



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Reverse-Engineering Malware: Malware Analysis Tools and Techniques (Forensics)"
at <http://www.giac.org/registration/grem>



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Reverse-Engineering Malware: Malware Analysis Tools and Techniques (Foren
at <http://www.giac.org/registration/grem>

Review of Windows 7 as a malware analysis environment

GIAC (GREM) Gold Certification

Author: Adam Kramer, adamkramer@hotmail.com

Advisor: Mark Stingley

Accepted: January 2nd, 2014

Abstract

The SANS course “FOR610 – Reverse Engineering of Malware” is designed around the use of Windows XP as a malware analysis environment. Mainstream support for this operating system ended over 4 years ago and for those with extended support contracts, these are due to end April 2014. This will prompt both enterprises and home users to consider the transition to a newer version of Windows which has ongoing support and regular updates. As users transition, the malware and therefore analyst will need to follow. In order to be ahead of the game and ready for the coming changes we will look at how each of many of the main tools used by malware analysis perform under Windows 7 x64 detailing any issues encountered and giving consideration to possible solutions.

1. Introduction

The SANS course “FOR610: Reverse Engineering of Malware” is designed using Windows XP as the malware analysis environment (SANS Institute, 2013). According to the Microsoft support lifecycle, their mainstream support for Windows XP ended over four years ago and for those with extended support contracts, they are due to end in April 2014 (Microsoft, 2013). In order to have access to software updates and maintain security, enterprises and consumers alike will be required to update to a newer version. As users upgrade to a new operating system, malware authors and therefore the malware analyst will need to follow. The cautious corporate administrator has the option of upgrading to either the latest version “Windows 8” or the more mature “Windows 7”. The online publication techrepublic.com has run a survey every 6 months since 2009 to try and identify the trend in operating system migration away from Windows XP and what people are choosing to move to. The indication in March 2013 was that companies had progressively been moving to “Windows 7” and 85% either had no plans or had decided against moving to “Windows 8” (Tech Republic, 2013).

In this review, the tools and techniques taught in FOR610 will be tested under Windows 7 x64 to establish on a tool by tool basis whether they are fully functional, can be made to work with modifications to the system or whether they will not work at all and in which case what alternatives are available.

2. Test environment

2.1 Design of tool testing program

A benign program called “Harmless_Malware.exe” was written and compiled in C++ to emulate the Windows system API calls and functions found in common malware. Access to the source code allows anyone conducting the tests to verify what the results should be (available as Appendix A).

This program has the following functionality:

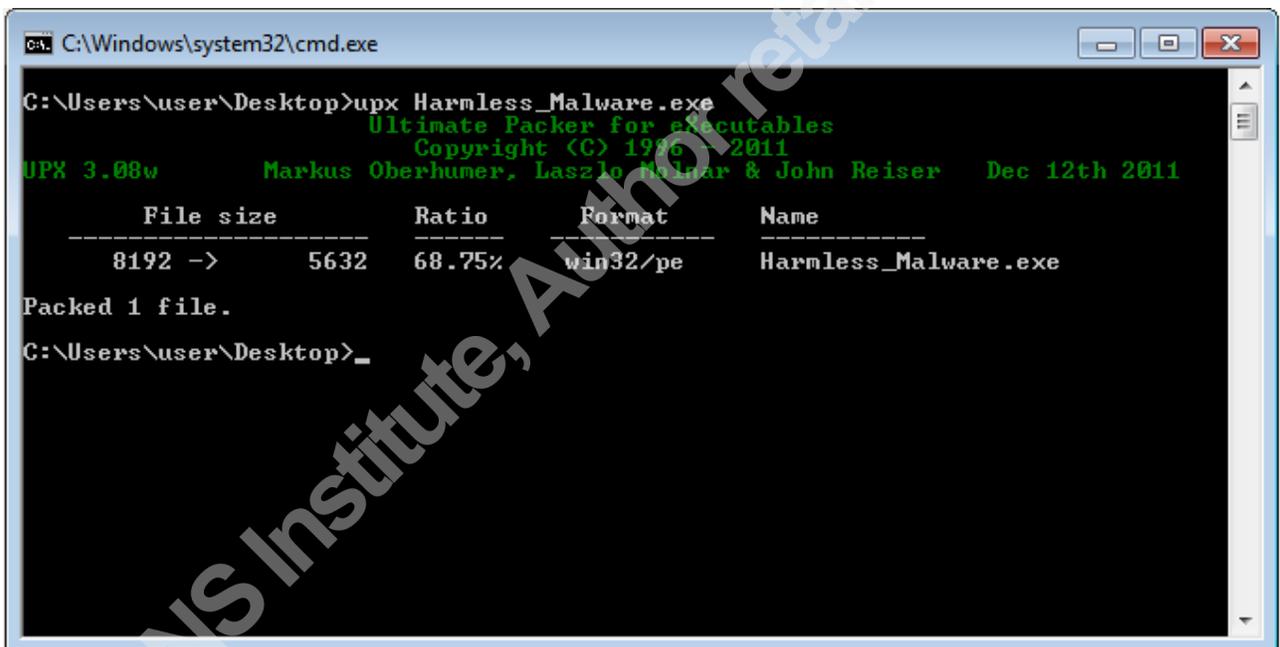
1. The process will identify the path it was executed from and add itself to the “Run” key in the registry in order that it be automatically started when the system loads.
2. The process will attempt to write the EICAR standard antivirus test string (EICAR, 2003) to a file (“avtest.txt”). This will emulate the functionality of a “dropper”.
3. The process will attempt to open an internet connection to www.sans.org and retrieve the homepage.

These three tasks will cover persistence, file system modification and network communication which are three tasks common to malware functionality.

The test program was created with the support of the MSDN online Windows API support pages and is available freely for anyone wishing to conduct similar testing (MSDN, 2013).

Consideration was given to the architecture in which to compile this code to. It was decided that as x86 programs can be run on both x86 and x64 platforms that this would be the favored choice of the malware author as their product would work on a higher proportion of systems. A x86 program run on x64 Windows uses the WoW64 system which supports the backwards compatibility of architecture execution (Microsoft - Dev Center (A), 2013).

Following the compilation of the code into “**Harmless_Malware.exe**” UPX packer (UPX, 2013) was used to create a packed version of the program which was named “**Harmless_Malware (packed).exe**”. This would become useful when testing tools and techniques which work on packed executables.



```

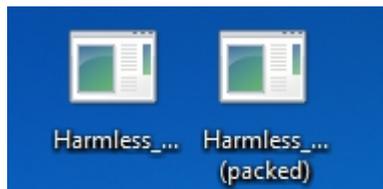
C:\Windows\system32\cmd.exe
C:\Users\user\Desktop>upx Harmless_Malware.exe
Ultimate Packer for executables
Copyright (C) 1996-2011
UPX 3.08w Markus Oberhumer, Laszlo Molnar & John Reiser Dec 12th 2011

  File size      Ratio      Format      Name
-----
  8192 ->    5632    68.75%    win32/pe    Harmless_Malware.exe

Packed 1 file.
C:\Users\user\Desktop>_

```

The resultant executables were then ready for the testing process to begin.



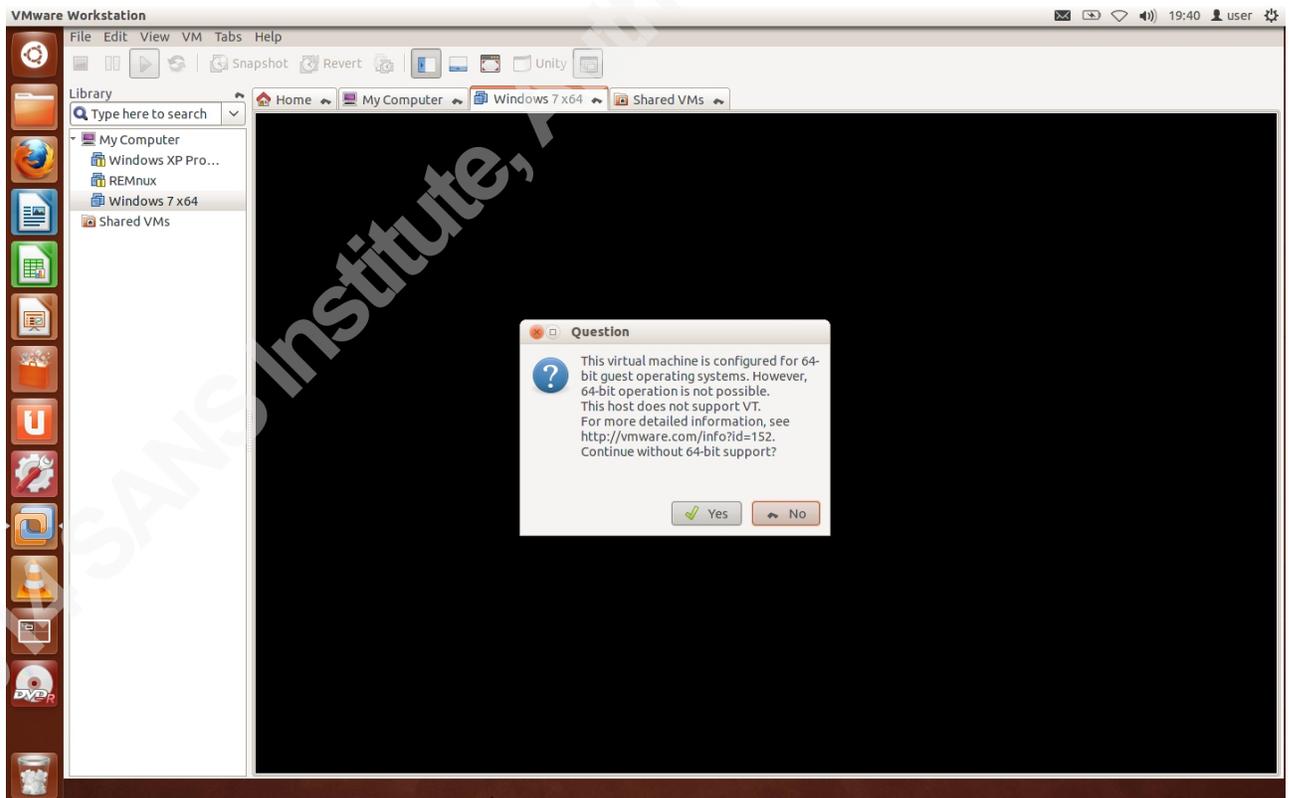
2.2 Setup of the Windows 7 x64 environment

At this point the setup of the Windows 7 x64 environment was started. Initially VMware was used to install the operating system to as this was the technique used in the FOR610 course and would allow quick roll back and the ability to save system states to aid with testing.

Unfortunately, very quickly an issue arose with the installation of a x64 operating system on VMware.

A dialog box appeared as soon as the operating system installer was launched with the following message:

“This virtual machine is configured for 64-bit guest operating systems. However, 64-bit operation is not possible. This host does not support VT. For more detailed information, see <http://vmware.com/info?id=152>. Continue without 64-bit support?”

