

A decorative graphic on the right side of the page. It features three sets of concentric circles in shades of blue. Two thin blue lines originate from the top left and extend diagonally towards the circles. The circles are positioned at different heights and are partially cut off by the right edge of the page.

Live Memory Acquisition for Windows Operating Systems:

Tools and Techniques for Analysis

The live acquisition of volatile memory (RAM) is an area in digital forensics that has not garnered much attention until most recently. The importance of the contents of physical memory has always taken a back seat to what is considered more important – the contents of physical media. However, a great deal of information can be acquired from RAM analysis which is unavailable during most typical forensic acquisition and analysis. This paper will take a look at the different tools available to the forensic examiner for memory acquisition and how to analyze the resulting data.

Naja Davis
Eastern Michigan University
IA 328

Table of Contents

Cover Page and Abstract.....	1
I. Introduction	3
II. Scope	3
III. Tools for live memory acquisition.....	4
Hardware-based solutions	4
“Tribble”	4
Firewire	4
Software-based solutions	5
Limitations of software-based acquisition.....	5
“DD” (data dumper)	5
Nigilant32	6
ProDiscover IR	6
KntDD	6
Microsoft Crash Dump	7
IV. Memory Analysis.....	7
Basics: What does an investigator need to know?	7
Tools.....	8
V. Acquisition	10
Suggested Procedures for Live Acquisition:.....	11
VI. Test Case, Step-by-Step	11
VII. Conclusion.....	21
Appendix A.....	22
References	23

I. Introduction

Until recently, the acquisition of volatile memory (RAM) has been practiced mainly by those involved in live incident response and largely ignored by those in the field. Memory acquisition from a live system requires specialized hardware or software – not all forensic utilities can access the `\\.\PhysicalMemory` object in Windows. The analysis of the resulting image file also requires specialized scripts and knowledge to be able to interpret the data. These two factors make memory acquisition and analysis more difficult than traditional forensic hard drive examinations; it requires a greater amount of care than the common method of pulling the power and preserving the crime scene.

However, with the advent of Microsoft Vista and BitLocker – Microsoft’s answer to full-disk encryption – and the increasing sophistication of malware, rootkits, and other viruses, live memory analysis has become even more important to the field of computer forensics. Important data such as passwords, IP addresses, what processes were running, and other data that might not be stored on the hard drive can be retrieved from a memory dump or image. Malware and rootkits often leave traces in resident memory that cannot be found by analyzing a hard drive image.

The Digital Forensic Research Workshop (DFRWS) [1], issued a “memory analysis challenge” in the summer of 2005, to encourage research and tool development in live memory acquisition. This challenge produced two winners, Chris Betz and the team of George M. Garner, Jr. and Robert-Jan Mora, who developed tools to complete the challenge. Memparser [2], Chris Betz’s winning entry, reconstructs processes lists and extracts information from process memory. Garner and Mora developed kntlist, which enables an examiner to dump the physical memory from Windows and extract information from the resulting file. These two works have spurred interest in the field of live memory acquisition and the issues surrounding it.

II. Scope

All tools and procedures in this document apply only to the Windows family of operating systems, including Windows 2000, XP, Vista, and Server 2003.

III. Tools for live memory acquisition

Hardware-based solutions

“Tribble”

The Tribble [3] was introduced in February 2004 in the *Digital Investigation Journal* by Brian Carrier and Joe Grand, of Grand Idea Studio, Inc. The Tribble is a hardware expansion card which can be used to retrieve the contents of physical memory. It is a PCI expansion card designed to be installed on a server before the event, with a switch that is enabled when the investigator wants to capture data.

This method of acquisition has its strengths and limitations. As a hardware device, the Tribble can access physical memory without introducing any software onto the target system, minimizing the impact on the data being retrieved. However, it must be installed prior to the incident, making it somewhat inconvenient for on-the-fly acquisition. It is also still a proof-of-concept device and not widely available.

Firewire

The second hardware solution available for live memory acquisition is through the use of a Firewire device. Firewire devices use direct memory access (DMA), without having to go through the CPU. The memory mapping is performed in hardware without going through the host operating system, which allows not only for high-speed transfers but also bypasses the problem with some versions of Windows that do not allow memory to be accessed from User mode.

Adam Boileau [4] developed software using Python to extract physical memory from a system on Linux. This tool can be used on Windows systems as well, by tricking Windows into giving the user DMA by masquerading as an iPod. This method is more convenient than the aforementioned Tribble device, as most systems today have Firewire ports available (usually built right into the motherboard). The current problem with this method is an issue with the Upper Memory Area (UMA) which causes some systems to suffer crashes during the acquisition process [5].

Software-based solutions

Limitations of software-based acquisition

With the release of Service Pack 2 for Windows XP the `\\.\PhysicalMemory` object is no longer accessible from user mode. This is also true for Windows Vista and Windows Server 2003 (Service Pack 1) - it can only be accessed via kernel-mode drivers. As such, some utilities which may have worked in the past will no longer work on versions of Windows. They may still apply to earlier or unpatched versions, however.

One issue that the forensic investigator needs to remain mindful of during live memory acquisition with software-based tools is the potential change to data during the acquisition process. Due to the volatile nature of RAM, introducing any new software onto the system may change the data which currently resides in memory. The memory introduced to the system will displace the data that previously occupied that space. The image acquired may also present a 'smeared' picture of the data, since the system is live and pages are changing as the acquisition progresses. This is certainly not ideal for forensically sound acquisition and subsequent analysis and must be given due consideration, particularly when evidentiary rules and standards apply.

"DD" (data dumper)

DD, better known as the "data dumper" tool from UNIX, is probably familiar to most forensic investigators as a tool for creating forensic images of hard drives and is included in many open source forensic utilities such as Helix (<http://www.e-fense.com/helix/>). The DD format is also supported by most major forensic applications. Forensic Acquisition Utilities (FAU) [6] uses a modified version of the data dumper tool which is capable of accessing the `\\.\PhysicalMemory` object in Windows. Unfortunately FAU will only work on versions earlier than Windows XP Service Pack 2, Windows Vista, or Server 2003 Service Pack 1, as it accesses the `PhysicalMemory` from user mode. (Note: The most recent version of FAU does *not* include a version of DD that works for memory acquisition – previous versions are still viable however). Also, not all versions of DD will allow access to the `\\.\PhysicalMemory` object.

Nigilant32

Nigilant32 [7] is a tool developed by Agile Risk Management that allows an investigator to preview a hard disk, image memory, and take a 'snapshot' of current running processes and open ports on the target system. Nigilant32 has a small footprint, using less than 1 MB in memory when loaded, supporting Agile's claim of minimal impact during acquisition. The program is currently in beta, however, it is free to download and use off of their website.

