214012057
               0 File(s)              0 bytes
               3 Dir(s)  281,285,427,200 bytes free

C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7>cd 20150214012057

C:\Program Files (x86)\MANDIANT\Memoryze\Audits\I7\20150214012057>dir
 Volume in drive C has no label.
 Volume Serial Number is EA78-9906

 Directory of C:\Program Files
(x86)\MANDIANT\Memoryze\Audits\I7\20150214012057

02/13/2015  11:20 PM    <DIR>          .
02/13/2015  11:20 PM    <DIR>          ..
02/13/2015  11:20 PM            20,080 BatchResults.xml
02/13/2015  11:20 PM               283 Issues.BatchResults.xml
02/13/2015  11:31 PM             1,299 issues.memory.06181037.img.xml
02/13/2015  11:31 PM    14,495,514,624 memory.06181037.img
               4 File(s) 14,495,536,286 bytes
               2 Dir(s)  281,285,427,200 bytes free
```

That's perfect! My system has 12 GB RAM then the size of the memory dump file is compatible.

Third technique: F-Response + FTK Imager

This third method is the most recommended for real cases because it uses the F-Response (https://www.f-response.com/) , which provides a secure infrastructure to remotely acquire the memory content by using the FTK through the iSCSI protocol without running the risk to contaminate the target machine.

In a simple form, we have two system where the first one is our target machine from where we are going to acquire the memory and the second one is our workstation system where we will install the FTK Imager (http://accessdata.com/product-download/digital-forensics/ftk-imager-version-3.2.0) for acquiring the memory from the first system through iSCSI service and, on a next article, it will be used to analyze the image by deploying the Volatility.