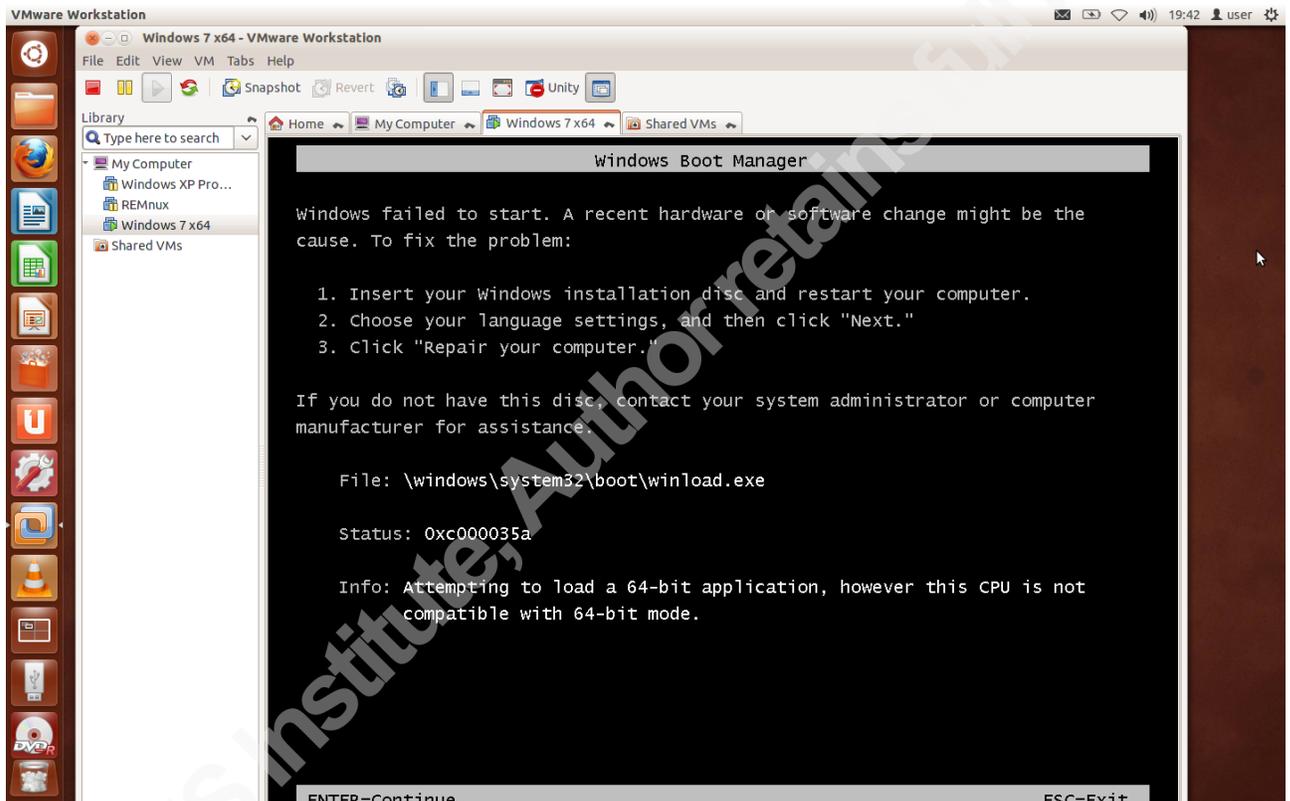


Adam Kramer, adamkramer@hotmail.com

On proceeding a message was presented which was part of the Windows 7 x64 installation process and gave the error message shown in the screenshot below.

As follows:

“Info: Attempting to load a 64-bit application, however this CPU is not compatible with 64-bit mode”



It appears that the hardware of the machine which was being used to do this testing was causing issues while loading the “winload.exe” program which formed part of the installation process. The advice provided by VMware was reviewed but it was not possible to get it working. At this point there was a choice of changing to a different machine or installing the operating system directly rather than using a virtual environment.

At this point a partition was created and Windows 7 was installed directly to the machine to prevent any further VM related issues.

Adam Kramer, adamkramer@hotmail.com

3. Individual tool testing

A selection of the core tools from the malware analysts toolkit as taught in FOR610 were tested using the benign sample and their compatibility with “Windows 7” is summarized below and detailed in the following pages.

Tool	Functionality	Identified issues
BinText	Fully functional	None
Process Monitor	Fully functional	None
Process Hacker	Fully functional	None
Autoruns	Fully functional	None
PEiD	Fully functional	None
Capture BAT	Partially functional	Network traffic only
Regshot	Fully functional	None
Lord PE	Appears functional *	Possible issue with automated header repair
CHimpREC	Testing failed	Loading & functionality issues – see report
Ollydbg v2	Appears functional *	None
IDA Pro	Appears functional *	None

Please note:

Where a tool is marked **Appears functional *** this is a recognition of the complex nature of code analysis, including the unreliable “trial and error” nature of process dumping & header repair in the analysis of a packed executable.

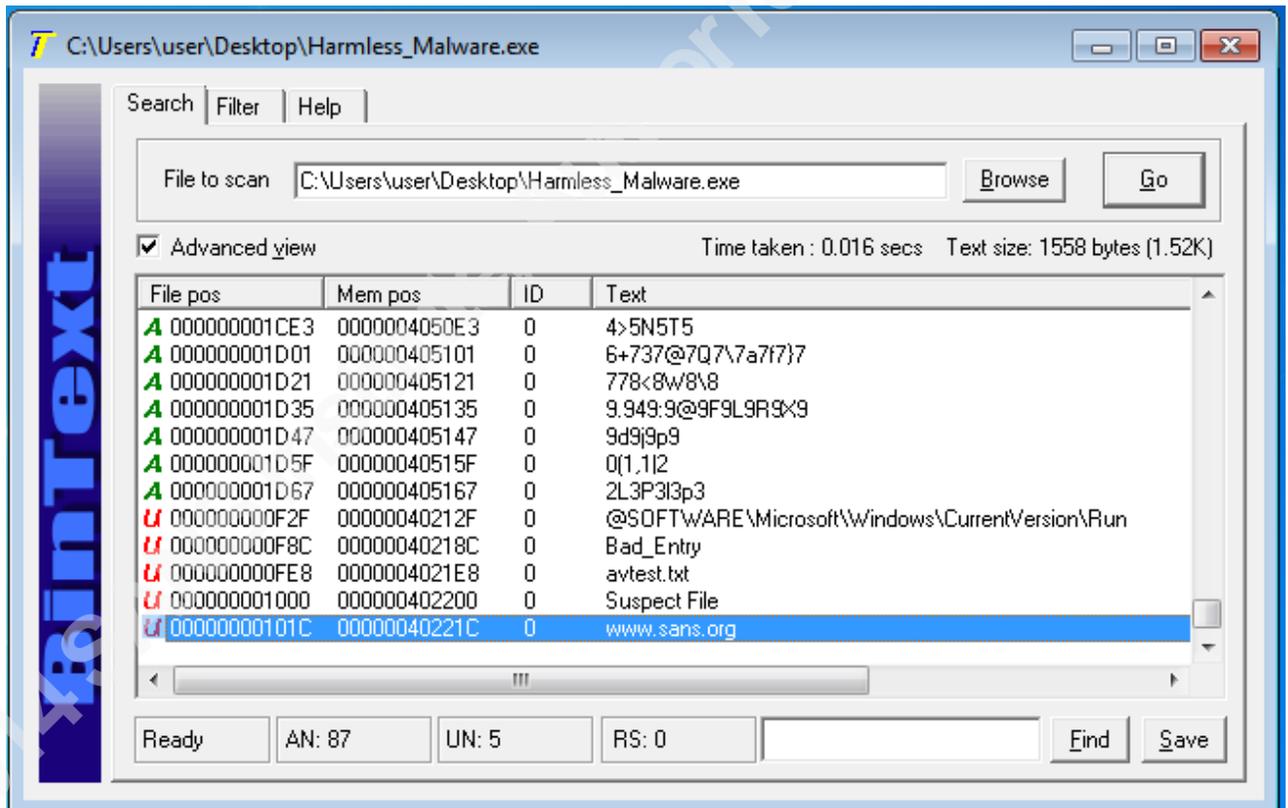
The individual report pages detail the testing that was completed and results achieved.

3.1 BinText – Fully functional

(BinText, 2013)

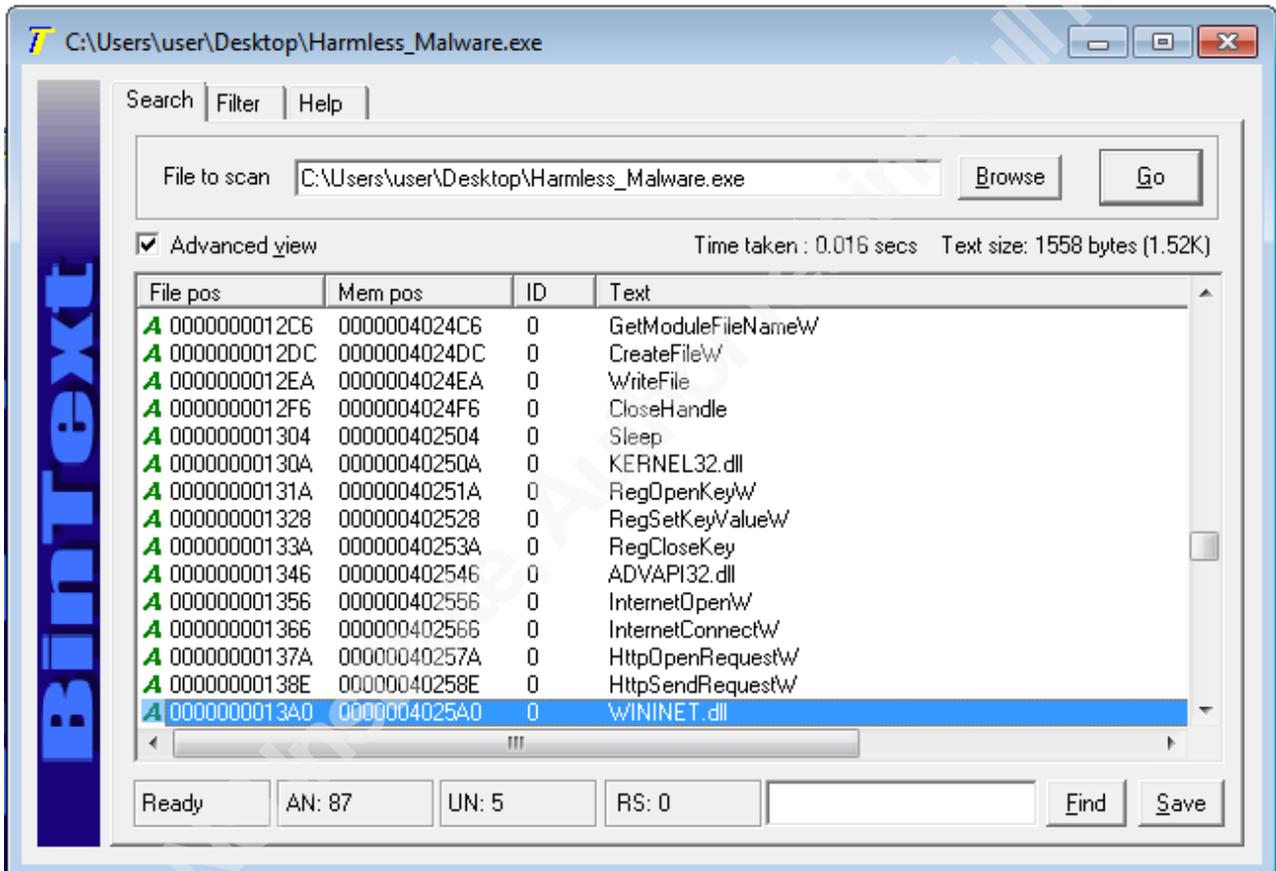
This tool was run against the unpacked executable and as can be seen in the screenshot below was able to identify various strings from static variables that we would expect to see including:

- SOFTWARE\Microsoft\Windows\CurrentVersion\Run
- Bad_Entry
- avtest.txt
- Suspect File
- www.sans.org



Adam Kramer, adamkramer@hotmail.com

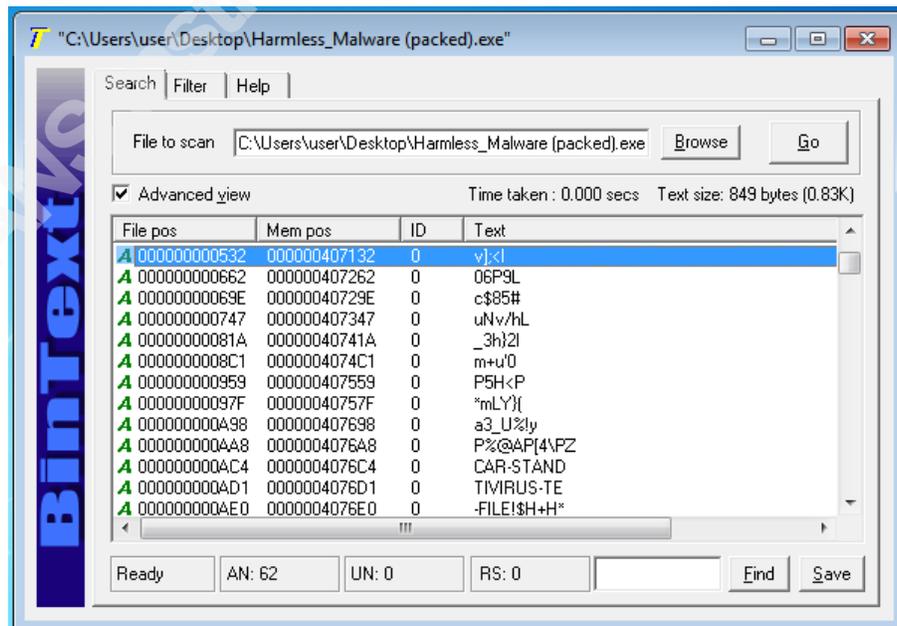
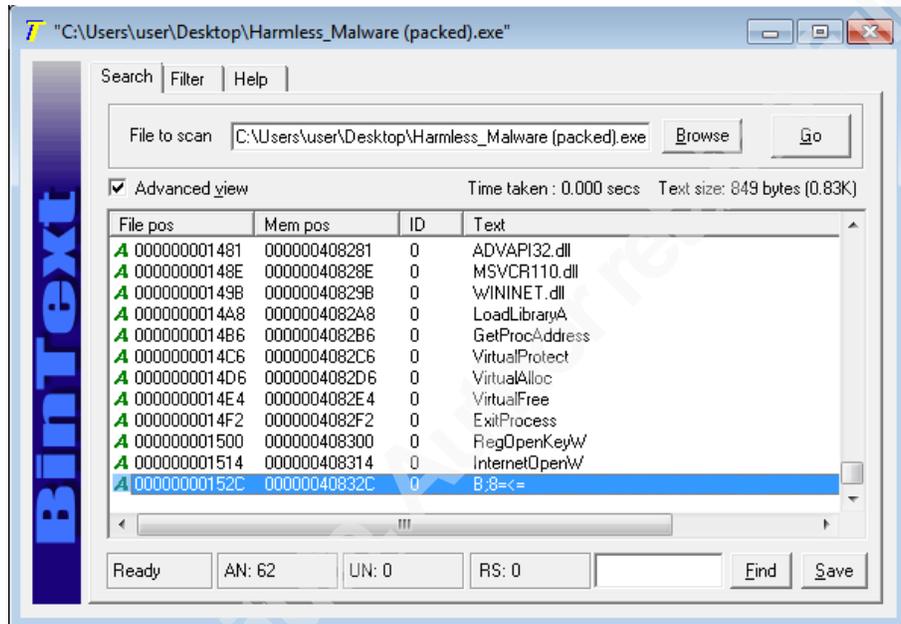
Further results from the same executable reveal API calls which were used in the program. We can see the calls made, in fact everything listed in the screenshot below would have been called or linked in to the program somehow. If we were using this tool in the primary stages of an analysis we would be able to identify some of the functionality of the program without much work.



Adam Kramer, adamkramer@hotmail.com

Now we can move on to the packed executable and as we can see from the two screenshots below there are some indications of the functionality of the packed program (in fact there are more clear text API calls than would typically be expected to seen from a packed executable).

Either way the tool was performing as expected and outputting correctly.



Adam Kramer, adamkramer@hotmail.com

3.2 Process Monitor – Fully functional

(Process Monitor, 2013)

Process monitor is designed to capture all the Windows API functions that have been called and log them in an interface that can be filtered and saved.

The screenshot below shows a small segment of the output which resulted from filtering to only show the CreateFile operations from our process.

As ever, there are vast numbers of Windows API calls which are being done in the background, but of note a CreateFile operation was highlighted which is linking in with the avtest.txt file.

As well as correctly capturing this call, Process Monitor managed to capture all the relevant calls made during the program including the full details and results.

Time ...	Process Name	PID	Operation	Path	Result	Detail
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\Prefetch\HARMLESS_MALWAR...	NAME NOT FOUND	Desired Access: Gener
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows	SUCCESS	Desired Access: Execu
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64win.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64win.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64cpu.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64cpu.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\System32\wow64log.dll	NAME NOT FOUND	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Users\user\Desktop	SUCCESS	Desired Access: Execu
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\sechost.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\sechost.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Users\user\Desktop\MSVCR110.dll	NAME NOT FOUND	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\msvcr110.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\msvcr110.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Users\user\Desktop\version.DLL	NAME NOT FOUND	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\version.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\version.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\imm32.dll	SUCCESS	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Users\user\Desktop\avtest.txt	SUCCESS	Desired Access: Gener
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Users\user\Desktop\Secur32.dll	NAME NOT FOUND	Desired Access: Read
12:29:...	Harmless_Malw...	2644	CreateFile	C:\Windows\SysWOW64\secur32.dll	SUCCESS	Desired Access: Read

Showing 132 of 117,452 events (0.11%) Backed by virtual memory

Adam Kramer, adamkramer@hotmail.com

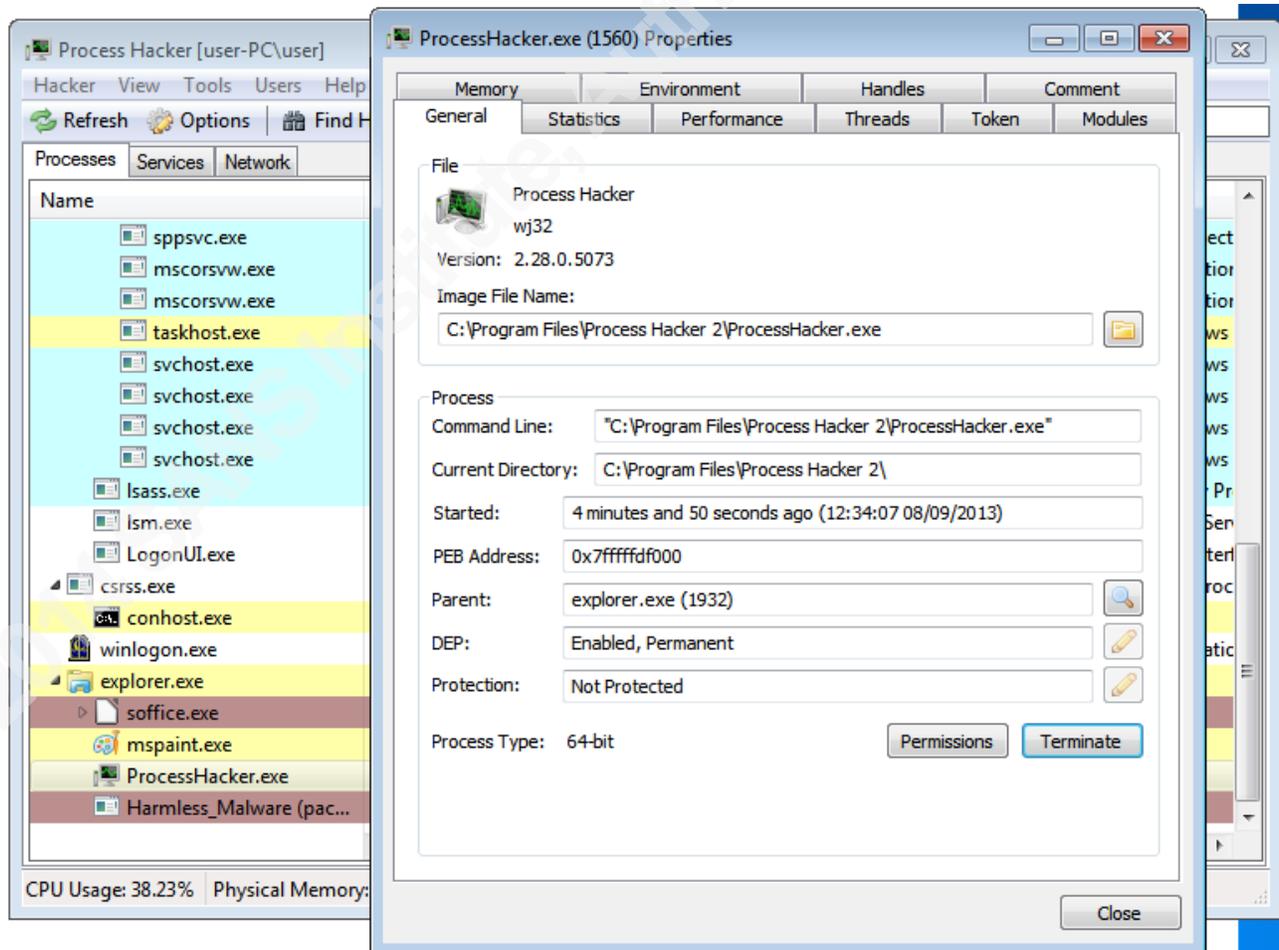
3.3 Process Hacker – Fully functional

(Process Hacker, 2013)

Process Hacker is a significantly more powerful task manager than the one built into Windows. It is able to correctly identify and detail both x86 and x64 bit processes and the tests into the functionality and detailing came back showing that it was operating as would be expected.