ProDiscover IR

Technology Pathway's forensic acquisition tool, ProDiscover [8], is an incident response tool that allows investigation of a live system anywhere on the network. The investigation can include imaging of physical media or memory, however, use of this tool requires a server applet to be installed on the target system prior to acquisition via removable storage media such as a USB drive or CD. This requirement makes this particular tool not as desirable a choice for field acquisition and perhaps better suited to a corporate network environment. (Note: This tool is restricted by the kernel-mode driver requirement for accessing \\.\PhysicalMemory in certain versions of Windows).

KntDD

KntDD is a memory acquisition tool developed by George Garner (also responsible for the Forensic Acquisition Toolkit) as a part of KntTools [9]. Garner developed KntTools in response to the restriction of accessing \\.\PhysicalMemory from User mode and supports Windows 2000 through Vista. Images can be acquired to a local removable drive or across the network. It also allows the investigator to convert a raw image to Microsoft crash dump format, so the data can be analyzed using the Microsoft Debugging Tools. This tool is only available to law enforcement or security professionals.

Microsoft Crash Dump

Analyzing crash dumps is another way to obtain information on the contents of RAM. Unlike other software methods of memory acquisition, the image obtained by a crash dump is an unaltered copy of the contents of a system's memory at the time the crash occurred. There is no introduction of software to the system that will alter the contents of memory. The drawback to this method is that crash dumps only occur when there is a problem with the system. There is a method to induce a crash dump; however, it requires an entry in the registry along with a reboot before it is useable [10], rendering it ineffective for field acquisition.

Despite this shortcoming, it is still important for an investigator to familiar with crash dumps as they can provide valuable information about a system. Not all versions of Windows generate full crash dumps and may generate smaller sized dumps. These files can be analyzed with the Windows Debugging Tools [11] and can give the investigator a means to practice and become familiar with memory analysis.

IV. Memory Analysis

Basics: What does an investigator need to know?

The EProcess structure is what represents a process on a Windows system. It includes information on the different attributes of the process along with pointers to other attributes and data structures which are related to it. However, EProcess block structure varies between operating systems, including between different versions of Windows. Typically, the offsets vary from version to version. It is important to make note of the version of Windows that the memory image or dump is taken from, as this will affect what tools you may be able to use to extract information. This can be done manually, however, it requires a bit more in-depth knowledge of Windows memory management than this paper covers. Harlan Carvey has written a Perl script [12], `osid.pl`, which will identify the operating system of an image.

The EProcess block contains the process environment block (PEB) which is very valuable to a forensic investigator in that it includes pointers to the loader data, such as modules used by the process. This is particularly useful in malware or rootkit analysis, but can also help present a clearer picture as to what exactly was going on in the system at the time in question.



The PEB also shows us where the image of the executable lies, the DLL paths, and the command line used to launch the process.

One issue that investigators need to be aware of when examining an image of memory, is that most likely it is not a complete picture. Windows memory management uses virtual addressing which assigns pointers to the true location of the physical data. According to Jesse Kornblum in his “Using every part of the buffalo in Windows memory analysis” [13], most memory analysis tools use a ‘naïve’ form of translation where pages with invalid pointers are ignored. Memory pages which have been swapped out due to paging will not show up in a memory dump, although they are on the system in the page file. All the tools tested in this paper do not (as far as this author is aware), include the page file. There are tools in development to address this issue, although none are publicly available (yet).

Tools

Due to the diligence of the computer forensics community, there are quite a few tools available to the investigator with which to analyze memory dumps. Some technical knowledge or familiarity with command-line interaction is recommended as many of the available tools are scripts which must be executed from a command prompt. There are only a few tools which have a GUI interface.

The following is a list of tools which can be used to extract process and other information from memory dumps (links to download locations will be included in Appendix A of this document):

Tool	Operating System	What it does	Requirements
Lsproc.pl	Windows 2k	Locates processes	Perl (http://www.perl.org)
Lspd.pl	Windows 2k	Lists details of processes	Perl (http://www.perl.org)
Osid.pl	Any	Identifies OS of	Perl (http://www.perl.org)

	Windows	memory image.	
PoolFinder (part of PoolTools)	Windows 2k, XP	Finds allocations of OS kernel in memory dump and pagefile.	Perl (http://www.perl.org)
PoolGrep (part of PoolTools)	Windows 2k, XP	Finds strings in pool allocations	Perl (http://www.perl.org)
PoolDump (part of PoolTools)	Windows 2k, XP	Hex dump of all allocations for a selected class.	Perl (http://www.perl.org)
PTFinder	Windows 2k, XP	Includes all scripts in PoolTools as well as osid.pl, but has a GUI. Produces graphical output of processes and threads.	Perl (http://www.perl.org) Graphviz (http://www.graphviz.org/) and ZGRViewer (http://zvtm.sourceforge.net/zgrviewer.html) to view the generated graphic file.
FTimes	Windows NT, XP, 2K	Comprehensive toolkit with various memory analysis functions.	If running in a Windows environment, you will need Visual Studio in order to compile and run the code. Requires advanced user knowledge.
Volatility	Windows NT, XP, 2K	Comprehensive toolkit with various memory analysis functions.	Needs Python to run. This can be accomplished in the Windows environment by installing Cygwin (http://www.cygwin.com/)

The above tools mainly deal with process information, which is where the bulk of memory forensic analysis has been focused. Other data can be extracted from a memory image as well, such as usernames, passwords, and email addresses. A good string search utility, such

as find.exe or strings.exe is essential. Forensic Tools such as AccessData's Forensic Toolkit [14] can be used to data carve to retrieve documents, graphic files, or web pages. One important note about data carved from memory images is to keep in mind that the data was retrieved under *volatile* conditions. As such, files retrieved from memory may be degraded due to the data not being static. This is illustrated by the following picture, carved from a test memory image:



V. Acquisition

Due to the volatile nature of live forensics, an investigator needs to develop a standard set of procedures. This is important not only to insure that the investigator knows exactly what

to do when arriving on the scene, but also so there are no unexpected consequences since the system is live - unintentionally changing data on the target system could invalidate the acquired evidence and also cause it to be inadmissible in a court of law. Before attempting a live acquisition, an investigator should test their toolset(s) extensively, under varying conditions (VMware [15] is excellent for this).

Suggested Procedures for Live Acquisition:

1. Document *all* steps. This is not only important for evidentiary reasons, but also for the investigator's own reference.
2. Is the system locked? If so, that will change the acquisition process. If you cannot obtain a password for access, then live acquisition may not be possible. Currently, no software utilities can image \\.\PhysicalMemory without full access.
3. Do *not* close any windows or close any documents/programs – leave them running. By closing a window or program you may be terminating a process, which will affect what is occurring on the system at that time.
4. Limit the acquisition process to as few steps as possible, when it comes to interacting with the target system – fewer steps = less impact on the system.
5. Use tools that have as small a footprint as possible. Nigilant32 (this author's recommended choice) uses less than 1MB of memory; Helix uses 17MB.