What program does provide the iSCSI service? The F-Response software does. There are some versions of F-Response software (the matrix includes the comparison is found here https://www.f-response.com/assets/pdfs/ProductMatrix-December2014.pdf) and, in this article, we will use the Tactical version. We should remember that F-Response is a commercial product, but it worth each invested dollar.

F-Reponse software is implemented in dongles (similar to pen drives) such as the shown below:

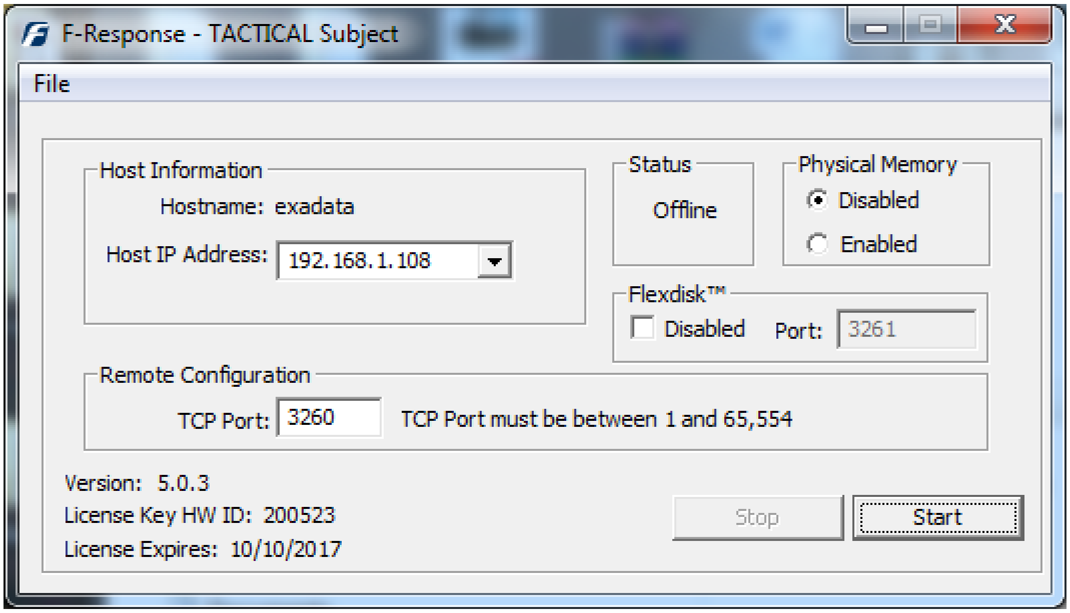
F-Response Tactical **Figure 3**

The blue dongle (Tactical Examiner) is plugged in the examiner workstation (where we will install the FTK and Volatility) and the silver dongle (Tactical Subject) is installed in the target system from where we will acquire the memory.

For performing our demonstration we are going to use a system running Windows 7 x64 with 12 GB RAM as the examiner workstation (here it will be installed the FTK and, in the future, the Volatility) and another system as the target system (from where the memory will be acquired) running a Windows 7 x64 with 16 GB RAM.

Finally, to acquire the memory from the target system, execute the following steps:

1. Install the FTK imager on the examiner system.
2. Insert the F-Response (silver – Tactical Subject) into to target system, go the logical letter (F:\ in this case) and execute the **f-response-tacsub.exe** program. The first screen will be shown as below:



Host Information
Hostname: exadata
Host IP Address: 192.168.1.108

Status
Offline

Physical Memory
● Disabled
○ Enabled

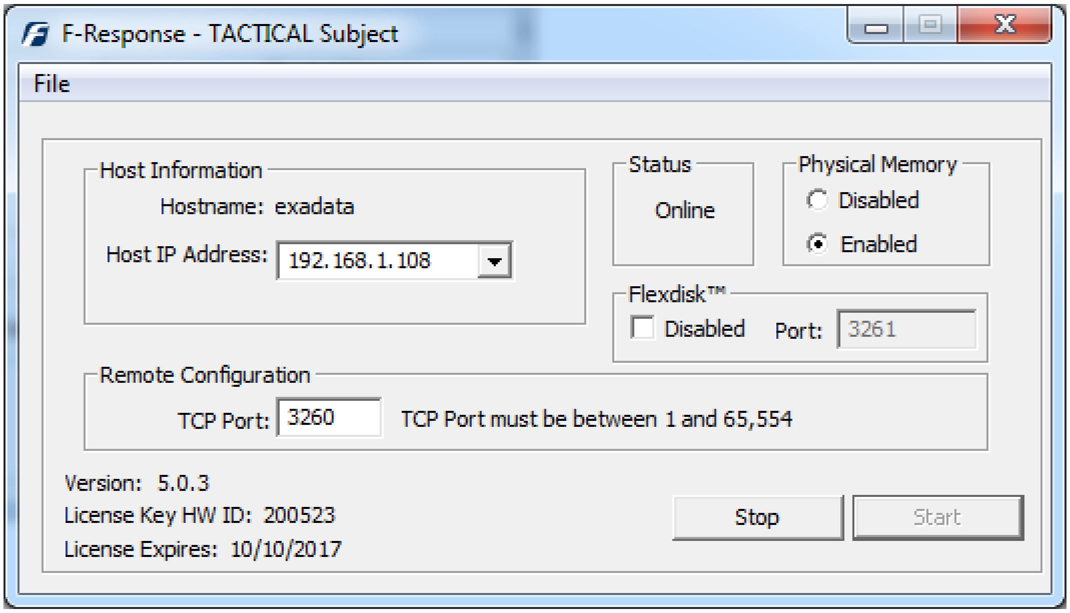
Flexdisk™
☐ Disabled Port: 3261

Remote Configuration
TCP Port: 3260 TCP Port must be between 1 and 65,554

Version: 5.0.3
License Key HW ID: 200523
License Expires: 10/10/2017

F-Response Tactical Subject – first screen **Figure 4**