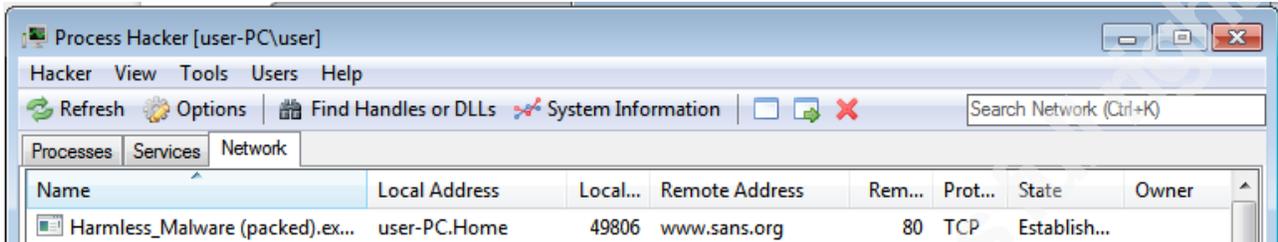
As shown in the screenshot below, our process is listed in the background. The properties window has been opened for process hacker itself and as we can see it is able to successfully show the overview details about the process.

The powerful functionality such as the ability to terminate a process directly was tested and was also a success.

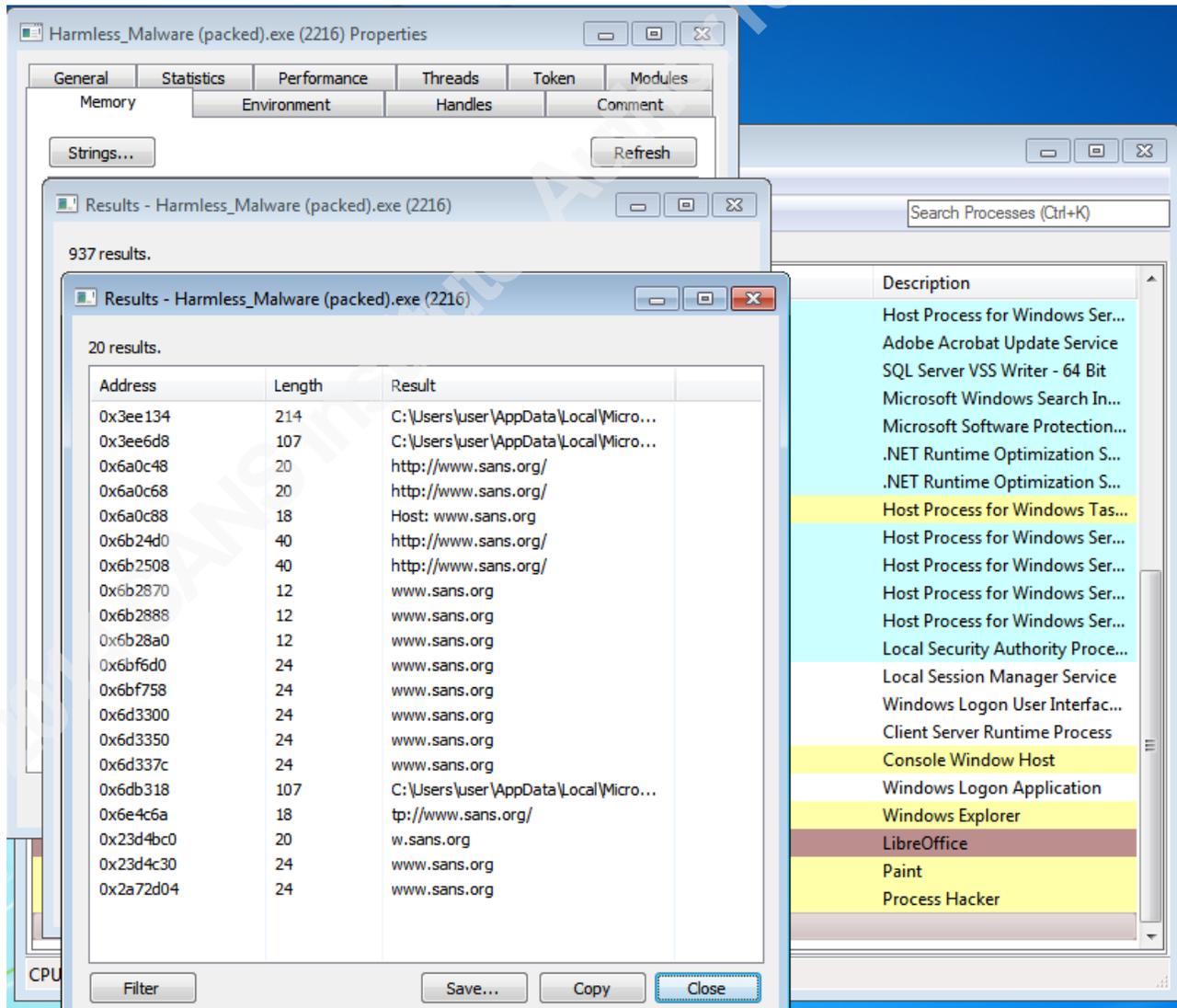


Adam Kramer, adamkramer@hotmail.com

Below is a screenshot of the network tab which show that it was able to successfully identify and display the connection that our program has made to www.sans.org and details the ports and current state.



Again, further functionality has been tested showing that it is able to successfully access the memory and extract strings on request.



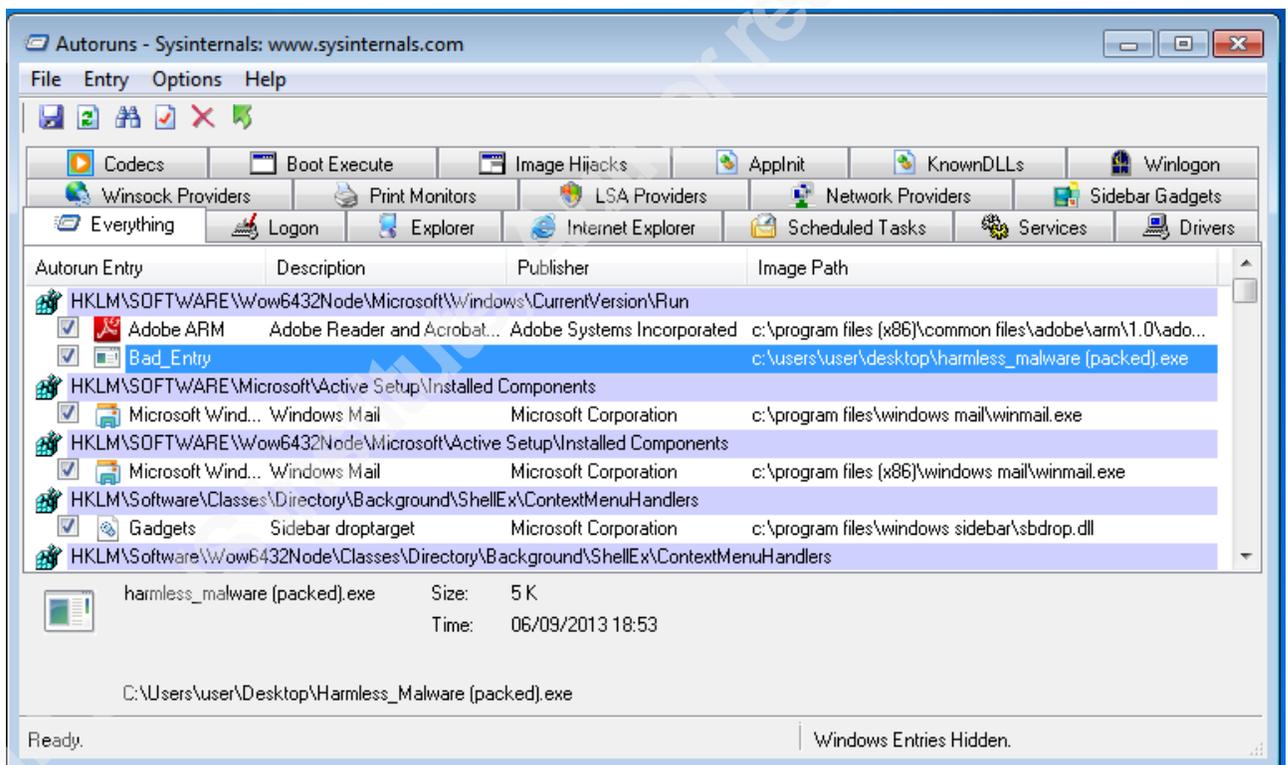
Adam Kramer, adamkramer@hotmail.com

3.4 Autoruns – Fully functional

(Autoruns, 2013)

We can see below that 'Autoruns' has been able to successfully identify and list the auto run entry that we added into the registry. This has been highlighted below, and we are able to use this software to modify the auto-run functionality.

As this entry has been created by a x86 program, the registry key enter was added under the Wow6432Node. This software is able to successfully identify and list that fact.



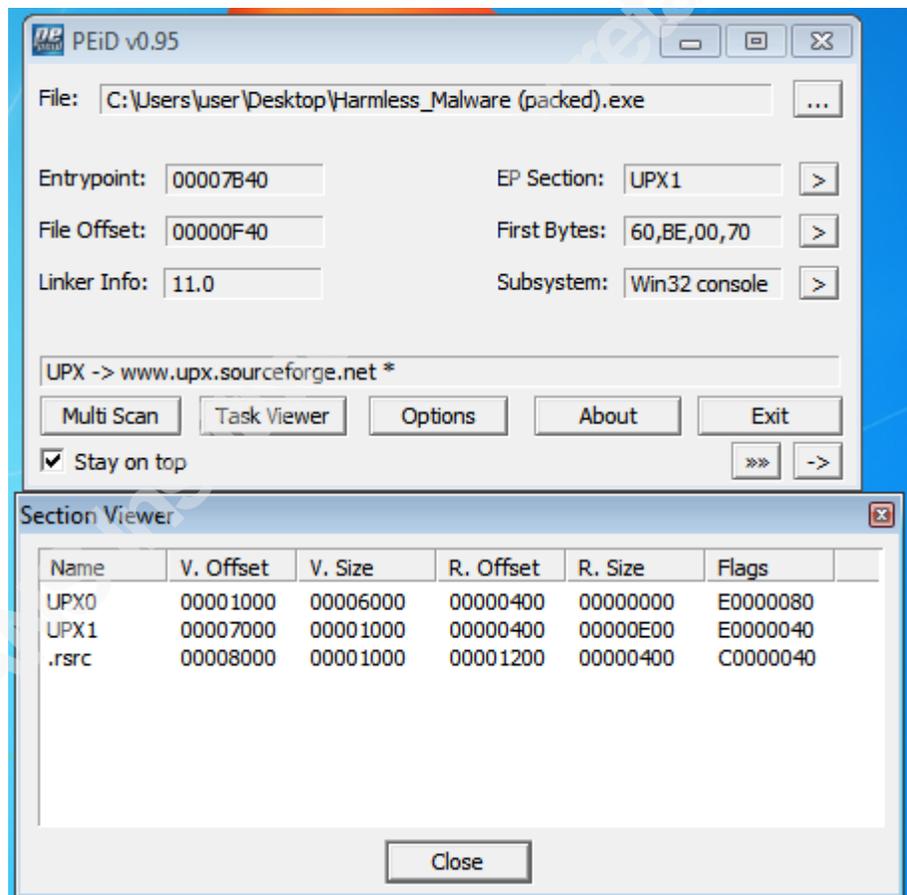
Adam Kramer, adamkramer@hotmail.com

3.5 PEiD – Fully functional

(PEiD, 2013)