VI. Test Case, Step-by-Step

Test system:

VMWare, Windows XP Professional Service Pack 2

Intel Dual Core Processor 2.6 MHz

512 MB RAM

Tool used for image acquisition: Nigilant32



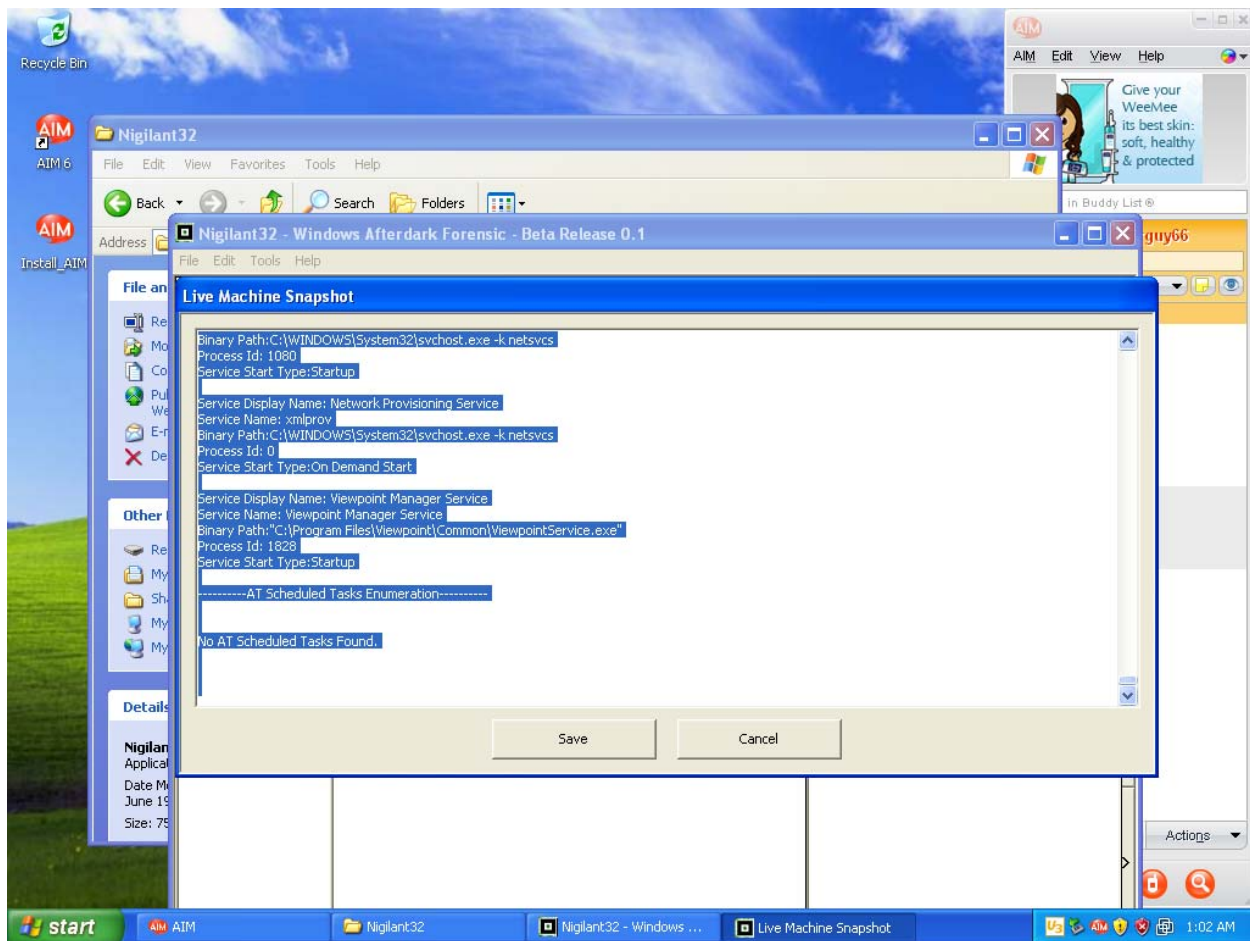
Desktop before live acquisition:



AOL Instant Messenger can be seen running.

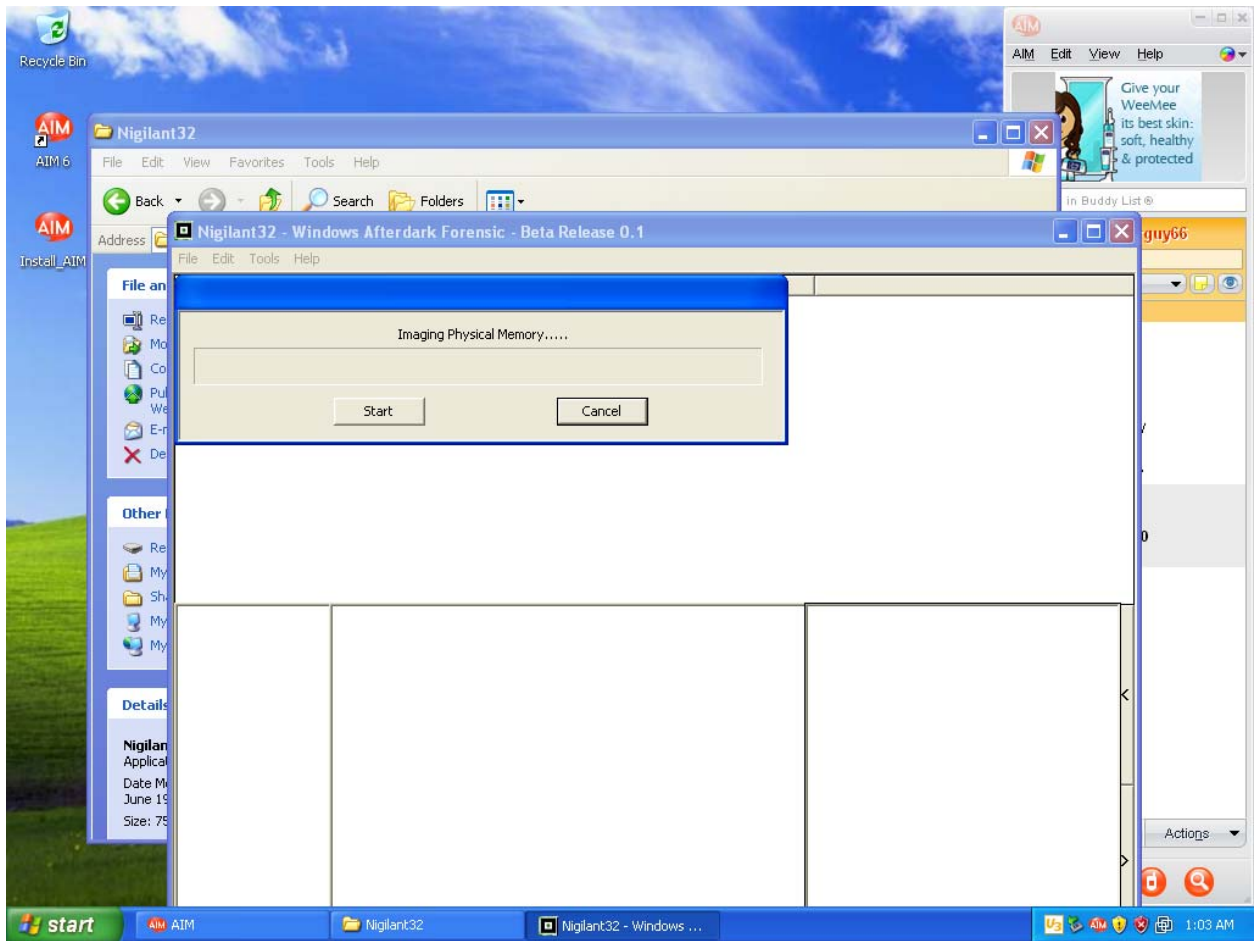
1. For this acquisition I chose to use a USB thumb drive for storing the image. Investigators should remember to wipe media thoroughly before each acquisition, so remnants of data from previous images are not a factor in analysis.