3. Change the **Physical Memory option to "Enabled"** and click on **Start button** as shown below:



Host Information
Hostname: exadata
Host IP Address: 192.168.1.108

Status
Online

Physical Memory
○ Disabled
● Enabled

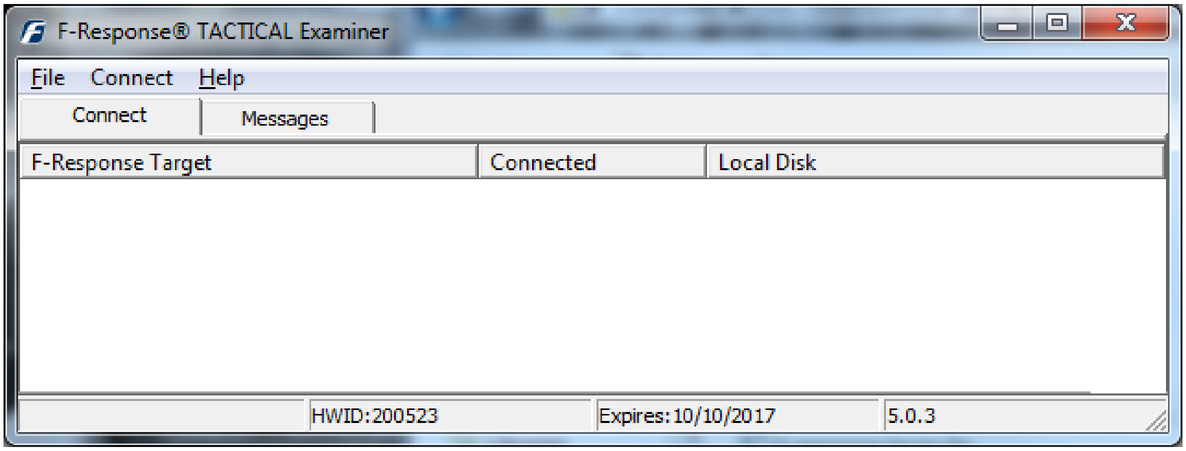
Flexdisk™
☐ Disabled Port: 3261

Remote Configuration
TCP Port: 3260 TCP Port must be between 1 and 65,554

Version: 5.0.3
License Key HW ID: 200523
License Expires: 10/10/2017

F-Response Tactical Subject – second screen **Figure 5**

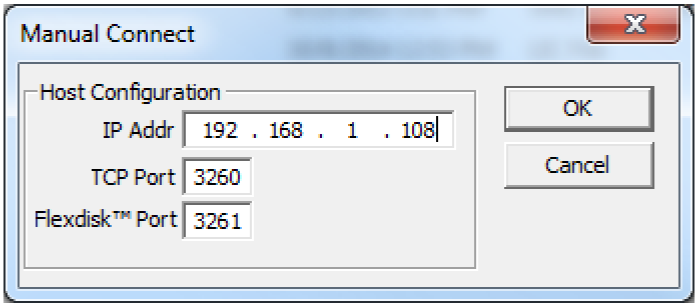
4. Insert the F-Response blue dongle (Tactical Examiner) in the the examiner system, go to "**TACTICAL Examiner**" directory and execute the "**f-response-tacex.exe**") application as shown below:



HWID:200523 | Expires:10/10/2017 | 5.0.3

F-Response Tactical Examiner – Welcome Screen **Figure 6**

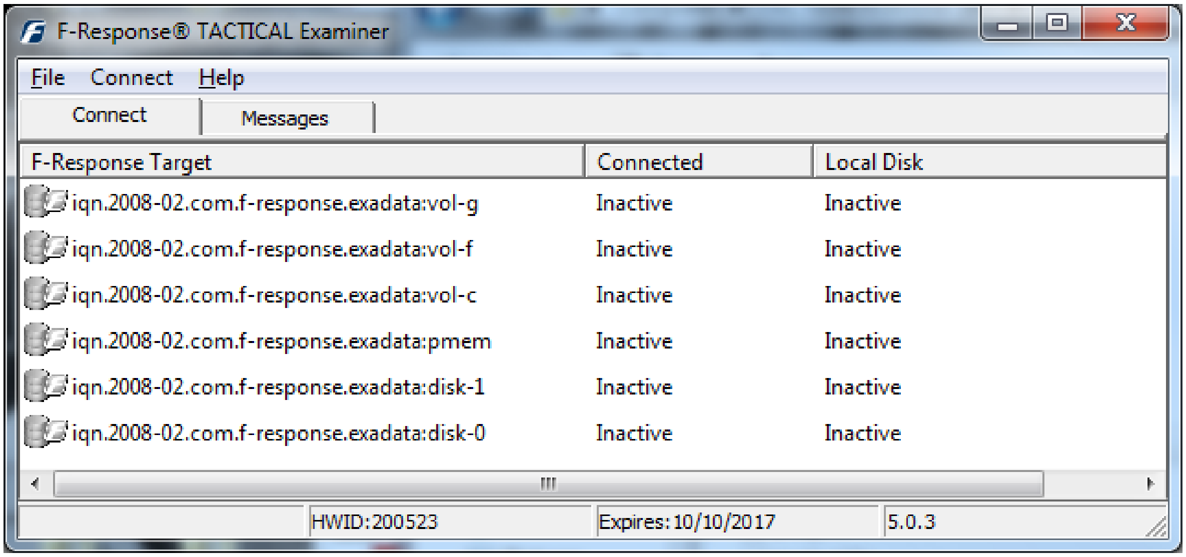
5. Go to **File → Manual Connect** menu (it also could be **Auto Connect option**) and fill the blanks with network address information from the target (subject) system as shown below:



IP Addr 192 . 168 . 1 . 108
TCP Port 3260
Flexdisk™ Port 3261

F-Response Tactical Examiner – Connection screen **Figure 7**

6. Click on **OK button** and we should see the following screen:

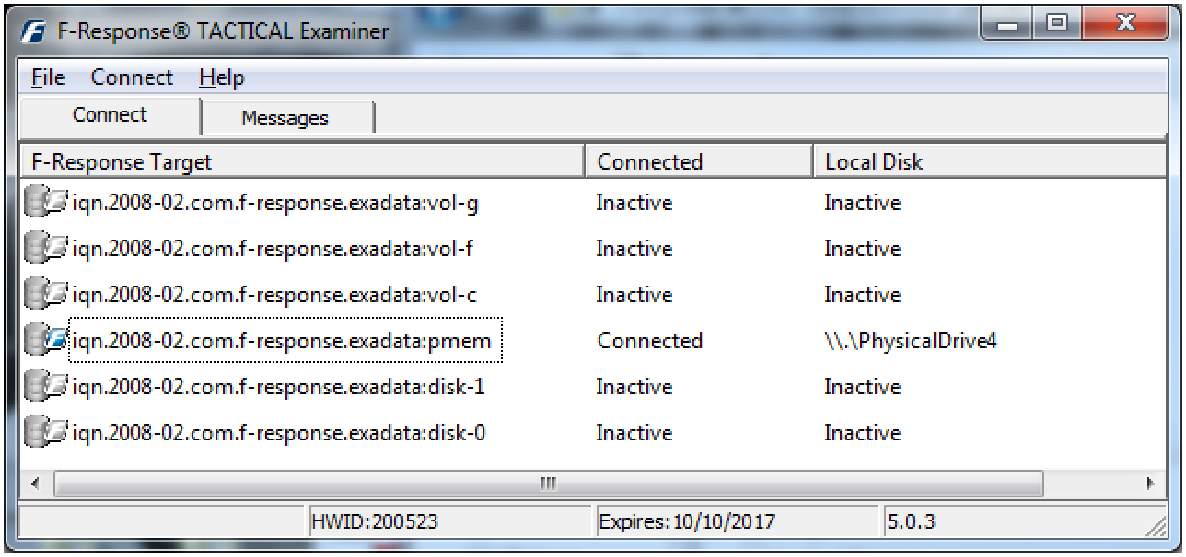


F-Response Target	Connected	Local Disk
iqn.2008-02.com.f-response.exadata:vol-g	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:vol-f	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:vol-c	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:pmem	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:disk-1	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:disk-0	Inactive	Inactive

HWID:200523 Expires:10/10/2017 5.0.3

F-Response Tactical Examiner – Target Screen **Figure 8**