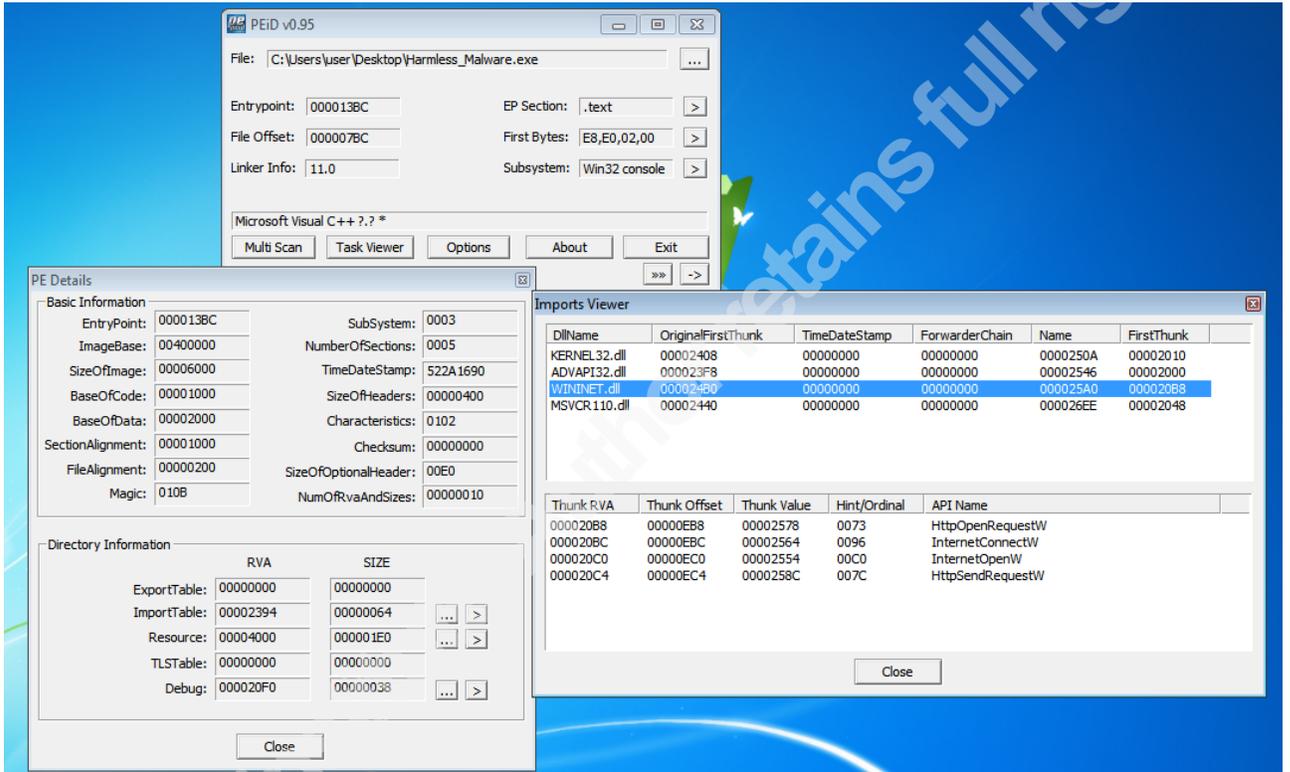
The packed executable was loaded into PEiD to see whether it would be able to identify the packer.

As can be seen, it has been able to identify the packer we have used as UPX and gives details about the sections within the file.



Adam Kramer, adamkramer@hotmail.com

Furthermore, digging into the functionality which details PE information including the import table and other headers.



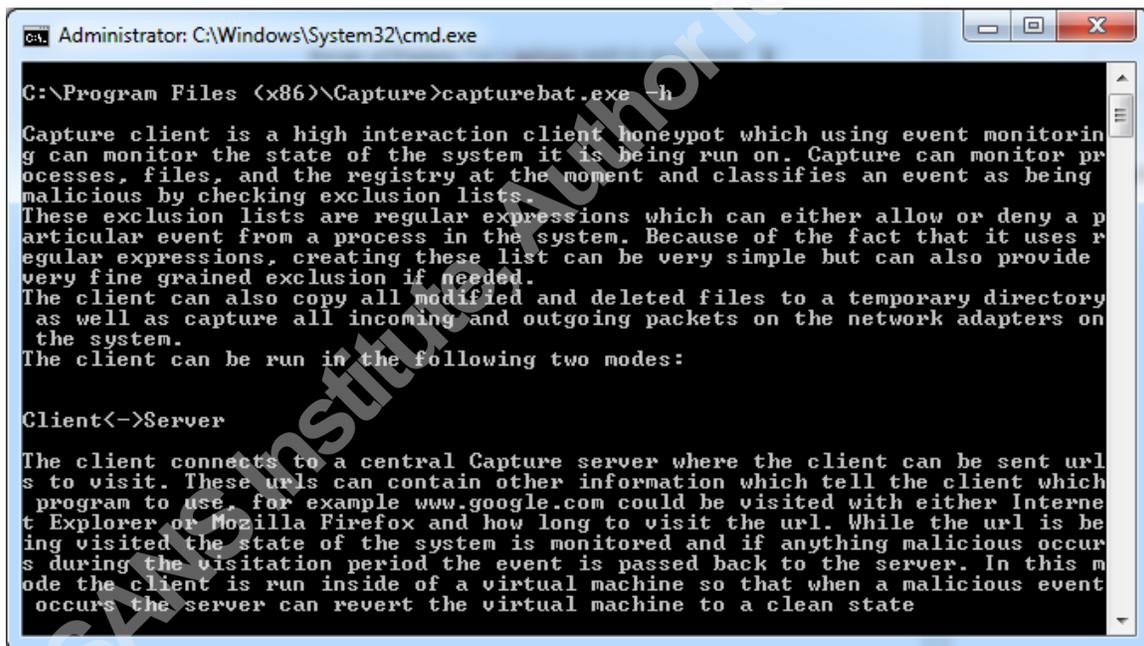
In the IAT view, APIs that have been used in the test executable can be easily recognized, in this screenshot the APIs used in the internet connection are seen.

3.6 Capture BAT – Partial functionality

(Capture BAT, 2013)

CaptureBAT is designed to monitor the system and attempts to identify suspicious activity. It has the functionality to monitor processes, files, registry and network traffic.

By running CaptureBAT with the -h argument, the help screen is displayed. This provides a detailed explanation of the full functionality of the program.



```
Administrator: C:\Windows\System32\cmd.exe

C:\Program Files (x86)\Capture>capturebat.exe -h

Capture client is a high interaction client honeypot which using event monitoring can monitor the state of the system it is being run on. Capture can monitor processes, files, and the registry at the moment and classifies an event as being malicious by checking exclusion lists. These exclusion lists are regular expressions which can either allow or deny a particular event from a process in the system. Because of the fact that it uses regular expressions, creating these list can be very simple but can also provide very fine grained exclusion if needed. The client can also copy all modified and deleted files to a temporary directory as well as capture all incoming and outgoing packets on the network adapters on the system.

The client can be run in the following two modes:

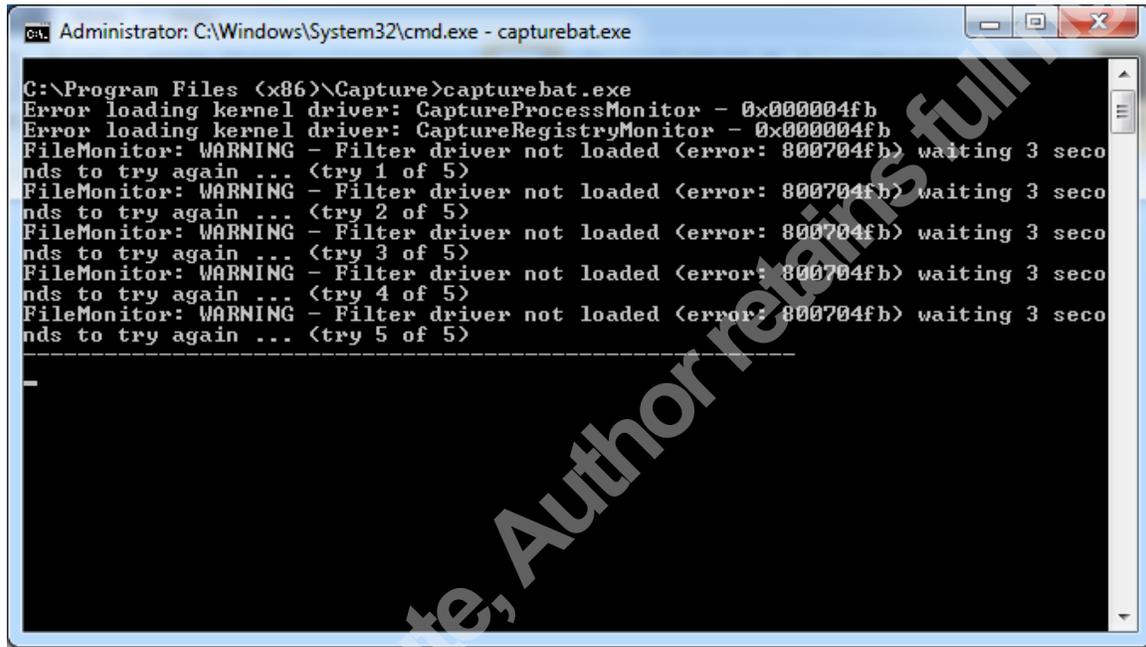
Client<->Server

The client connects to a central Capture server where the client can be sent urls to visit. These urls can contain other information which tell the client which program to use, for example www.google.com could be visited with either Internet Explorer or Mozilla Firefox and how long to visit the url. While the url is being visited the state of the system is monitored and if anything malicious occurs during the visitation period the event is passed back to the server. In this mode the client is run inside of a virtual machine so that when a malicious event occurs the server can revert the virtual machine to a clean state
```

Unfortunately although the program will execute and display what it was designed to do, it doesn't work fully under Windows 7 x64.

Adam Kramer, adamkramer@hotmail.com

If we execute the program without any parameters we get the following output:



```
Administrator: C:\Windows\System32\cmd.exe - capturebat.exe
C:\Program Files (x86)\Capture>capturebat.exe
Error loading kernel driver: CaptureProcessMonitor - 0x000004fb
Error loading kernel driver: CaptureRegistryMonitor - 0x000004fb
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 1 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 2 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 3 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 4 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 5 of 5)
-----
-
```

As can be seen the user is presented with error 0x4FB which according to the Microsoft 'System Error Codes' (Microsoft – Dev Center (B), 2013)

```
ERROR_DRIVER_BLOCKED
1275 (0x4FB)
```

This driver has been blocked from loading.