After inserting your CD with the Nigilant software on it, browse to My Computer and explore the drive (if it doesn't already open due to Auto Run). Run the Nigilant32 executable and go to Tools → Snapshot Computer. This option will enumerate the currently running processes, users, and open ports and allow the investigator to save this data to a plain text file. Save the text file to your thumb drive, naming it appropriately. You can also enumerate processes via other scripts after image acquisition, if you wish to validate this output.



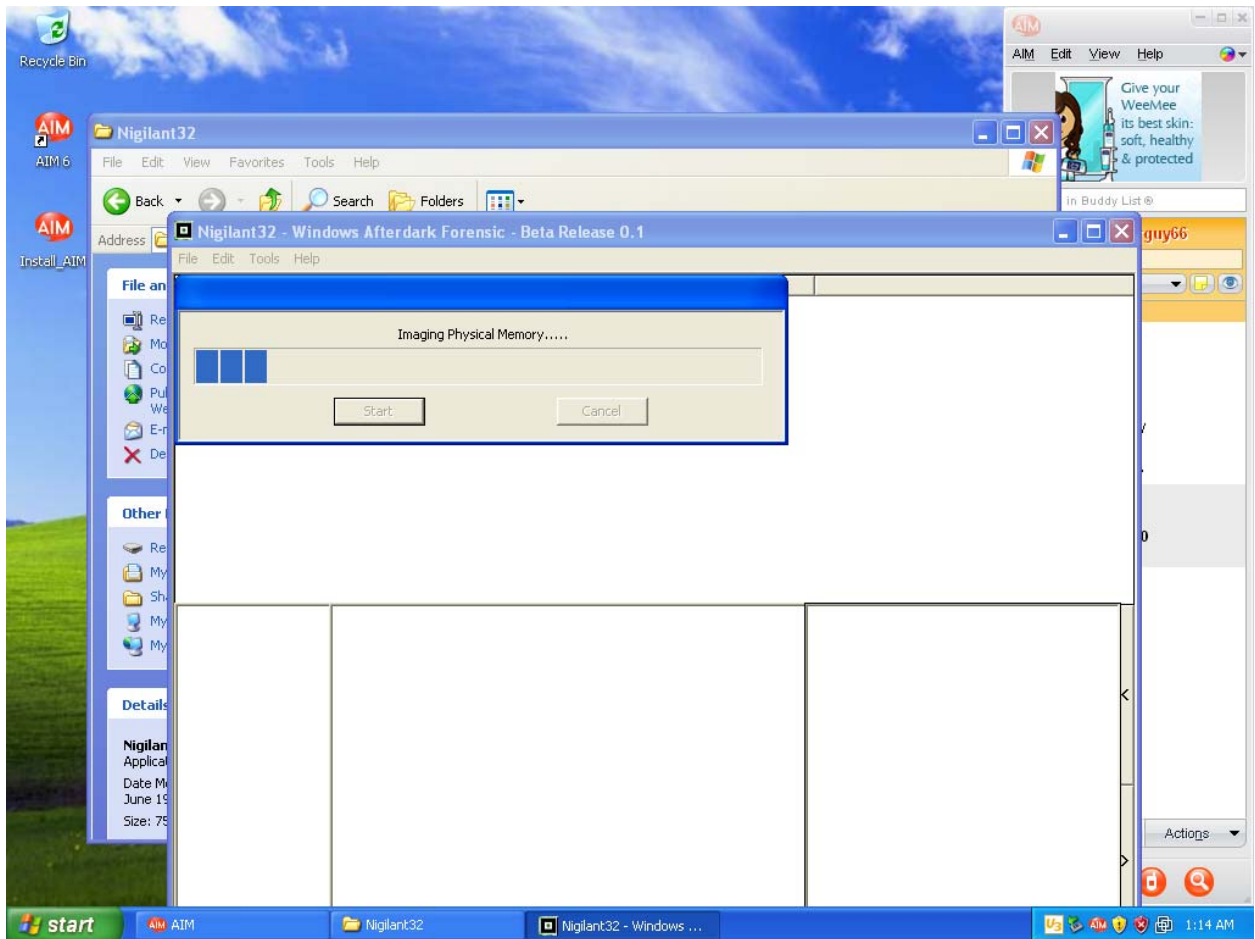
Note: You can put the Nigilant executable on the thumb drive and run it from there, however, be mindful if your data will be used as evidence. It may be best to burn it to a CD with your other memory acquisition tools, so there is no question as to the integrity of your image.