7. The iSCSI targets shown above are all the disks from system and the memory (ended with "pmem" string). Thus, click on "**iqn.2008-02.com.f-response.exadata:pmem**" target, go to **Connect menu** and choose **Login to F-Response Disk** option:



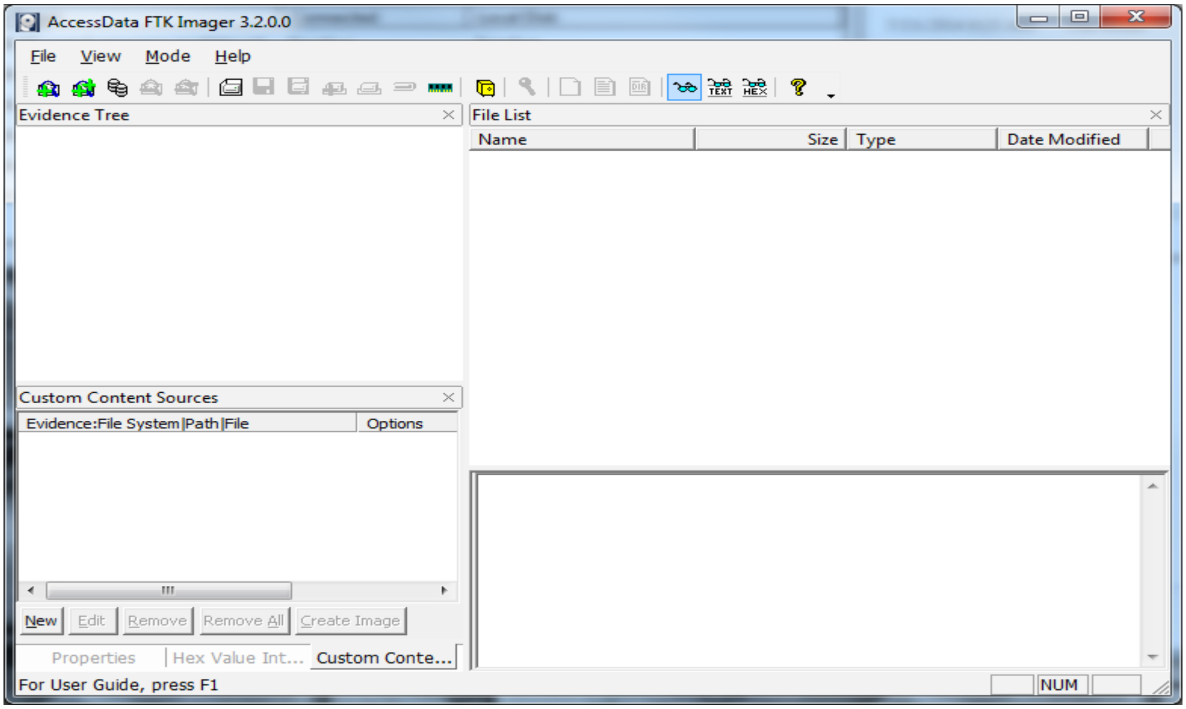
F-Response Target	Connected	Local Disk
iqn.2008-02.com.f-response.exadata:vol-g	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:vol-f	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:vol-c	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:pmem	Connected	\\.\PhysicalDrive4
iqn.2008-02.com.f-response.exadata:disk-1	Inactive	Inactive
iqn.2008-02.com.f-response.exadata:disk-0	Inactive	Inactive

HWID:200523 Expires:10/10/2017 5.0.3

F-Response Tactical Examiner – Memory Connection screen **Figure 8**

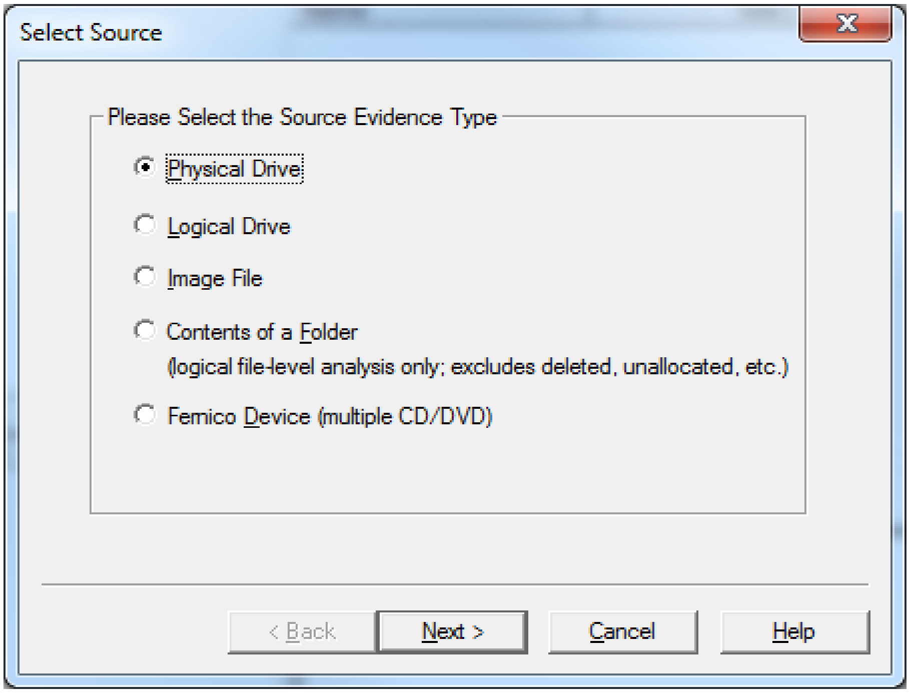
That is done! The F-Response has associated the **\\.\PhysicalDrive 4** device path to subject's memory.

8. On the examiner system, open the FTK Imager:



FTK Imager – Welcome screen **Figure 9**

9. Go to **File → Create Disk Image** menu and the following screen will come up:



Please Select the Source Evidence Type

● Physical Drive

○ Logical Drive

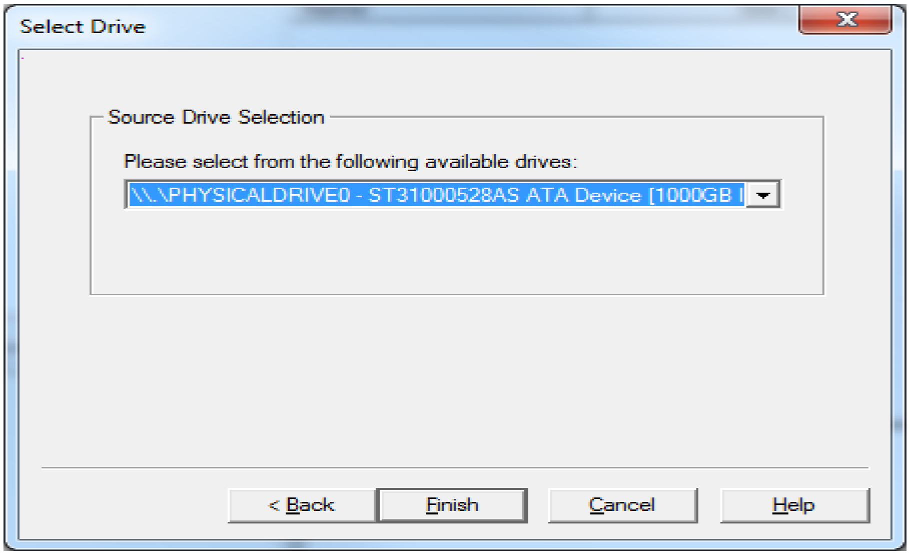
○ Image File

○ Contents of a Folder
(logical file-level analysis only; excludes deleted, unallocated, etc.)

○ Femico Device (multiple CD/DVD)

FTK Imager – Source Type screen **Figure 10**

10. Keep marked the Physical Drive option and click on **Next button**:

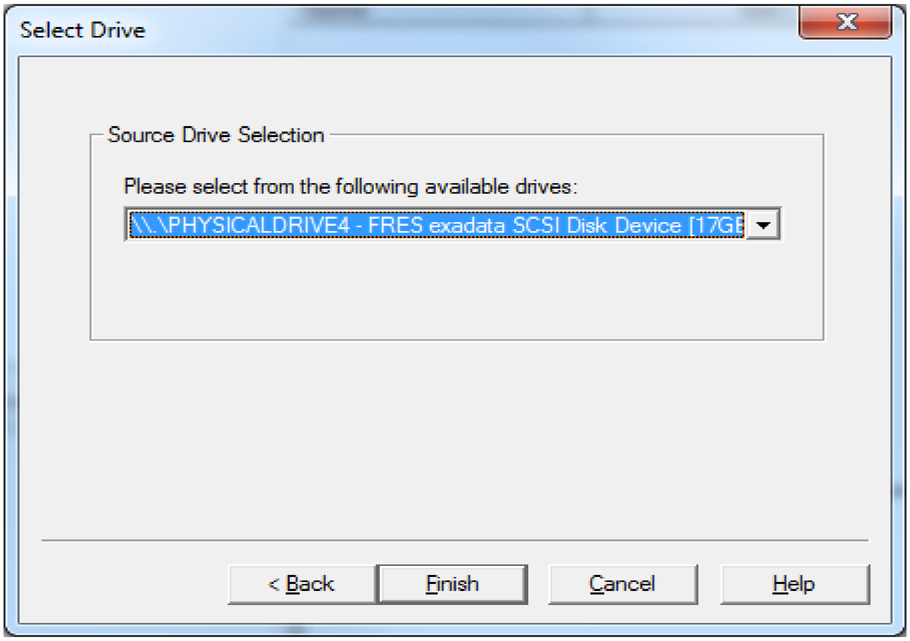


Please select from the following available drives:

\\.\PHYSICALDRIVE0 - ST31000528AS ATA Device [1000GB I

FTK Imager – Source Selection screen **Figure 11**

11. Choose **"\\.\PHYSICALDRIVE4**" , which points to target's memory as shown below:

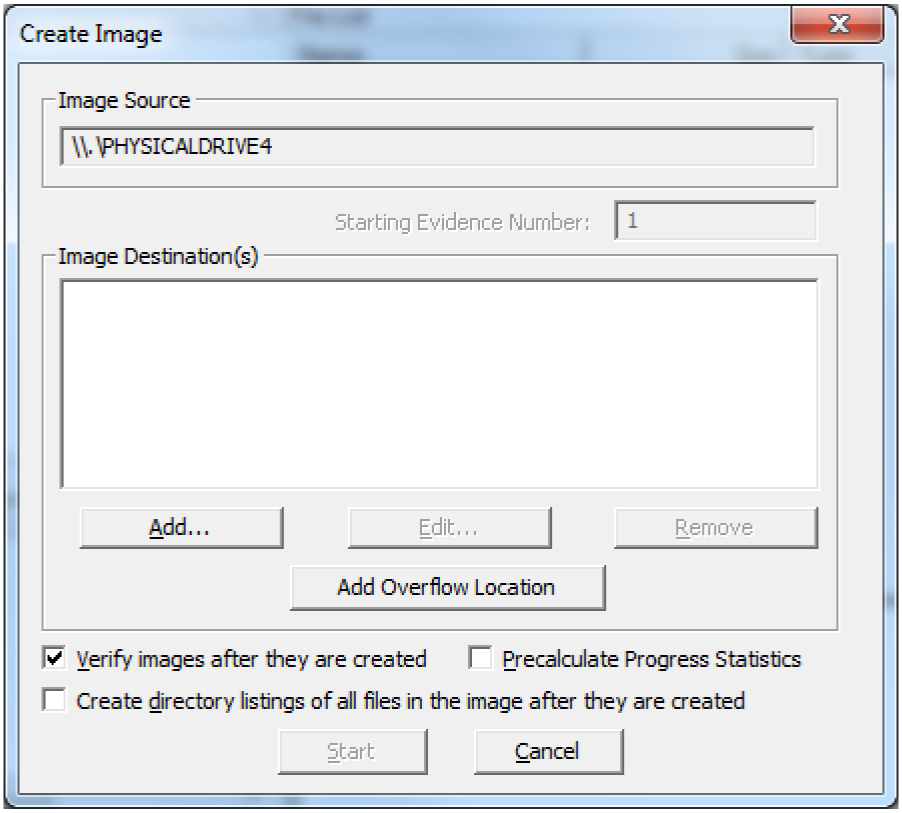


Please select from the following available drives:

\\.\PHYSICALDRIVE4 - FRES exadata SCSI Disk Device [17GE

FTK Imager – Source Selection screen 2 **Figure 12**

12. Click on **Finish button**:



Create Image

Image Source: \\.\PHYSICALDRIVE4

Starting Evidence Number: 1

Image Destination(s)

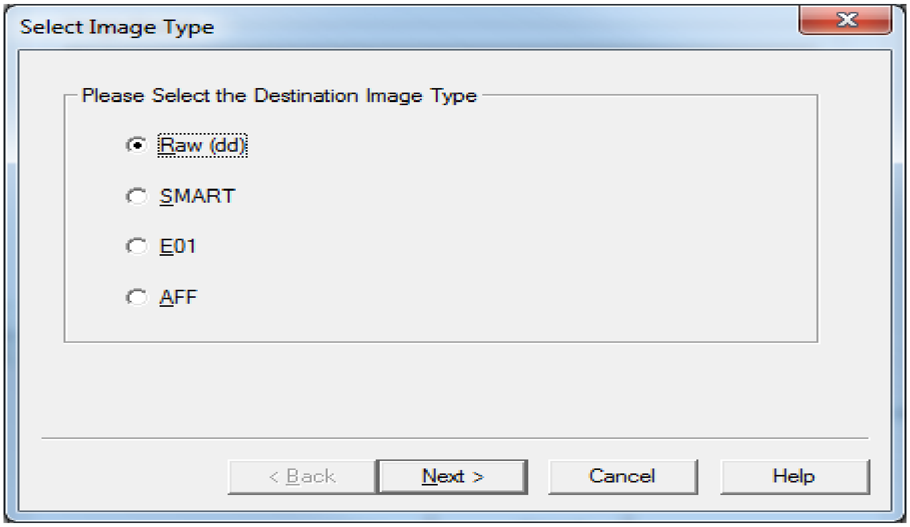
☑ Verify images after they are created

□ Precalculate Progress Statistics

□ Create directory listings of all files in the image after they are created

FTK Imager – Create Image screen **Figure 13**

13. Click on **Add button**:



Select Image Type

Please Select the Destination Image Type

● Raw (dd)

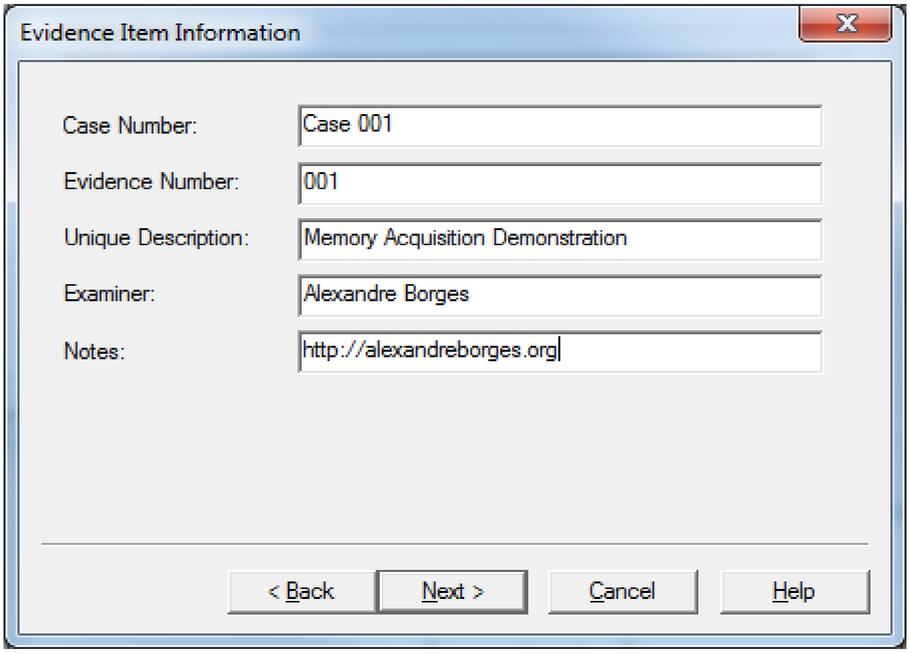
○ SMART

○ E01

○ AFF

FTK Imager – Select Image Type screen **Figure 14**

14. Mark the **Raw (dd)** option, click on **Next button** and fill the fields as shown below:



Case Number: Case 001

Evidence Number: 001

Unique Description: Memory Acquisition Demonstration

Examiner: Alexandre Borges

Notes: http://alexandreborges.org

FTK Imager – Evidence Item Information screen **Figure 15**

15. Click on **Next button** and fill the fields. It is very important to input "**0**" into "**Image Fragment Size (MB)**" textbox as shown in the following screen:

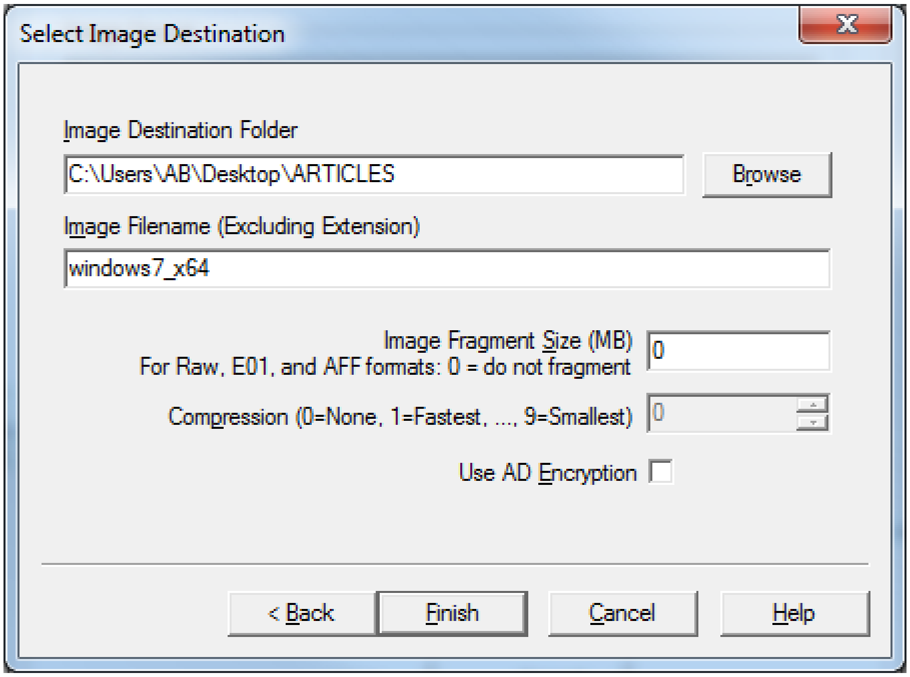


Image Destination Folder: C:\Users\AB\Desktop\ARTICLES

Image Filename (Excluding Extension): windows7_x64

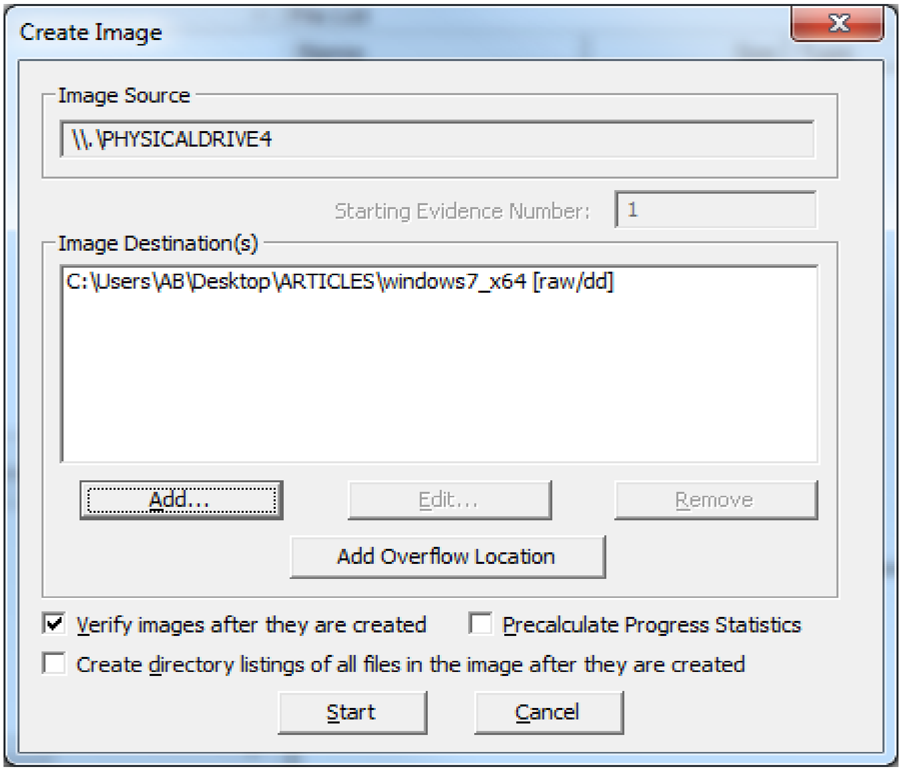
Image Fragment Size (MB) For Raw, E01, and AFF formats: 0 = do not fragment: 0

Compression (0=None, 1=Fastest, ..., 9=Smallest): 0

Use AD Encryption □

FTK Imager – Select Image Destination screen **Figure 16**

16. Click on **Finish button** and the following screen will be shown:



\\.\PHYSICALDRIVE4