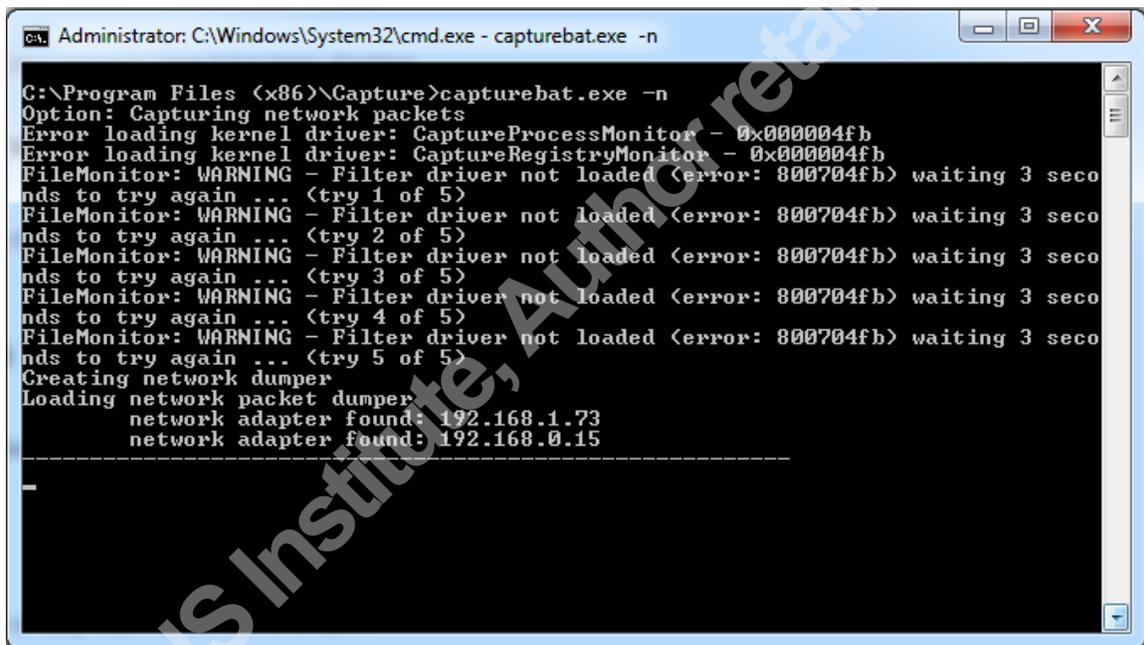
Windows 7 is not happy with the driver and has not allowed it to load. The kernel driver provides the core functionality of the program and accordingly without it, the execution fails.

However, there is limited functionality that does work under Windows 7. If the program is run with the -n argument, according to the help screen (-h) it will do the following:

Adam Kramer, adamkramer@hotmail.com

- n Capture all incoming and outgoing network packets from the network adapters on the system and store them in .pcap files in the log directory

This element of the functionality appears to work correctly and loads after the errors thrown from the kernel driver load failure.



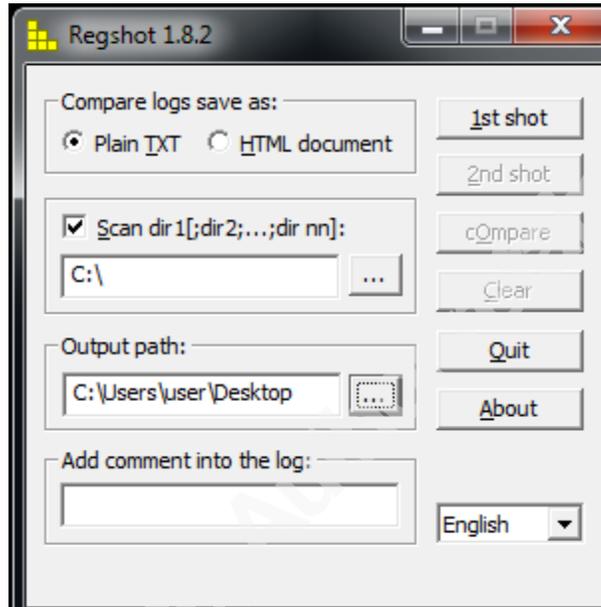
```
Administrator: C:\Windows\System32\cmd.exe - capturebat.exe -n
C:\Program Files (x86)\Capture>capturebat.exe -n
Option: Capturing network packets
Error loading kernel driver: CaptureProcessMonitor - 0x000004fb
Error loading kernel driver: CaptureRegistryMonitor - 0x000004fb
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 1 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 2 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 3 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 4 of 5)
FileMonitor: WARNING - Filter driver not loaded (error: 800704fb) waiting 3 seconds to try again ... (try 5 of 5)
Creating network dumper
Loading network packet dumper
network adapter found: 192.168.1.73
network adapter found: 192.168.0.15
```

The PCAP files are successfully created and if loaded into Wireshark show what appears to be a successful packet capture.

3.7 Regshot – Fully functional

(Regshot, 2013)

Regshot is designed to let the user take a snapshot of the file system and registry, execute a program, take a second snapshot and then compare the two.



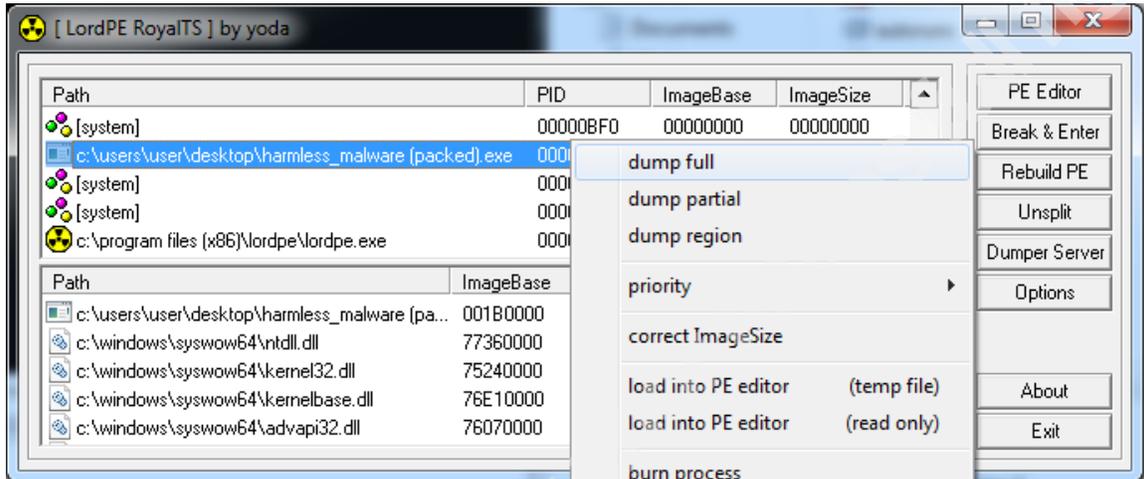
When tested by executing 'Harmless_Malware.exe' between the two snapshots the expected file system and registry changes are successfully captured and this program appears to be fully functional under the Windows 7 x64 environment.

3.8 LordPE – Appears functional

(LordPE, 2013)

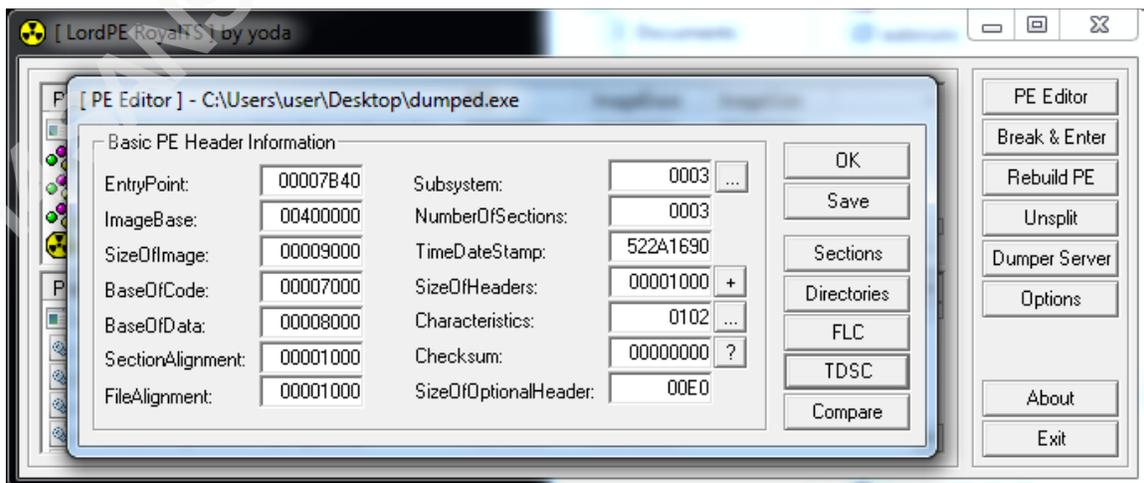
LordPE is designed to dump running processes from memory and edit the PE headers.

As can be seen below LordPE correctly identifies the process and associated DLLs.



The process can be 'dumped' from memory into a file and when the resultant executable file is run through BinText to view embedded strings a similar result to the unpacked executable is seen.

Unsurprisingly the dumped executable will not run, likely due to incorrect headers. We can view and edit the headers but unless you are able to manually repair the file we may not be able to fix it. There are possible limitations in LordPE's ability to repair PE headers in Windows 7.



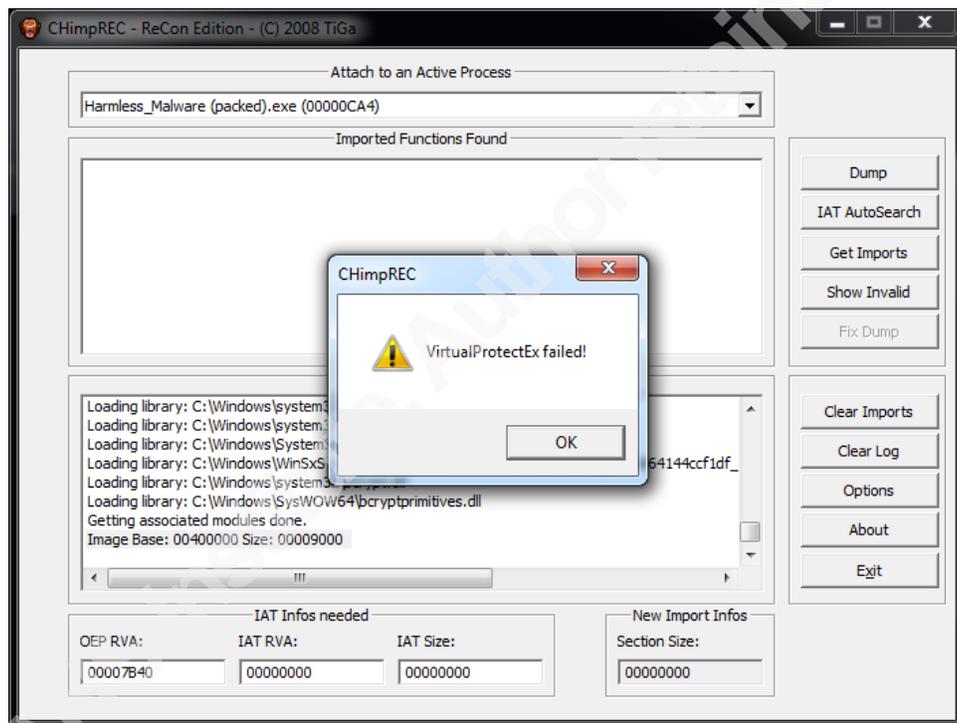
Adam Kramer, adamkramer@hotmail.com

3.9 CHimpREC – Testing failed

(CHimpREC, 2013)

CHimpREC is an alternative to LordPE with the functionality to dump a process from memory to a file and then attempt to fix the headers so the process can be executed.