2. After saving the text file, browse to Tools → Image Physical Memory. A prompt will appear – click on 'Start'



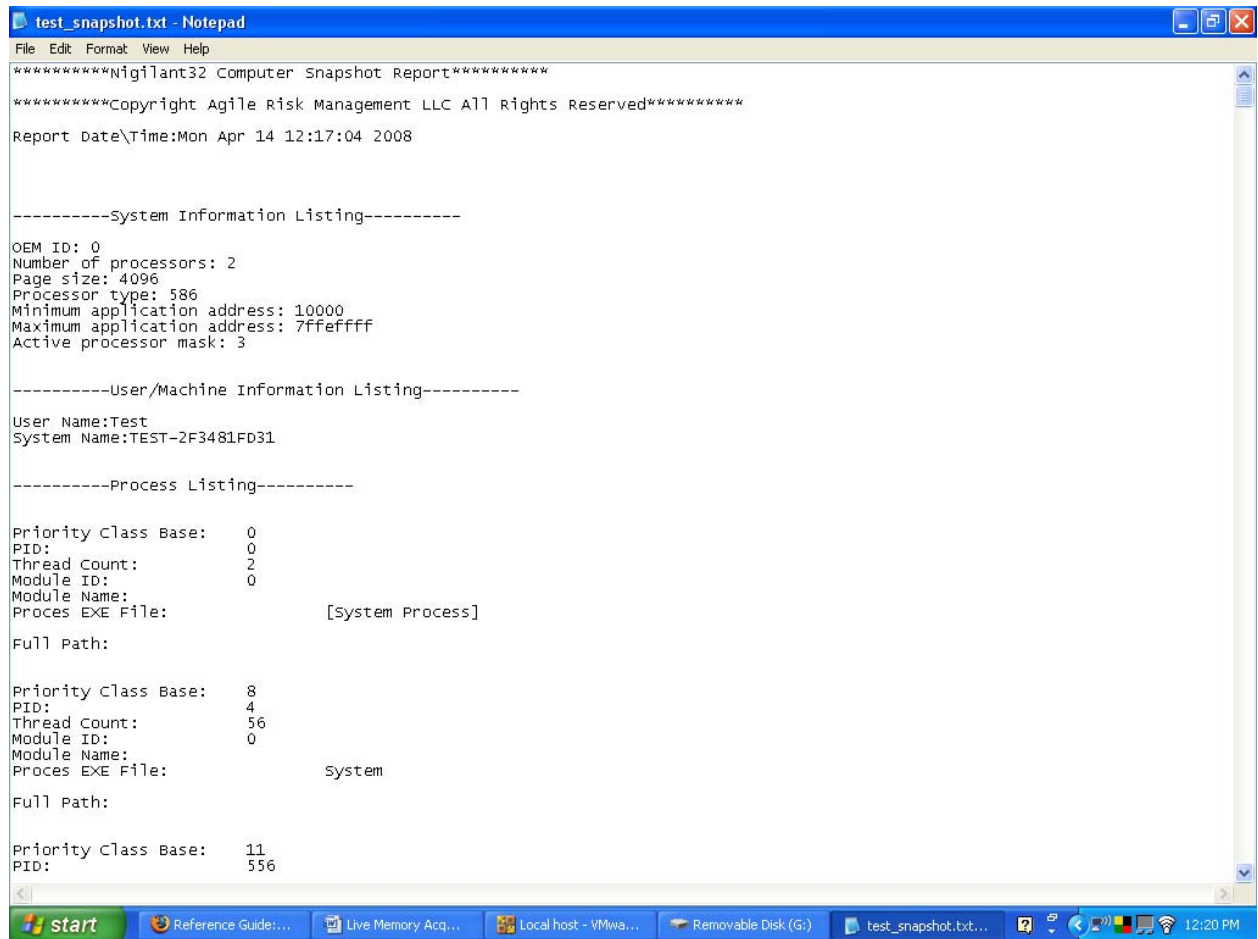
You will be prompted to choose a location and name for your image.

Acquiring physical memory takes a bit of time, as with normal data acquisition. A progress indicator will appear to let you know how far along you are:



3. After the image is complete, close the Nigilant software. Unfortunately, Nigilant does not have an ability to hash the image file after acquisition – the investigator will have to do this before beginning analysis.
4. Before beginning analysis, the investigator should make another copy of the memory image to work on – never work on the original media! Since this isn't like a hard drive acquisition, there is no original *physical* media – the image we just made *is* the original. For evidentiary purposes, it is a good practice to hash the original media (the thumb drive) and the memory image and make a working copy of the memory image before proceeding with analysis.