1

C:\Users\AB\Desktop\ARTICLES\windows7_x64 [raw/dd]

☑ Verify images after they are created ☐ Precalculate Progress Statistics

☐ Create directory listings of all files in the image after they are created

FTK Imager – Create Image screen - 2 **Figure 17**

17. Click on **Start button**:

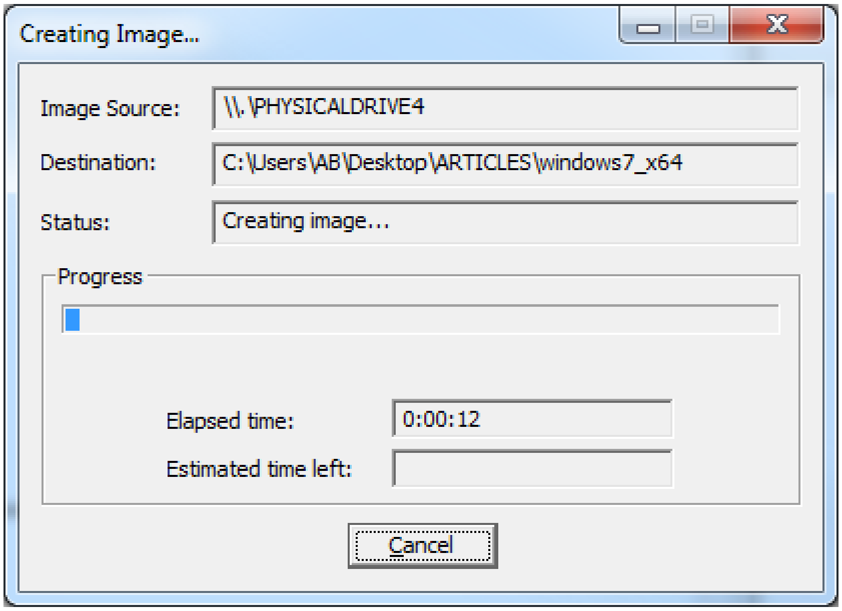


Image Source: \\.\PHYSICALDRIVE4

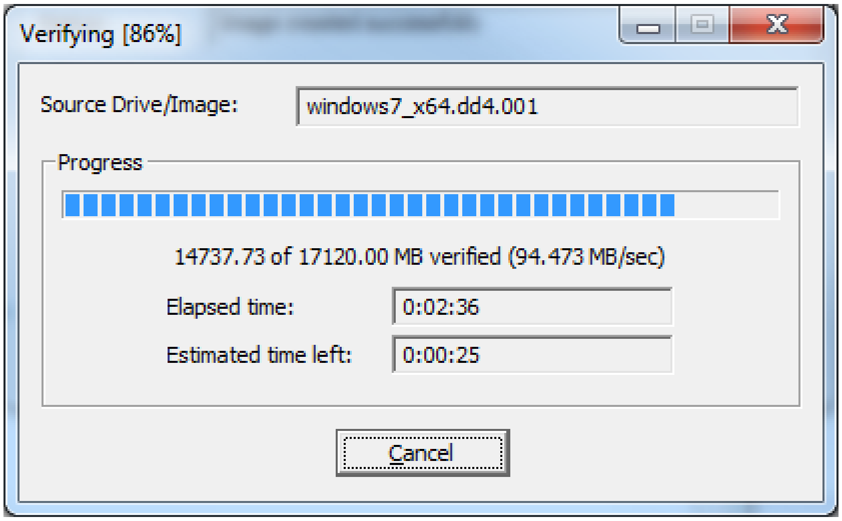
Destination: C:\Users\AB\Desktop\ARTICLES\windows7_x64

Status: Creating image...

Elapsed time: 0:00:12

Estimated time left:

FTK Imager – Creating Image screen **Figure 18**



Source Drive/Image: windows7_x64.dd4.001

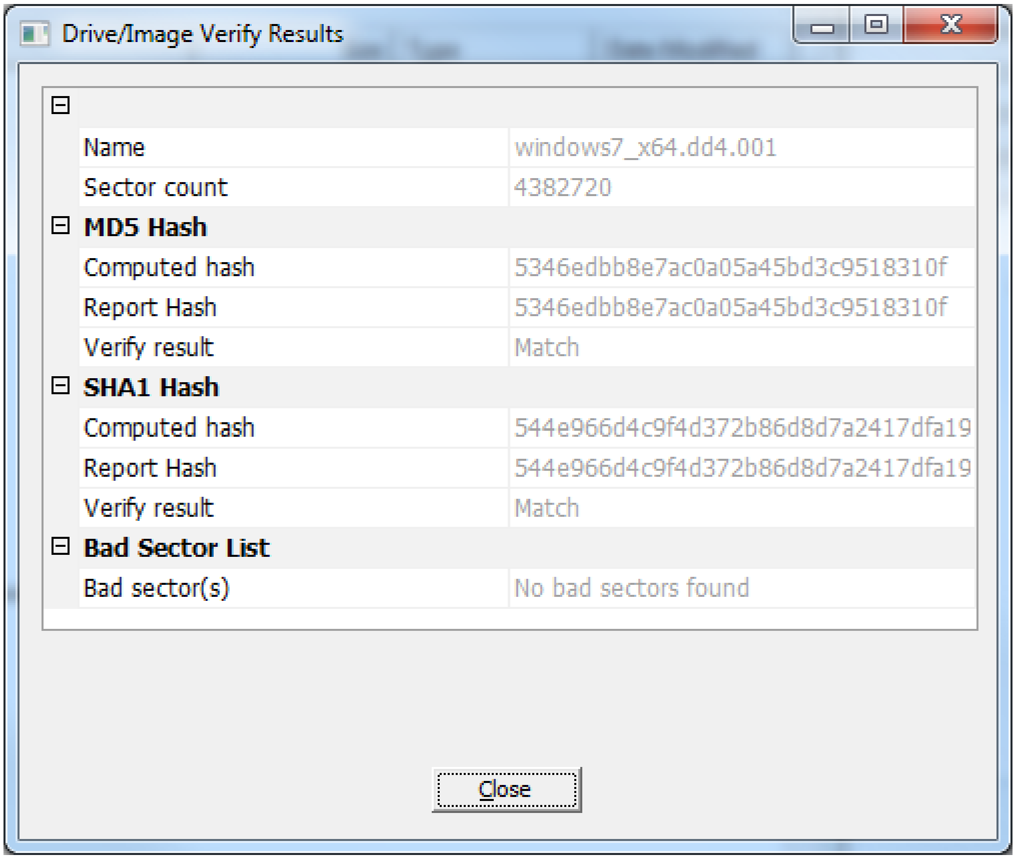
Progress

14737.73 of 17120.00 MB verified (94.473 MB/sec)

Elapsed time: 0:02:36

Estimated time left: 0:00:25

FTK Imager – Verifying screen **Figure 19**



Name	windows7_x64.dd4.001
Sector count	4382720
MD5 Hash	
Computed hash	5346edbb8e7ac0a05a45bd3c9518310f
Report Hash	5346edbb8e7ac0a05a45bd3c9518310f
Verify result	Match
SHA1 Hash	
Computed hash	544e966d4c9f4d372b86d8d7a2417dfa19
Report Hash	544e966d4c9f4d372b86d8d7a2417dfa19
Verify result	Match
Bad Sector List	
Bad sector(s)	No bad sectors found

FTK Imager – Drive/Image Verify Results **Figure 20**

18. Click on **Close button** from **Drive/Image Verify Results** screen and on **Close button** from **Creating Image screen**. Close the FTK Imager, go to **Connect menu** from F-Response Examiner and click on **Logout of F-Response Disk** option. Close the **F-Response Tactical Examiner** on examiner system. On the target system, click on **Stop button** and close the **F-Response Tactical Subject program.** On both systems, eject the F-Response Tactical dongle.

19. On the examiner system, list the memory image acquired from target system:

```
C:\Users\AB\Desktop\ARTICLES>dir

 Volume in drive C has no label.
 Volume Serial Number is EA78-9906

 Directory of C:\Users\AB\Desktop\ARTICLES

02/16/2015  08:40 PM    <DIR>          .
02/16/2015  08:40 PM    <DIR>          ..
02/16/2015  06:44 PM        35,910,816 AccessData FTK Imager3-2-
0.exe
02/16/2015  08:40 PM    17,951,621,120 windows7_x64.dd4.001
02/16/2015  08:43 PM             1,390 windows7_x64.dd4.001.txt
               3 File(s) 17,987,533,326 bytes
               2 Dir(s)  240,280,698,880 bytes free
```

20. Download the Volatility standalone version for Windows (http://downloads.volatilityfoundation.org/releases/2.4/volatility_2.4.win.standalone.zip). By running Volatility, it is possible to confirm (or making a good guess, as least) that the memory image has come from a Windows 7 x64 version such as demonstrated below:

```
c:\Volatility24>volatility_24.exe -f
c:\Users\AB\Desktop\ARTICLES\windows7_x64.dd4.001 imageinfo

Volatility Foundation Volatility Framework 2.4
Determining profile based on KDBG search...

          Suggested Profile(s) : Win7SP0x64, Win7SP1x64,
Win2008R2SP0x64, Win2008R2SP1x64
                     AS Layer1 : AMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace
(C:\Users\AB\Desktop\ARTICLES\windows7_x64.dd4.001)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf80003c570a0L
          Number of Processors : 8
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff80003c58d00L
                KPCR for CPU 1 : 0xfffff8800330f000L
                KPCR for CPU 2 : 0xfffff88003381000L
                KPCR for CPU 3 : 0xfffff880033f3000L
                KPCR for CPU 4 : 0xfffff880009cf000L
                KPCR for CPU 5 : 0xfffff88002090000L
                KPCR for CPU 6 : 0xfffff88002102000L
                KPCR for CPU 7 : 0xfffff88002174000L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2015-02-16 22:01:35 UTC+0000
     Image local date and time : 2015-02-16 20:01:35 -0200
```

Forth Technique: LiME (for Linux)

Certainly, on a Linux system, one of the best tool to acquire memory in a forensic way is the LiME (Linux Memory Extractor - https://github.com/504ensicsLabs/LiME/archive/master.zip) that works as a LKM (Loadable Kernel Module), it is little intrusive and works on any modern Linux distribution and Android systems.

We are going to test LiME on an OEL7 (Oracle Enterprise Linux 7 x64 - https://edelivery.oracle.com/linux) with 4 GB RAM. Thus, to use LiME, execute the following commands:

```
[alexandreborges@oel7 /]$ cd

[alexandreborges@oel7 ~]$ cd Downloads/

[alexandreborges@oel7 Downloads]$ ls
LiME-master.zip

[alexandreborges@oel7 Downloads]$ unzip LiME-master.zip
Archive:  LiME-master.zip
ddb240bfefe671354660513490aa15e7a522b26a
   creating: LiME-master/
  inflating: LiME-master/LICENSE
  inflating: LiME-master/README.md
   creating: LiME-master/doc/
  inflating: LiME-master/doc/README.md
   creating: LiME-master/src/
  inflating: LiME-master/src/Makefile
  inflating: LiME-master/src/Makefile.sample
  inflating: LiME-master/src/disk.c
  inflating: LiME-master/src/lime.h
  inflating: LiME-master/src/main.c
  inflating: LiME-master/src/tcp.c

[alexandreborges@oel7 Downloads]$ cd LiME-master/

[alexandreborges@oel7 LiME-master]$ ls
doc  LICENSE  README.md  src

[alexandreborges@oel7 LiME-master]$ cd src

[alexandreborges@oel7 src]$ ls
disk.c  lime.h  main.c  Makefile  Makefile.sample  tcp.c

[alexandreborges@oel7 src]$ make

make -C /lib/modules/3.8.13-55.1.5.el7uek.x86_64/build
M=/home/alexandreborges/Downloads/LiME-master/src modules
make[1]: Entering directory `/usr/src/kernels/3.8.13-
55.1.5.el7uek.x86_64'
  CC [M]  /home/alexandreborges/Downloads/LiME-master/src/tcp.o
  CC [M]  /home/alexandreborges/Downloads/LiME-master/src/disk.o
  CC [M]  /home/alexandreborges/Downloads/LiME-master/src/main.o
  SDTSTB  /home/alexandreborges/Downloads/LiME-master/src/lime.sdtstub.S
  AS [M]  /home/alexandreborges/Downloads/LiME-master/src/lime.sdtstub.o
  LD [M]  /home/alexandreborges/Downloads/LiME-master/src/lime.o
  Building modules, stage 2.
  MODPOST 1 modules
  SDTINF  /home/alexandreborges/Downloads/LiME-master/src/lime.sdtinfo.c
  CC      /home/alexandreborges/Downloads/LiME-master/src/lime.mod.o
  CTF
  LD [M]  /home/alexandreborges/Downloads/LiME-master/src/lime.ko
make[1]: Leaving directory `/usr/src/kernels/3.8.13-55.1.5.el7uek.x86_64'
strip --strip-unneeded lime.ko
mv lime.ko lime-3.8.13-55.1.5.el7uek.x86_64.ko

[alexandreborges@oel7 src]$ ls
disk.c  lime-3.8.13-55.1.5.el7uek.x86_64.ko  lime.mod.c  lime.o
lime.sdtstub.o  main.c  Makefile         modules.order   tcp.c
disk.o  lime.h                               lime.mod.o  lime.sdtinfo.c
lime.sdtstub.S  main.o  Makefile.sample  Module.symvers  tcp.o

[alexandreborges@oel7 src]$
```

The LiME tool was configured! Now, we should insert a pen drive on the target system (OEL7) to save the dump that will be acquired by LiME.

```
[alexandreborges@oel7 src]$ cp lime-3.8.13-55.1.5.el7uek.x86_64.ko
/run/media/alexandreborges/BC53-4A3F

[alexandreborges@oel7 src]$ cd

[alexandreborges@oel7 ~]$ su - root
Password:
Last login: Tue Feb 17 06:02:01 BRST 2015 on pts/0

[root@oel7 ~]# insmod /run/media/alexandreborges/BC53-4A3F/lime-3.8.13-
55.1.5.el7uek.x86_64.ko "path=/run/media/alexandreborges/BC53-
4A3F/oel7_memory.bin format=lime"

[root@oel7 ~]# ls -lh /run/media/alexandreborges/BC53-4A3F/

total 3.6G
-rw-r--r--. 1 alexandreborges alexandreborges  63K Feb 17 06:06 lime-
3.8.13-55.1.5.el7uek.x86_64.ko
-rw-r--r--. 1 alexandreborges alexandreborges 3.6G Feb 17 06:25
oel7_memory.bin
drwx------. 2 alexandreborges alexandreborges 4.0K Feb 17 04:26 System
Volume Information
```

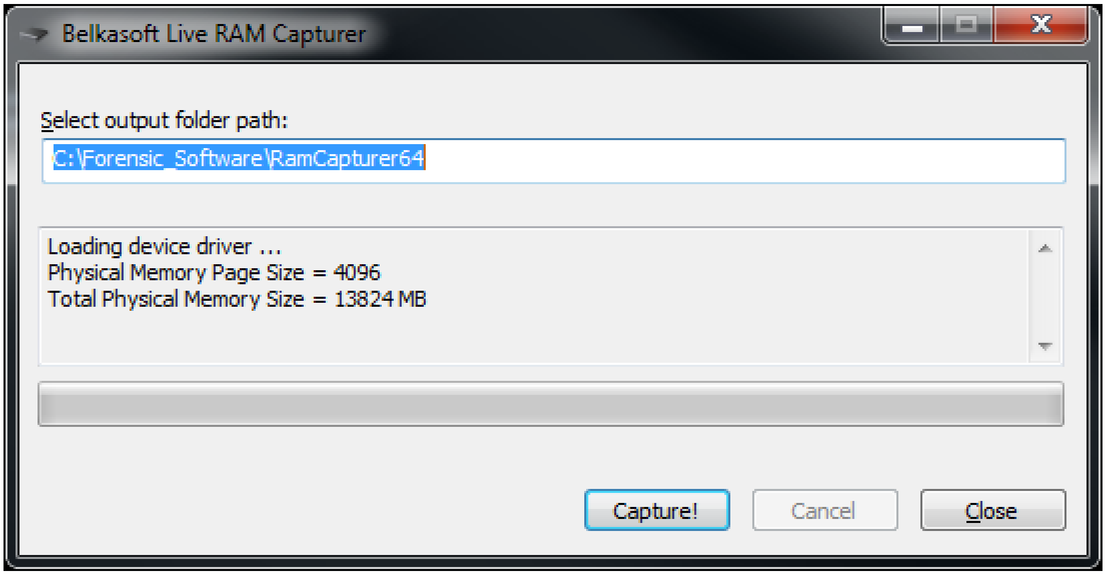
Fifth Technique: Belkasoft Live RAM Capturer

The Live RAM Capturer is a small and very powerful tool to acquire memory on systems such as Windows XP, Windows 7, Windows 8, Windows 2003, Windows 2008, and so on.

Both versions (32 bits and 64 bits) are available to free download on http://belkasoft.com/en/ram/download.asp. An excellent feature of Live RAM Capturer is that it is able to manage acquiring memory from systems with anti-debugging and anti-memory dumping enabled. At same time, images dumped by Belkasoft Live RAM Capturer are full forensics compatible and they can be analyzed by Volatility.

The Live RAM Capturer 64-bits is composed by two files (RamCapure64.exe and RamCaptureDriver64.sys) and we have to execute the former to acquire the memory as show below:

```
c:\Forensic_Software\RamCapturer64>dir
 Volume in drive C has no label.
 Volume Serial Number is EA78-9906

 Directory of c:\Forensic_Software\RamCapturer64

02/23/2015  06:11 PM    <DIR>          .
02/23/2015  06:11 PM    <DIR>          ..
07/29/2013  04:29 AM           148,192 RamCapture64.exe
07/29/2013  04:29 AM            13,344 RamCaptureDriver64.sys
               2 File(s)        161,536 bytes
               2 Dir(s)  280,240,762,880 bytes free

c:\Forensic_Software\RamCapturer64> RamCapture64.exe
```



C:\Forensic_Software\RamCapturer64

```
Loading device driver ...
Physical Memory Page Size = 4096
Total Physical Memory Size = 13824 MB
```

Belkasoft Live RAM Capturer **Figure 21**

Click on **"Capture!" button** to dump the memory. After a few minutes, the memory is dumped:

```
c:\Forensic_Software\RamCapturer64>dir

Directory of c:\Forensic_Software\RamCapturer64

02/23/2015  06:30 PM    <DIR>          .
02/23/2015  06:30 PM    <DIR>          ..
02/23/2015  06:34 PM    14,495,514,624 20150223.mem
07/29/2013  04:29 AM           148,192 RamCapture64.exe
07/29/2013  04:29 AM            13,344 RamCaptureDriver64.sys
               3 File(s) 14,495,676,160 bytes
               2 Dir(s)  265,743,437,824 bytes free
```

Likewise as we have done with the FTK Imager tool, it’s straight to check the operating system information of the image by running the following Volatility command:

```
c:\Volatility24>volatility_24.exe -f
c:\Forensic_Software\RamCapturer64\20150223.mem imageinfo

Volatility Foundation Volatility Framework 2.4
Determining profile based on KDBG search...

          Suggested Profile(s) : Win2008R2SP0x64, Win7SP1x64, Win7SP0x64,
Win2008R2SP1x64
                     AS Layer1 : AMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace
(C:\Forensic_Software\RamCapturer64\20150223.mem)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf800030340a0L
          Number of Processors : 8
     Image Type (Service Pack) : 0
                KPCR for CPU 0 : 0xfffff80003035d00L
                KPCR for CPU 1 : 0xfffff880009eb000L
                KPCR for CPU 2 : 0xfffff88002f64000L
                KPCR for CPU 3 : 0xfffff88002fd5000L
                KPCR for CPU 4 : 0xfffff880009b2000L
                KPCR for CPU 5 : 0xfffff88003088000L
                KPCR for CPU 6 : 0xfffff880030f9000L
                KPCR for CPU 7 : 0xfffff8800316a000L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2015-02-23 21:30:52 UTC+0000
     Image local date and time : 2015-02-23 18:30:52 -0300
```

Even better, we could give a step forward and, by using the simplest way to check running processes, run:

```
c:\Volatility24>volatility_24.exe --profile=Win7SP0x64 -f
c:\Forensic_Software\RamCapturer64\20150223.mem pslist
```

```
Volatility Foundation Volatility Framework 2.4
Offset(V)          Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start
Exit
------------------ -------------------- ------ ------ ------ -------- ------ ------ -------
----------------------- ------------------------------
0xfffffa800a3469e0 System                    4      0    155     2278 ------      0 2015-
02-23 16:56:33 UTC+0000
0xfffffa800c1b95f0 smss.exe                400      4      2       37 ------      0 2015-
02-23 16:56:33 UTC+0000
0xfffffa800be68b30 csrss.exe               508    500      9     1008      0      0 2015-
02-23 16:56:38 UTC+0000
0xfffffa800d118b30 wininit.exe             584    500      3      100      0      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800d141060 csrss.exe               612    596     13     1040      1      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800d1975e0 services.exe            652    584      6      336      0      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800d199b30 lsass.exe               668    584      9      978      0      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800d19d960 lsm.exe                 676    584     11      198      0      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800cd3e730 svchost.exe             788    652     13      420      0      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800cd8f6d0 nvvsvc.exe              868    652      4      134      0      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800cd95340 winlogon.exe            904    596      3      123      1      0 2015-
02-23 16:56:39 UTC+0000
0xfffffa800cd41b30 nvSCPAPISvr.ex          916    652      6      111      0      1 2015-
02-23 16:56:39 UTC+0000
```

```
0xfffffa800cf54b30 GbpSv.exe               964    652     10      251      0      1 2015-
02-23 16:56:39 UTC+0000
0xfffffa800c921b30 svchost.exe             148    652     10      475      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800c930b30 svchost.exe             516    652     24      671      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d75f060 svchost.exe             804    652     34      815      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d536810 svchost.exe            1036    652     36     1275      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d7b8b30 audiodg.exe            1124    516      0 --------      0      0 2015-
02-23 16:56:47 UTC+0000   2015-02-23 17:02:13 UTC+0000
0xfffffa800d7e8b30 svchost.exe            1196    652     19      673      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d80f730 nvxdsync.exe           1324    868     10      240      1      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d4d6060 nvvsvc.exe             1332    868      5      165      1      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d89e730 svchost.exe            1516    652     15      389      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d82f920 svchost.exe            1544    652     12      170      0      0 2015-
02-23 16:56:47 UTC+0000
0xfffffa800d8c2b30 vsmon.exe              1620    652     31      655      0      1 2015-
02-23 16:56:47 UTC+0000
0xfffffa800dac0b30 AvastSvc.exe           1860    652     96     3101      0      1 2015-
02-23 16:56:50 UTC+0000
0xfffffa800dadab30 ISWSVC.exe             1924    652     10      218      0      0 2015-
02-23 16:56:50 UTC+0000
0xfffffa800db98b30 spoolsv.exe            2032    652     19      433      0      0 2015-
02-23 16:56:51 UTC+0000
```

(truncated output)

Conclusion

Memory Forensic Analysis is a very interesting area and the first step to follow it is to acquire the memory as shown in this article. There are other good ways to acquire the memory such as using LiME with netcat (for remote acquisition), using the excellent KnTDD (a commercial tool found on http://www.gmgsystemsinc.com/knttools/) for local and remote dump, pausing a virtual machine (for example, one running on VMware Workstation) and coping the .vmem file (eventually, the .vmsn and .vmss files too) to analyze. Anyway, this article should be an incentive to start the studies on the fascinating Memory Forensic area by using Volatility.

Alexandre Borges.