It comes in two flavors, CHimpREC and CHimpREC 64. During tests of CHimpREC on Windows 7 x64 the test process was identified and able to be selected however when the 'Dump' button was selected the error message “VirtualProtectEx failed!” arose. This occurred whether the process was executed as administrator or not.



Accordingly as this was being executed on a 64 bit operating system attempts were made to run the 'CHimpREC 64' version, however unfortunately the following error occurred prior to any window appearing:

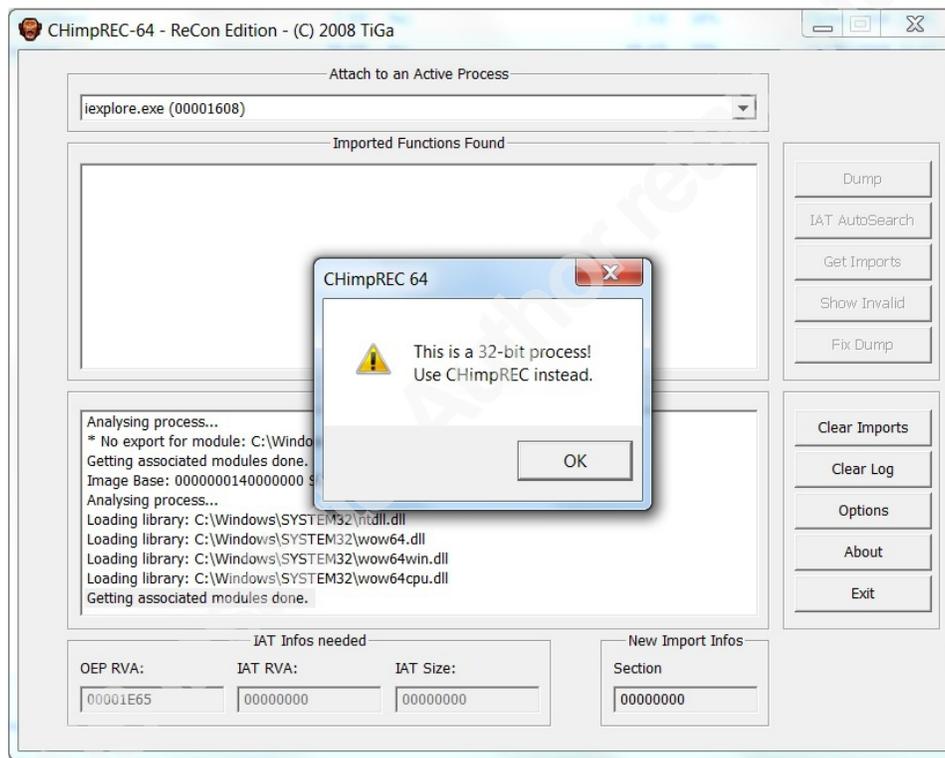
“The application has failed to start because its side-by-side configuration is incorrect. Please see the application event log or use the command-line sxstrace.exe tool for more detail.”

This is likely due to the installed version of the required runtime libraries however after several attempts at various runtime library versions there was no successful execution.

Adam Kramer, adamkramer@hotmail.com

As the side-by-side configuration issue was likely due to the run time libraries installed on that particular machine, the program was run on another system running Windows 7 x64.

In this case, the program ran and loaded, however when a 32 bit process was selected an error message appeared stating “This is a 32-bit process! Use CHimpREC instead.” This is shown in the screenshot below.



On the same system, the 32 bit version of CHimpREC was run and the same program selected.

Unfortunately again in this case the same issue arose and the “VirtualProtectEx failed!” error appeared when a dump was attempted.

Accordingly during the testing this program did not function correctly.

3.10 Ollydbg – Appears functional

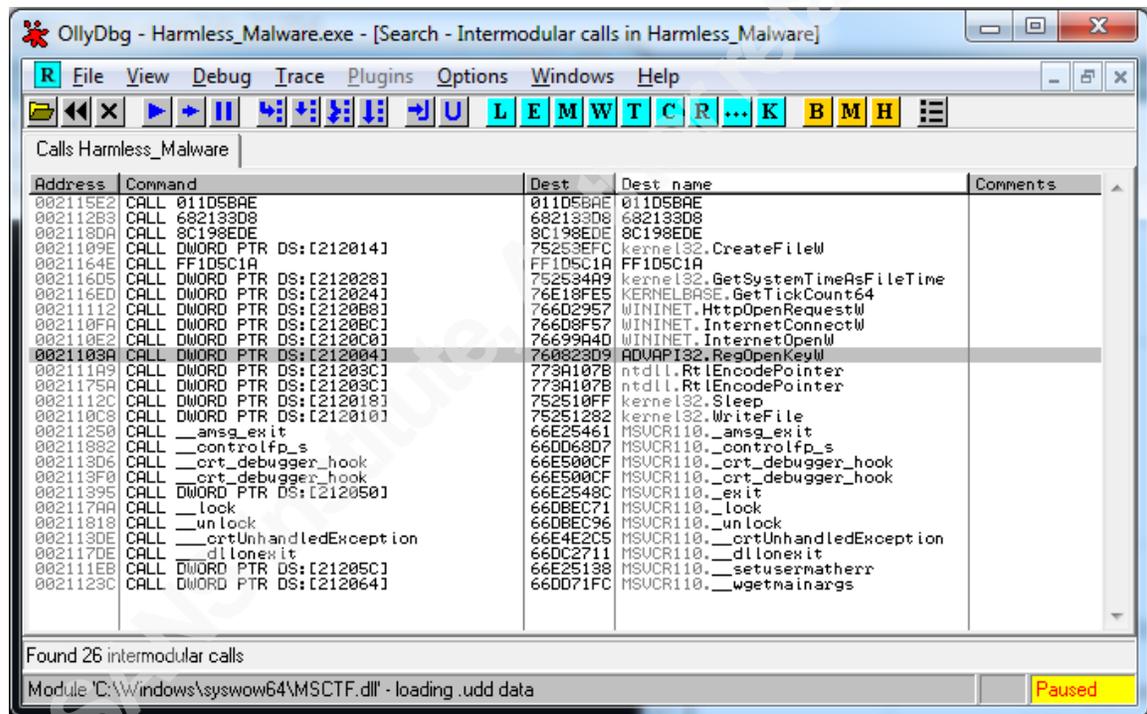
(Ollydbg, 2013)

In the analysis of executable files there are an unlimited number of issues that could be encountered depending on code sequences, hardware and debug settings. Accordingly a number of focused tests were completed to identify whether the core functionality is present.

These tests are based on Ollydbg v2 as v1 causes a number of errors on launch.

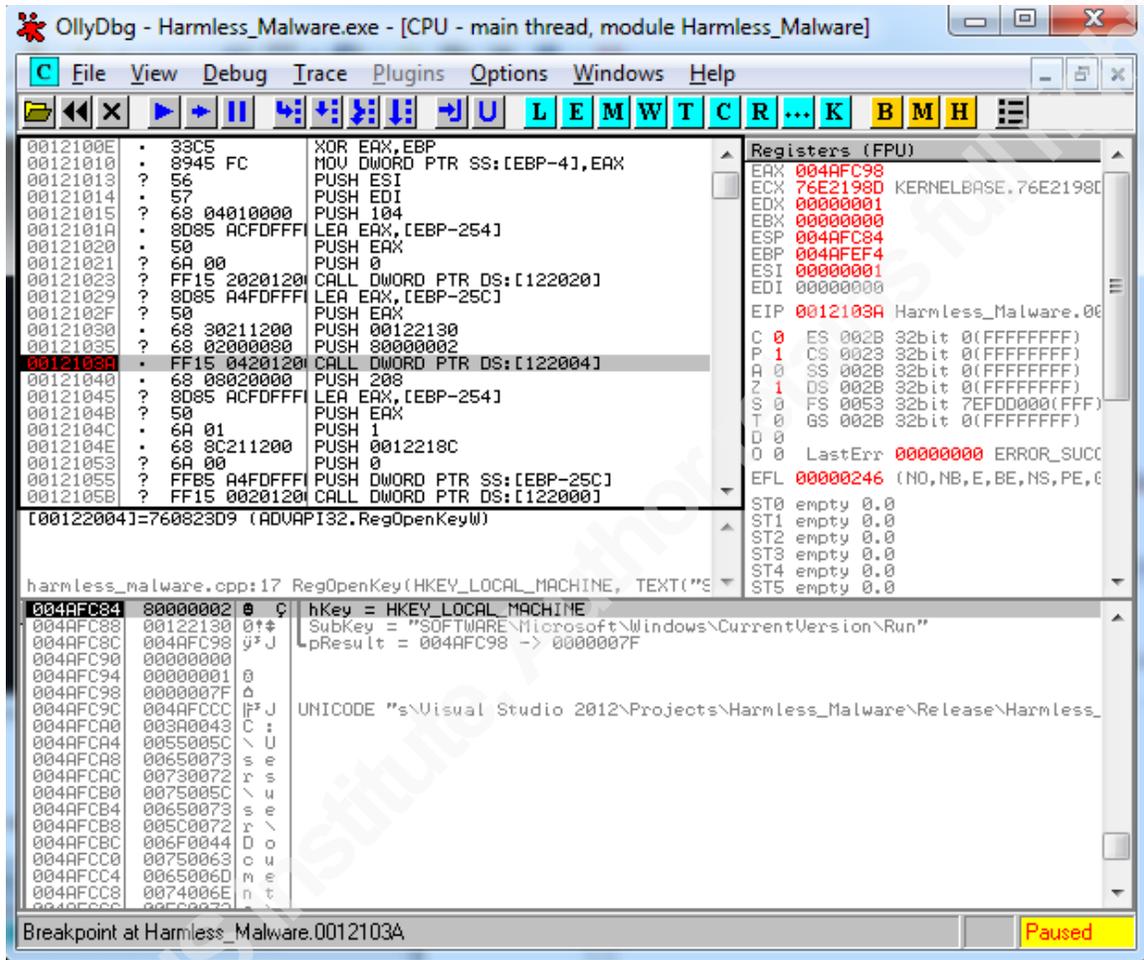
Ollydbg v2 was launched and the test program was loaded in – **Successful**

The intermodular call list was loaded and all expected APIs were identified – **Successful**



Adam Kramer, adamkramer@hotmail.com

A breakpoint was added in the part of code that interacts with the registry and on break the screenshot shows in clear text the registry key to be added – **Successful**