- As discussed earlier, memory analysis differs from hard drive analysis in that even slight changes in operating system version (Windows 2k vs. Windows XP) will determine which tools will be the most effective. Nigilant32 has done a lot of the work for us already, by providing us with a snapshot of the OS version, running processes, users, and open network ports:



```
test_snapshot.txt - Notepad
File Edit Format View Help
*****Nigilant32 Computer Snapshot Report*****
*****Copyright Agile Risk Management LLC All Rights Reserved*****
Report Date\Time:Mon Apr 14 12:17:04 2008

-----System Information Listing-----
OEM ID: 0
Number of processors: 2
Page size: 4096
Processor type: 586
Minimum application address: 10000
Maximum application address: 7ffeffff
Active processor mask: 3

-----User/Machine Information Listing-----
User Name:Test
System Name:TEST-2F3481FD31

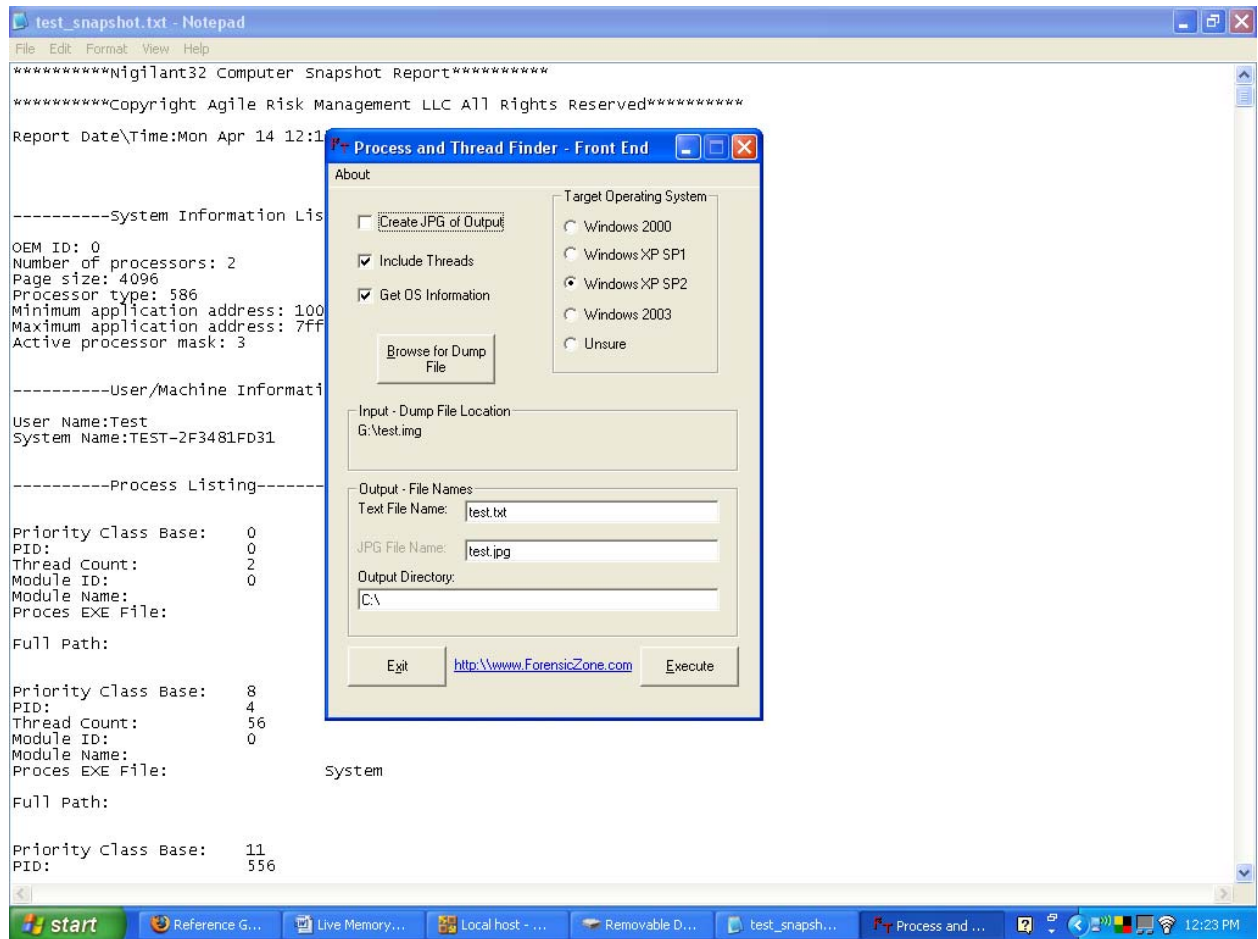
-----Process Listing-----

Priority Class Base: 0
PID: 0
Thread Count: 2
Module ID: 0
Module Name:
Proces EXE File: [System Process]
Full Path:

Priority Class Base: 8
PID: 4
Thread Count: 56
Module ID: 0
Module Name:
Proces EXE File: System
Full Path:

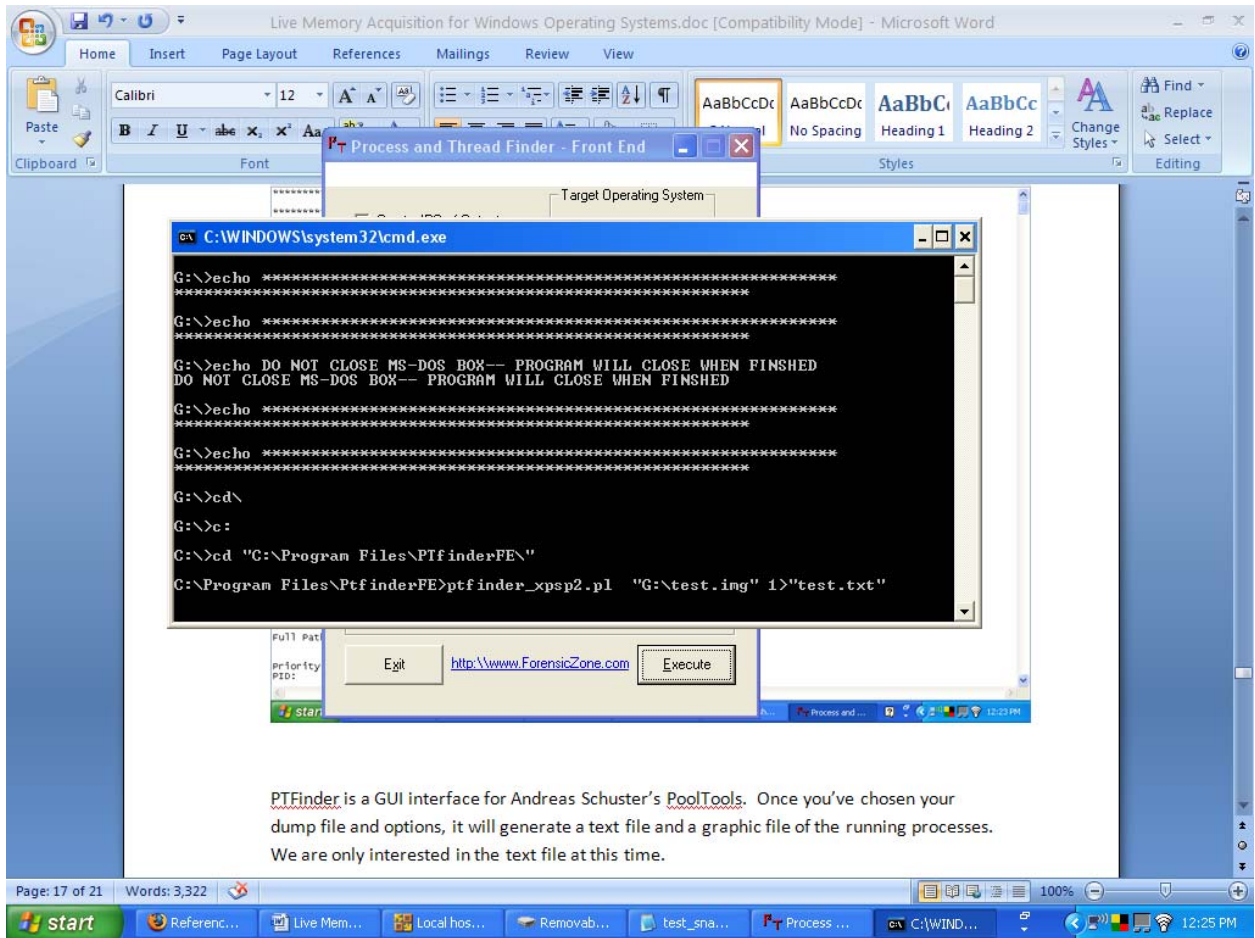
Priority Class Base: 11
PID: 556
```


An investigator could verify output by running another analysis tool and enumerating the processes. I will demonstrate this here by using PTFinder:



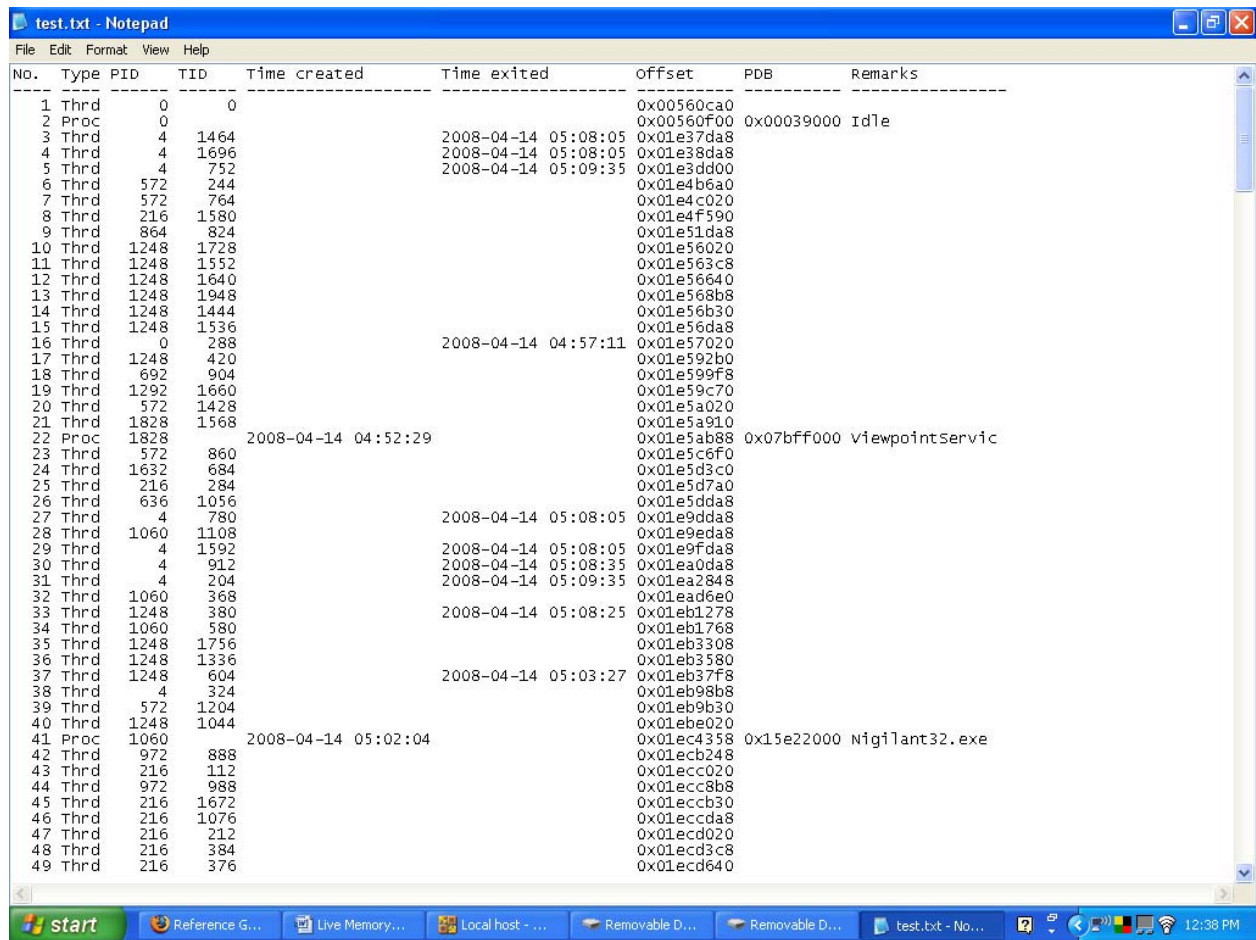
PTFinder is a GUI interface for Andreas Schuster's PoolTools. Once you've chosen your dump file and options, it will generate a text file and a graphic file of the running processes. We are only interested in the text file at this time. After clicking 'Execute' you will be prompted to run a batch file – click 'Yes'.

A DOS prompt will open up:



When the analysis is complete, PTfinder will close on its own.

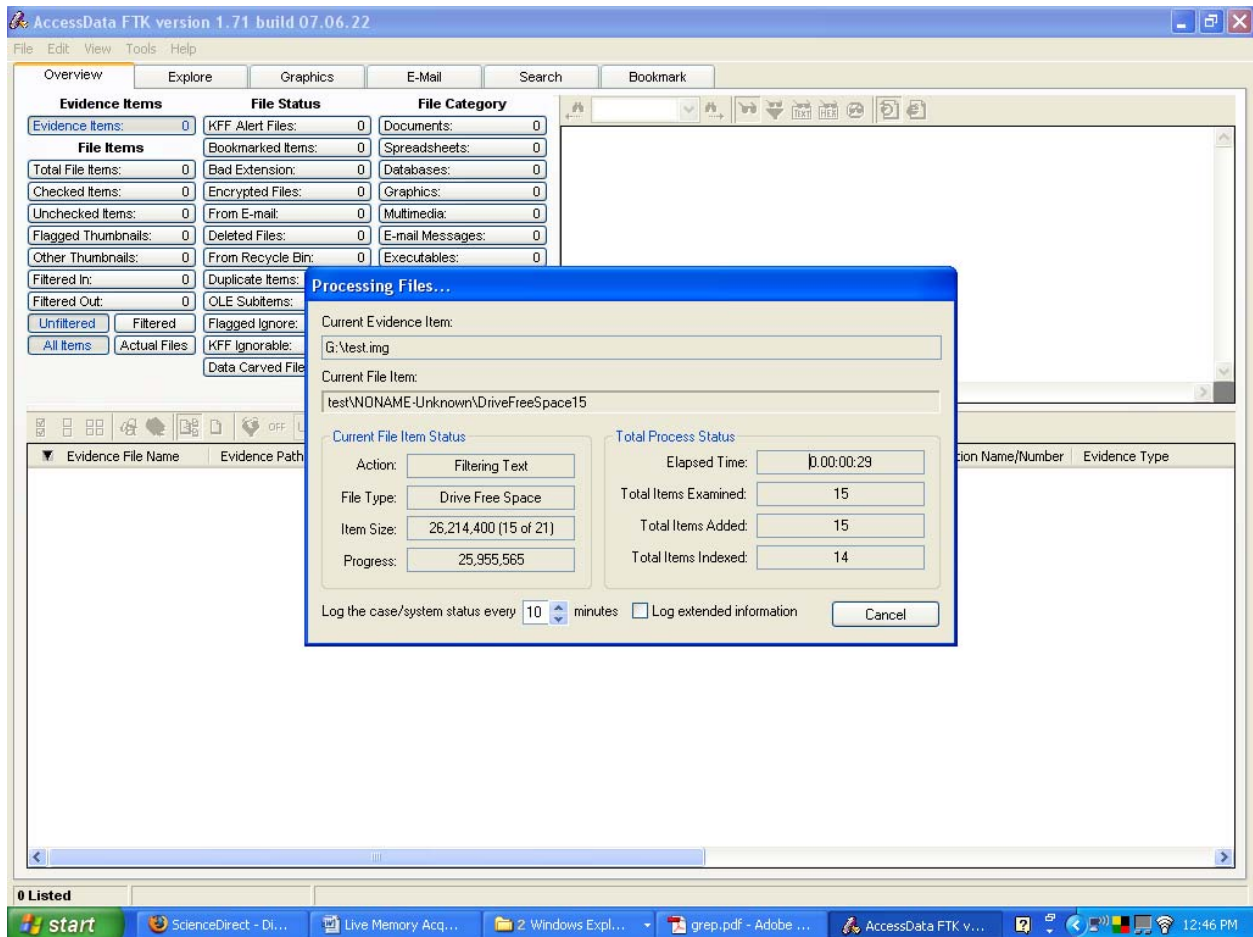
The resulting text file looks like this:



No.	Type	PID	TID	Time created	Time exited	Offset	PDB	Remarks
1	Thrd	0	0			0x00560ca0		
2	Proc	0				0x00560f00	0x00039000	Idle
3	Thrd	4	1464		2008-04-14 05:08:05	0x01e37da8		
4	Thrd	4	1696		2008-04-14 05:08:05	0x01e38da8		
5	Thrd	4	752		2008-04-14 05:09:35	0x01e3dd00		
6	Thrd	572	244			0x01e4b6a0		
7	Thrd	572	764			0x01e4c020		
8	Thrd	216	1580			0x01e4f590		
9	Thrd	864	824			0x01e51da8		
10	Thrd	1248	1728			0x01e56020		
11	Thrd	1248	1552			0x01e563c8		
12	Thrd	1248	1640			0x01e56640		
13	Thrd	1248	1948			0x01e568b8		
14	Thrd	1248	1444			0x01e56b30		
15	Thrd	1248	1536			0x01e56da8		
16	Thrd	0	288		2008-04-14 04:57:11	0x01e57020		
17	Thrd	1248	420			0x01e592b0		
18	Thrd	692	904			0x01e599f8		
19	Thrd	1292	1660			0x01e59c70		
20	Thrd	572	1428			0x01e5a020		
21	Thrd	1828	1568			0x01e5a910		
22	Proc	1828		2008-04-14 04:52:29		0x01e5ab88	0x07bff000	ViewpointService
23	Thrd	572	860			0x01e5c6f0		
24	Thrd	1632	684			0x01e5d3c0		
25	Thrd	216	284			0x01e5d7a0		
26	Thrd	636	1056			0x01e5dda8		
27	Thrd	4	780		2008-04-14 05:08:05	0x01e9dda8		
28	Thrd	1060	1108			0x01e9eda8		
29	Thrd	4	1592		2008-04-14 05:08:05	0x01e9fda8		
30	Thrd	4	912		2008-04-14 05:08:35	0x01ea0da8		
31	Thrd	4	204		2008-04-14 05:09:35	0x01ea2848		
32	Thrd	1060	368			0x01ead6e0		
33	Thrd	1248	380		2008-04-14 05:08:25	0x01eb1278		
34	Thrd	1060	580			0x01eb1768		
35	Thrd	1248	1756			0x01eb3308		
36	Thrd	1248	1336			0x01eb3580		
37	Thrd	1248	604		2008-04-14 05:03:27	0x01eb37f8		
38	Thrd	4	324			0x01eb98b8		
39	Thrd	572	1204			0x01eb9b30		
40	Thrd	1248	1044			0x01ebe020		
41	Proc	1060		2008-04-14 05:02:04		0x01ec4358	0x15e22000	Nigilant32.exe
42	Thrd	972	888			0x01ecb248		
43	Thrd	216	112			0x01ecc020		
44	Thrd	972	988			0x01ecc8b8		
45	Thrd	216	1672			0x01eccb30		
46	Thrd	216	1076			0x01eccda8		
47	Thrd	216	212			0x01ecd020		
48	Thrd	216	384			0x01ecd3c8		
49	Thrd	216	376			0x01ecd640		

The output from PTFinder is not as clean as what you will see from Nigilant, but provides more than enough information to compare running processes. Note: PTFinder will not provide network information or users, only process information.

- Now that we have process information, we can proceed with analyzing the image file with other tools. In this case, we will use Forensic Toolkit:



After analyzing the image the investigator can examine carved data and perform string searches as with a normal image file.

VII. Conclusion

While there are many tools available for live memory acquisition and analysis, it is still a relatively new endeavor in the area of digital forensics; many of the tools and techniques developed thus far are still in the growing phase and require refinement. Today's computer forensic investigator, in order to be successful, will need to be well-informed and be intimately familiar with the internal workings of Windows memory management in order to acquire a complete picture of memory from an evidentiary standpoint. Thankfully there have been many forensic investigators, such as Harvey Carlan, Andreas Schuster, and Mariusz Burdach who have started along the path and created a foundation for others to build upon. As the tools become better and the procedures more sound, examiners will have a new weapon in their arsenal to utilize during forensic investigations.

Appendix A

Lsproc.pl – http://sourceforge.net/project/showfiles.php?group_id=164158

Lspd.pl – http://sourceforge.net/project/showfiles.php?group_id=164158

Osid.pl – http://sourceforge.net/project/showfiles.php?group_id=164158

PoolTools (PoolFinder, PoolGrep, PoolDump) –
http://computer.forensikblog.de/en/2007/11/pooltools_1_3_0.html

PTFinder – http://computer.forensikblog.de/en/2006/03/ptfinder_0_2_00.html

FTimes - <http://ftimes.sourceforge.net/FTimes/>

Volatility - <https://www.volatilesystems.com/VolatileWeb/volatility.gsp>

References

1. Digital Forensics Research Workshop, “DFRWS”, <http://www.dfrws.org/>. [Accessed March 15, 2008]
2. C. Betz, “Memparser”, <http://sourceforge.net/projects/memparser>. [Accessed March 15, 2008]
3. B. D. Carrier and J. Grand, “A Hardware-Based Memory Acquisition Procedure for Digital Investigations” *Journal of Digital Investigations*, March 2004.
4. A. Boileau, “Firewire and DMA”, March 2008, <http://www.storm.net.nz/projects/16>. [Accessed March 16, 2008].
5. A. Vidstrom, “Memory dumping over Firewire – UMA Issues”, <http://www.ntsecurity.nu/onmymind/2006/2006-09-02.html>. [Accessed March 16, 2008].
6. G. Garner, “Forensic Acquisition Utilities”, November 2007, <http://gmgsystemsinc.com/fau/>. [Accessed March 20, 2008].
7. Agile Risk Management, “Nigilant32”, http://www.agilerm.net/publications_4.html. [Accessed March 20, 2008].
8. Technology Pathways, “Prodiscover IR”, <http://www.techpathways.com/ProDiscoverIR.htm>. [Accessed March 20, 2008].
9. GMG Systems, Inc, “KntTools with KntList”, <http://www.gmgsystemsinc.com/knttools/>. [Accessed March 20, 2008].
10. Microsoft, Inc., “Windows feature lets you generate memory dump file by using the keyboard”, December 2007, <http://support.microsoft.com/kb/244139>. [Accessed March 21, 2008].

11. Microsoft, Inc., "Debugging Tools for Windows – Overview", <http://www.microsoft.com/whdc/DevTools/Debugging/default.mspix>. [Accessed March 21, 2008].
12. J. Kornblum, "Using every part of the buffalo in Windows memory analysis", *Digital Investigation*, vol. 4, issue 1, pp 24-29. March 2007.
13. H. Carvey, *Windows Forensic Analysis*, Burlington, MA: Syngress Publishing, 2007.
14. AccessData, "Forensic Toolkit 2.0", <http://www.accessdata.com/Products/ftk2test.aspx>. [Accessed March 22, 2008]
15. VMWare, "VMWare Server", <http://www.vmware.com/products/server/>. [Accessed April 8, 2008]