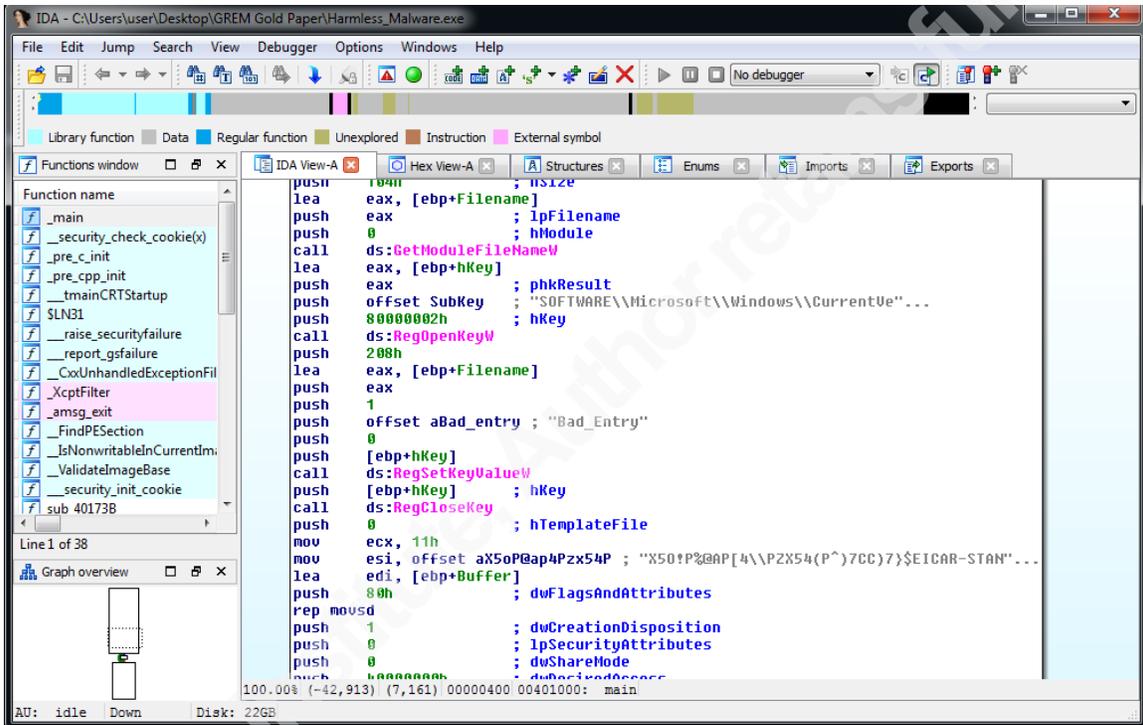


The basic tests have passed, the user is able to load, step through, set breakpoints and analyze various areas of code with access to the memory dump, registers and stack.

There are a number of plugins available that are compatible with Ollydbg v2 including one designed to dump running processes from memory with the ability to set various elements as was available with a similar version 1 plugin.

3.11 IDA Pro – Appears functional (IDA Pro, 2013)

The test executable was successfully loaded into IDA Pro and as can be seen from the screenshot below, a successful disassemble results in the registry key, test file output and expected API calls being available for visual inspection.



As with Ollydbg, there is a vast amount of functionality available in IDA Pro and accordingly a significant amount of further testing on various programs would be required to establish full compatibility.

4. Conclusion

It appears from the tests conducted that the majority of programs are able to run successfully on Windows 7 (x64) when the malicious executable being run has been compiled on x86 architecture as might be expected with an author intending for the widest possible distribution.

The programs which are used for behavioral analysis are the most successful, likely because they are independent of the malicious program being analyzed and are linking directly with the Windows API.

Code analysis being the more complex side of reverse engineering is, by its very nature, more likely to cause unpredictable results, especially in the area of PE header repair such that it can be successfully run on the analysis system.

However having reviewed Ollydbg v2 as a debugger and IDA Pro as a disassembler there is nothing to suggest that a successful code analysis could not take place.

There may be other tools which can be used to dump processes from memory although LordPE appears to have been able to do this successfully, and there are plugins available for Ollydbg v2 to assist with this.

The author believes that there is sufficient functionality available in the tested tools for a thorough analysis to take place in the Windows 7 x64 environment and for the analyst to build a complete tool set and be able to operate successfully.

5. References

SANS Institute (2013). FOR610: Reverse-Engineering Malware Analysis Tools and Techniques - Laptop requirements.

Retrieved from: [http://www.sans.org/course/reverse-engineering-malware-malware-analysis-tools-techniques#section with details laptop description](http://www.sans.org/course/reverse-engineering-malware-malware-analysis-tools-techniques#section%20with%20details%20laptop%20description)

Microsoft (2013). Windows lifecycle fact sheet.

Retrieved from: <http://windows.microsoft.com/en-US/windows/products/lifecycle>

Tech Republic (2013). Latest poll results: What percentage of your enterprise is running Windows XP?

Retrieved from: <http://www.techrepublic.com/blog/windows-and-office/latest-poll-results-what-percentage-of-your-enterprise-is-running-windows-xp/>

EICAR (2003). EICAR anti-malware test file.

<http://www.eicar.org/86-0-Intended-use.html>

MSDN (2013). MSDN library.

<http://msdn.microsoft.com/library>

Microsoft – Dev Center (A) (2013) – Running 32-bit applications.

[http://msdn.microsoft.com/en-us/library/windows/desktop/aa384249\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/aa384249(v=vs.85).aspx)

UPX (2013). Ultimate Packer for eXecutables.

Available from: <http://upx.sourceforge.net>

BinText (2013). BinText - Finds Ascii, Unicode and Resource strings in a file.

Available from: <http://www.mcafee.com/uk/downloads/free-tools/bintext.aspx>

Process Monitor (2013). Windows sysinternals - Process Monitor.

Available from: <http://technet.microsoft.com/en-gb/sysinternals/bb896645.aspx>

Process Hacker (2013). Process Hacker - A free, powerful, multi-purpose tool that helps you monitor system resources, debug software and detect malware.

Available from: <http://processhacker.sourceforge.net/>

Autoruns (2013). Autoruns for Windows.

Available from: <http://technet.microsoft.com/en-gb/sysinternals/bb963902.aspx>

PEiD (2013).

[Official site no longer active - Available from various untrusted sources]

Adam Kramer, adamkramer@hotmail.com

Capture BAT (2013). [No longer available from official site - Available from: <http://zeltser.com/reverse-malware/capturebat.html>]

Microsoft – Dev Center (B) (2013). System Error Codes (1000 – 1299). Retrieved from: [http://msdn.microsoft.com/en-us/library/windows/desktop/ms681383\(v=vs.85\).aspx#ERROR_DRIVER_BLOCKED](http://msdn.microsoft.com/en-us/library/windows/desktop/ms681383(v=vs.85).aspx#ERROR_DRIVER_BLOCKED)

Regshot (2013). Regshot - open-source (LGPL) registry compare utility Available from: <http://sourceforge.net/projects/regshot/>

LordPE (2013). [Official site no longer active - Available from various untrusted sources]

CHimpREC (2013). [Official site no longer active - Available from various untrusted sources]

Ollydbg (2013). OllyDbg is a 32-bit assembler level analyzing debugger for Microsoft® Windows® Available from: <http://www.ollydbg.de/>

IDA Pro (2013). IDA is the Interactive DisAssembler. Available from: <https://www.hex-rays.com>

Appendix A – Source code of Harmless_Malware.exe

COPYRIGHT NOTICE

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

PROGRAM SOURCE – Harmless_Malware.cpp

```
#include "stdafx.h"
```

```
int _tmain(int argc, _TCHAR* argv[])  
{
```

```
    /* Test Sequence: REGISTRY WRITE
```

```
    *****/
```

```
    1. Identify the path to the executable
```

```
    2. Add a registry key for an autorun
```

```
    *****/
```

```
    TCHAR current_filepath[MAX_PATH];
```

```
    HKEY registry_key;
```

```
    GetModuleFileName(NULL, current_filepath, MAX_PATH);
```

```
    RegOpenKey(HKEY_LOCAL_MACHINE,
```

```
    TEXT("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run"),
```

```
    &registry_key);
```

```
    RegSetKeyValue(registry_key, 0, TEXT("Bad_Entry"), REG_SZ,  
    current_filepath, sizeof(current_filepath));
```

```
    RegCloseKey(registry_key);
```

Adam Kramer, adamkramer@hotmail.com

```
/* Test Sequence: FILE WRITE
*****
1. Set a variable to the EICAR test string
2. Open & write to the file (avtest.txt)
*****/
HANDLE output_file;
DWORD bytes_written = 0;
char data_to_dump[] = "X5O!P%@AP[4\\PZX54(P^)7CC)7}$EICAR-
STANDARD-ANTIVIRUS-TEST-FILE!$H+H*";

output_file = CreateFile(TEXT("avtest.txt"), GENERIC_WRITE, 0, NULL,
    CREATE_NEW, FILE_ATTRIBUTE_NORMAL, NULL);

WriteFile(output_file, data_to_dump, strlen(data_to_dump),
    &bytes_written, NULL);

CloseHandle(output_file);

/* Test Sequence: INTERNET CONNECTION
*****
1. Open internet connection to www.sans.org
2. Request the main page
Note: We are not interpreting the response
*****/
HINTERNET internet_session = InternetOpen(TEXT("Suspect File"),
    INTERNET_OPEN_TYPE_PRECONFIG, NULL, NULL, 0);

HINTERNET internet_connection = InternetConnect(internet_session,
    TEXT("www.sans.org"), INTERNET_DEFAULT_HTTP_PORT, NULL, NULL,
    INTERNET_SERVICE_HTTP, 0, 1);

HINTERNET internet_request = HttpOpenRequest(internet_connection, NULL,
    TEXT("/"), NULL, NULL, NULL, 0, 0);

HttpSendRequest(internet_request, NULL, 0, NULL, 0);

/* Test Sequence: 10 MINUTE SLEEP
*****
1. Sleep for 10 mins allowing time to attach
*****/
Sleep(600000);

return 0;
}
```

Upcoming Training

Click Here to
{Get CERTIFIED!}



SANS Security West 2014	San Diego, CA	May 08, 2014 - May 17, 2014	Live Event
Mentor Session - FOR 610	Columbia, MD	May 21, 2014 - Jul 23, 2014	Mentor
Digital Forensics & Incident Response Summit	Austin, TX	Jun 03, 2014 - Jun 10, 2014	Live Event
Community SANS Ottawa	Ottawa, ON	Jun 16, 2014 - Jun 21, 2014	Community SANS
SANSFIRE 2014	Baltimore, MD	Jun 21, 2014 - Jun 30, 2014	Live Event
SANS vLive - FOR610: Reverse-Engineering Malware: Malware Analysis Tools and Techniques	FOR610 - 201407,	Jul 14, 2014 - Aug 20, 2014	vLive
SANS Virginia Beach 2014	Virginia Beach, VA	Aug 18, 2014 - Aug 29, 2014	Live Event
SANS Baltimore 2014	Baltimore, MD	Sep 22, 2014 - Sep 27, 2014	Live Event
SANS DFIR Prague 2014	Prague, Czech Republic	Sep 29, 2014 - Oct 11, 2014	Live Event
SANS vLive - FOR610: Reverse-Engineering Malware: Malware Analysis Tools and Techniques	FOR610 - 201410,	Oct 13, 2014 - Nov 19, 2014	vLive
Community SANS Paris @ HSC - FOR610 (in French)	Paris, France	Nov 24, 2014 - Nov 28, 2014	Community SANS
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced